NO. 55
JUNE 1987

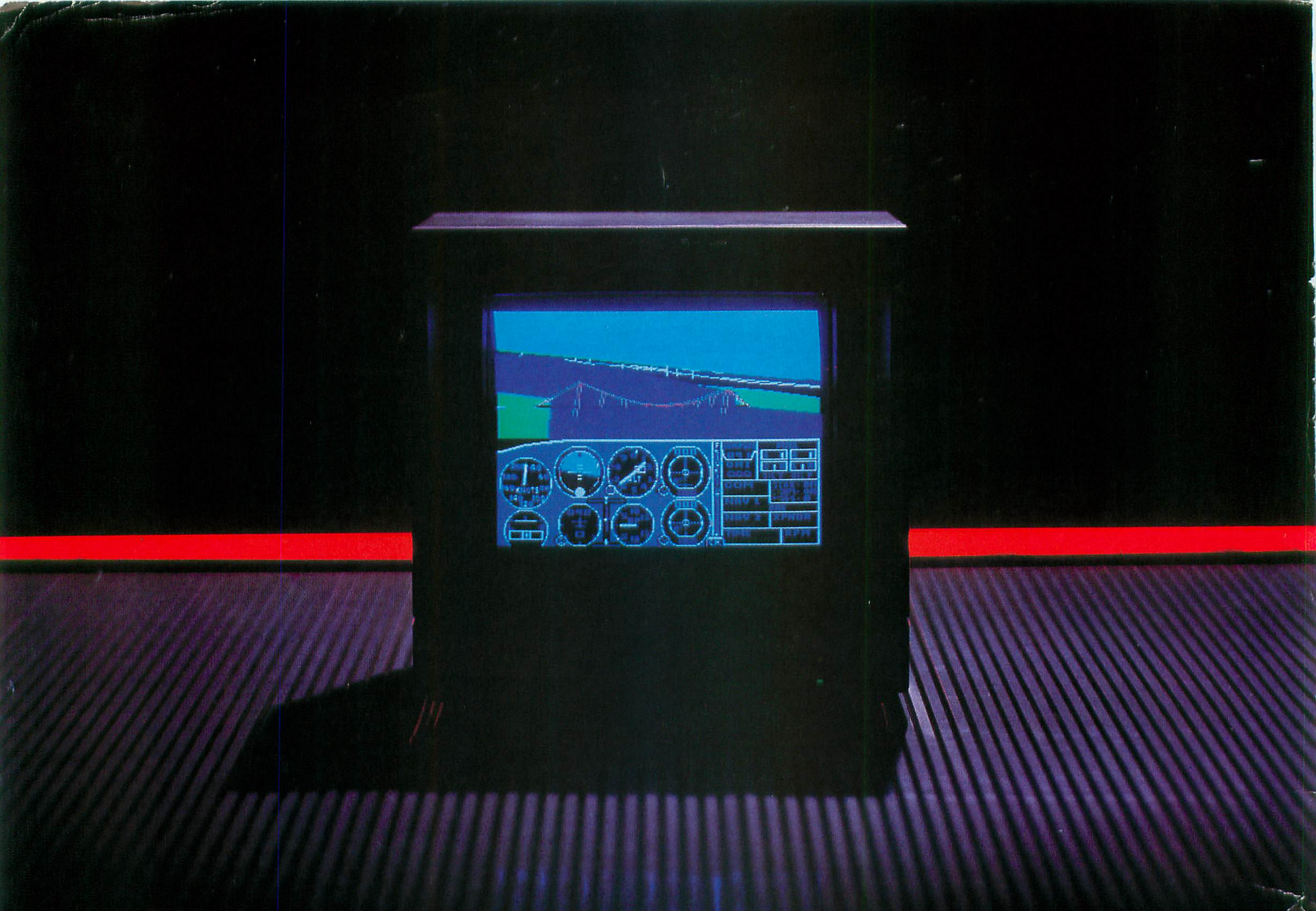
U.S.A. $3.50
CANADA $4.75

THE #1 MAGAZINE FOR ATARI® COMPUTER OWNERS

A.N.A.L.O.G. COMPUTING™

THIS MONTH:
Programming
Issue

FEATURING:
Trade Secrets
Window Graphics
Life in the Fast Lane
Dragonlord Dungeon Editor



• Flight Simulator II
• Scenery Disks

The Challenge of Accomplished Flight

With a realism comparable to (and in some ways even surpassing) $100,000 aircraft flight simulators, Flight Simulator II includes full flight instrumentation and avionics, and provides a full-color out-the-window view. Instruments are arranged in the format standard to modern aircraft. All the radios needed for IFR flight are included. Front, rear, left, right, and diagonal views let you look in any direction. Program features are clearly documented in a 96-page Pilot's Operating Handbook.

For training in proper flight techniques, Flight Simulator II includes another 96-page instruction manual, compiled by two professional flight instructors with over 8,000 hours flight time and 12,000 hours of aviation teaching experience. You'll learn correct FAA-recommended flight procedures, from basic aircraft control through instrument approaches. To reward your accomplishments, the manual even includes a section on aerobatic maneuvers.

The Realism and Beauty of Flight

Go sight-seeing over detailed, realistic United States scenery. High-speed graphic drivers provide an animated out-the-window view in either day, dusk, or night flying modes.

Flight Simulator II features over 80 airports in four different scenery areas: New York, Chicago, Seattle, and Los Angeles. Additional Scenery Disks covering much of the United States are now available for all computer versions of Flight Simulator II. Each disk covers a geographical region of the country in detail, and is very reasonably priced.

The Pure Fun of "World War I Ace"

When you think you're ready, you can test your flying skills with the "World War I Ace" aerial battle game. This game sends you on a bombing run over heavily-defended enemy territory. Six enemy fighters will attempt to engage you in combat as soon as war is declared. Your aircraft can carry five bombs, and your machine guns are loaded with 100 rounds of ammunition.

See Your Dealer. Flight Simulator II is available on disk for the Apple II, Atari 800/XL/XE, and Commodore 64/128 computers for $49.95. Additional Scenery Disks are $19.95 each. A complete Western U.S. Scenery six-disk set is also available for $99.95. For additional product or ordering information, call (800) 637-4983.

Apple II is a trademark of Apple Computer, Inc.
Atari XL and XE are trademarks of Atari Corp.
Commodore 64 and 128 are trademarks of Commodore Electronics Ltd.

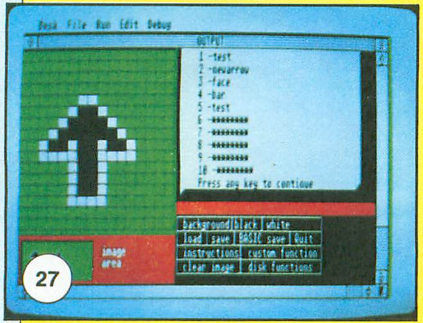
subLOGIC
Corporation
713 Edgebrook Drive
Champaign IL 61820
(217) 359-8482 Telex: 206995
ORDER LINE: (800) 637-4983
(except in Illinois, Alaska and Hawaii)
Open 7 AM to 9 PM Central Time

CIRCLE #101 ON READER SERVICE CARD

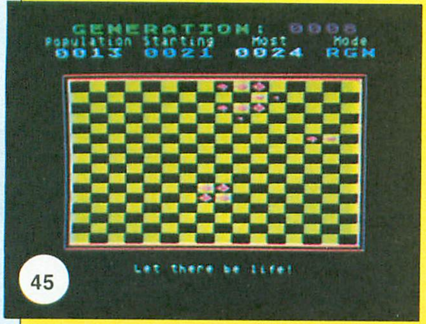
ISSUE 55
JUNE 1987

CONTENTS

FEATURES

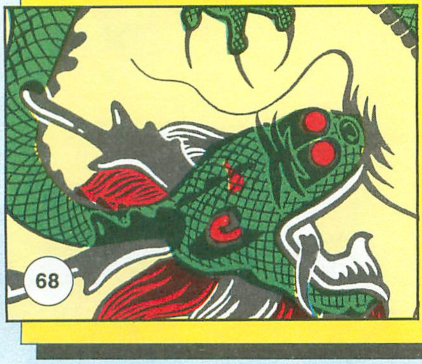

Desk File Run Edit Debug
OUTPUT
background black white
load save BASIC save Quit
instructions custom function
clear image disk functions
image area
27

```
1 -test
2 -newarrow
3 -face
4 -bar
5 -test
6 -********
7 -********
8 -********
9 -********
10 -********
Press any key to continue
```

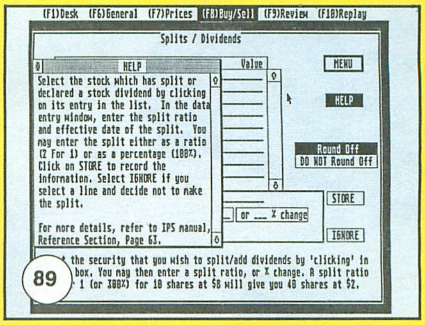


```
GENERATION: 0008
Population Starting   Most    Mode
0013       0021       0024    RGN

Let there be life!
```



REVIEWS



Select the stock which has split or declared a stock dividend by clicking on its entry in the list. In the data entry window, enter the split ratio and effective date of the split. You may enter the split either as a ratio (2 For 1) or as a percentage (100%). Click on STORE to record the information. Select IGNORE if you select a line and decide not to make the split.

For more details, refer to IPS manual, Reference Section, Page 63.

t the security that you wish to split/add dividends by 'clicking' in
box. You may then enter a split ratio, or % change. A split ratio
1 (or 300%) for 10 shares at $8 will give you 40 shares at $2.

COLUMNS

ANALOG Computing (ISSN 0744-9917) is published monthly for $28.00 ($36.00 in Canada, $39.00 foreign) per year by ANALOG 400/800 Corp., 565 Main Street, Cherry Valley, MA 01611. Second class postage paid at Worcester, MA and additional mailing offices. POSTMASTER: Send address changes to **ANALOG Computing**, P.O. Box 625, Holmes, PA 19043. No portion of this magazine may be reproduced in any form without written permission of the publisher (see "Permissions" on staff page).

ANALOG'S BEST!

Over 88 of ANALOG Computing's best and most popular programs available.

GAMES 1
Stuntman
Fill 'Er Up!
Adventure in the
Fifth Dimension
Lumberjack
Space Assault
Darts
Harvey Wallbanger
Supereversion

GAMES 2
Retrofire
Roundup
Livewire
Bricklayer's Nightmare
Knights and Chalices
Air Attack
Avalanche

GAMES 3
Planetary Defense
Crash Dive!
Battle in the B-Ring
Bacterion!
Climber
Money Hungry
Buzz-zap!

GAMES 4
Race in Space
Bopotron!
Bopotron Construction Set
Fire Bug
Basic Burger
Cosmic Defender
Munch'In Clim'In
Shooting Stars

GAMES 5
Adventure at Vandenburg
Popcorn
Demon Birds (Action!)
R.O.T.O. (Action!)
Boulder Bombers
TwoGun
Dragonlord
Lunar Patrol

DISK UTILITIES
Disk Tool (1 and 2)
Burp!
Black Rabbit
Snail
Disk Cataloging Utility
Disk Directory Dump
BASIC Disk Utilities
AlterDOS
Disk Miser

GRAPHICS
Sketch Pad
Graphic Violence!
Multicolor Screen Generator
Moving Missiles in BASIC
Stars 3-D
Bar Chart Subroutine
Solid States
Scredit
Graph E's
P/M Creator/Animator

EDUCATION
Typing Trainer
Observational
Astronomy
What Is It?
Spanish Study Guide
Math Attack
Word Scramble
Typing Evaluator
The Reading Program

UTILITIES 1
System Status
Buncrush
Unicheck
Creating an AUTORUN.SYS
NOREM
Hexpad
mUSE
Sound FX
Disassembler in BASIC
ConTEXT
Default

UTILITIES 2
Create-A-Font
Create-A-Font Datamaker
Fast Repeat Key
H:BUG
Auto Line Renumbering
Binary File Menu Loader
Minicomp
Saturday Night Special
Touch-Tone™ Dialer
Stopwatch

To order, send $9.95 per disk
VISA (plus $2.50 per order shipping and handling) MasterCard
NO CODs ACCEPTED
to:
ANALOG Computing—Disks
P.O. Box 23, Worcester, MA 01603
or call (617) 892-3488

THE #1 MAGAZINE FOR ATARI® COMPUTER OWNERS
ANALOG
COMPUTING

ANALOG COMPUTING STAFF

Editors/Publishers
MICHAEL J. DES CHENES
LEE H. PAPPAS

Managing Editor
DIANE L. GAW

East Coast Editor
ARTHUR LEYENBERGER

Midwest Editor
MATTHEW J.W. RATCLIFF

Contributing Editors
LEE S. BRILLIANT, M.D.
STEVE PANAK
D.F. SCOTT
KARL E. WIEGERS

Contributing Artists
LINDA RICE

Technical Editors
CHARLES BACHAND
CLAYTON WALNUM
DOUGLAS WEIR

Graphics
CONNIE MOORE

Production Editor
KATHY WIESNER

Type Production
EDYTHE STODDARD
LISA OPENSHAW

Advertising Manager
MICHAEL J. DES CHENES

Accounting/Circulation
ROBIN LEVITSKY

Production/Distribution
LORELL PRESS, INC.

Contributors
STEVE ANDERSON
FRANK COHEN
HOWARD GREEN
JAMES HAGUE
DARRYL W. HOWERTON
GREG KNAUSS
MACK McLEOD
SAVEEN V. REDDY

U.S. newsstand distribution by
Eastern News Distributors, Inc.,
1130 Cleveland Rd., Sandusky, OH 44870

ANALOG Computing magazine (ANALOG 400/800 Corp.) is in no way affiliated with Atari. Atari is a trademark of Atari Corp.

WHERE TO WRITE

All editorial material (programs, articles, letters and press releases) should be sent to: Editor, **ANALOG Computing**, P.O. Box 23, Worcester, MA 01603.

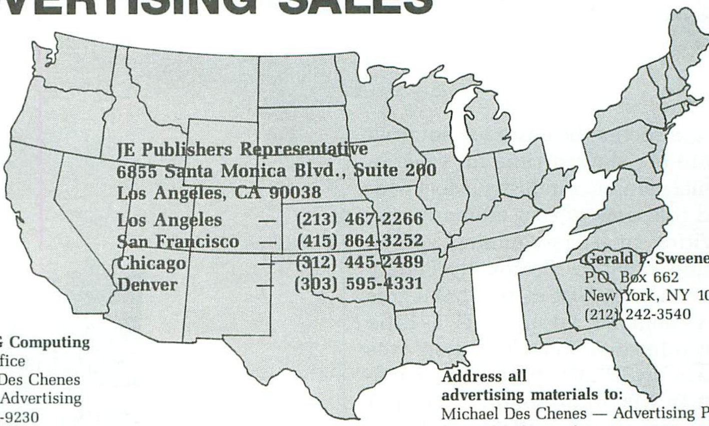
Correspondence regarding subscriptions, including problems and changes of address, should be sent to: **ANALOG Computing**, 100 Pine Street, Holmes, PA 19043, or call 1-800-345-8112 (in Pennsylvania, call 1-800-662-2444).

Correspondence concerning a regular column should be sent to our editorial address, with the name of the column included in the address.

We cannot reply to all letters in these pages, so if you would like an answer, please enclose a self-addressed, stamped envelope.

An incorrectly addressed letter can be delayed as long as two weeks before reaching the proper destination.

ADVERTISING SALES



JE Publishers Representative
6855 Santa Monica Blvd., Suite 200
Los Angeles, CA 90038

Los Angeles	—	**(213) 467-2266**
San Francisco	—	**(415) 864-3252**
Chicago	—	**(312) 445-2489**
Denver	—	**(303) 595-4331**

Gerald F. Sweeney & Assoc.
P.O. Box 662
New York, NY 10113
(212) 242-3540

ANALOG Computing
Home Office
Michael Des Chenes
National Advertising
(617) 892-9230

Address all
advertising materials to:
Michael Des Chenes — Advertising Production
ANALOG Computing
565 Main Street, Cherry Valley, MA 01611

PERMISSIONS

No portion of this magazine may be reproduced in any form, without written permission from the publisher. Many programs are copyrighted and not public domain.

Due, however, to many requests from Atari club libraries and bulletin board systems, our new policy allows club libraries or individually-run BBSs to make certain programs from **ANALOG Computing** available during the month printed on that issue's cover. For example, software from the July issue can be made available July 1.

This does not apply to programs which specifically state that they are *not* public domain and, thus, are not for public distribution.

In addition, any programs used must state that they are taken from **ANALOG Computing** magazine. For further information, contact **ANALOG Computing** at (617) 892-3488.

SUBSCRIPTIONS

ANALOG Computing, P.O. Box 625, Holmes, PA 19043; or call toll-free: 1-800-345-8112 (PA 1-800-662-2444). Payable in U.S. funds only. U.S.: $28-1 yr.; $52-2 yrs.; $76-3 yrs. Canada: $36-1 yr.; $68-2 yrs.; $99-3 yrs. Foreign: $39-1 yr.; $72-2 yrs.; $104-3 yrs. For disk subscriptions, see the cards at the back of this issue.

AUTHORS

When submitting articles and programs, both program listings and text should be provided in printed *and* magnetic form, if possible. Typed or printed text copy is mandatory, and should be in upper- and lowercase, with double spacing. If a submission is to be returned, please send a self-addressed, stamped envelope.

Editorial

Mega ST.

As always seems the case, the summer months are the slowest time for sales in the personal computer industry. Most people spend this time of year enjoying outside activities, such as swimming, sports, cruising and sweating. Personally, I enjoy the winter months just as well. But this past New England winter was a little much, even for me. Shoveling snow was never one of my favorite pastimes—especially not in May! I'm hoping I can put my shovel away now until November.

But back to business. . . Let's not kid ourselves. The releases of new 8-bit software haven't been flowing as they were in the past. I may be wrong, but wouldn't this be the perfect time for the software companies to release new products? Or has everything been written for the Atari that you would want?

I don't think so. I do have most of the software that I'd ever need for my 8-bit and, if I never bought another piece of software, I could be using what I have for at least the next five years. But I'd still plunk down the cash for a new piece of *good* software.

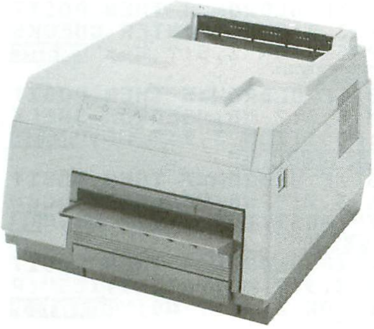
Entry-level PC.

Perhaps it's easier to go with the flow and write software for the computers everyone else is creating for. I suppose it *is* safer. Some companies have the right idea: release their software with flip-side disks. One side is for, let's say, the Apple, and the other for the Atari. They'll sell more, and the cost is much less than separate packaging and marketing.

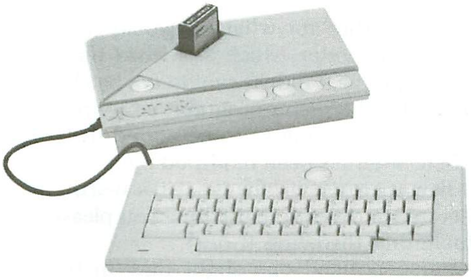
So what do you say, software companies? There are tens of thousands of people out there with the cash, just waiting for your products. I promise I'll make my neighbor buy his own.

Expandable PC System.

What does Atari have in store for us this summer?



SLM Laser Printer.



XE Computer Game System.

We attended the Atari Computer Show which took place at the Novotel Exhibition Complex in London this past April. Because of their enormous success in Europe, the STs were a big attraction again, with a major emphasis on MIDI applications and the three new Mega STs. The new Atari SLM Laser Printer was also shown with the Mega STs as a low-cost, high-quality desktop publishing package. With the growing interest now in MIDI-compatible computers and desktop publishing, Atari should be a leader in these fields if they play their cards right.

The two new Atari PCs—the entry-level system and the expandable system—were also causing crowds to gather. It seems that the success of the STs has overshadowed the old game reputation Atari once had in Europe. Perhaps dealers and distributors here in the States should follow suit.

Of more interest to us 8-bit owners was the new Atari XE computer games system, which can be expanded with joysticks, data recorder (ugh!) and light gun. It is compatible with all existing Atari games software. I didn't hear a mention of one, but hope that you'll be able to hook up a disk drive, also.

Atari realizes that video games are still alive and well. This seems to me like the perfect solution. We already have tons of the best games software. And making it compatible with the XE can only mean more new programs for us.

So let's see what Atari does this summer. From what I've heard, don't be surprised to see an Atari commercial on your TV. 'Bout time!

Michael J. DesChenes

Michael J. DesChenes
Publisher
ANALOG Computing

Reader comment

Missing links.

In the **Atari Picture Storage Techniques** article in **ANALOG Computing**'s issue 50, there was an error in the paragraph immediately underneath the illustration of the **MicroIllustrator** picture format (Figure 1 on page 34). The last sentence in that paragraph should read:

"A special case occurs when these 7 bits equal *0*; then the next 2 bytes are interpreted as the count value in MSB/LSB form."

Also, a couple of sentences were inadvertently left out of the **MicroIllustrator** description, immediately following the above text. Those sentences should read:

"When a **MicroIllustrator** picture is vertically compressed, the screen is scanned in columns from left to right, a byte (4 pixels) at a time. Data is compressed first from the even-numbered scan lines, then the odd."

Sorry for any inconvenience this missing data may have caused anyone!

Charles F. Johnson
Pacific Palisades, CA

Checking modifier.

This is my first letter to your fine magazine, and I would like to say that I have enjoyed it for several years now.

I am writing to thank you for publishing two fine programs in issue 53: **Hard Copy** from Boyd E. Arnold and **CheckWriter** by Jeff Killeen. I've been using **MicroCheck** almost since it was published, and I think it is the finest applications program ever published in any computer magazine.

Now to the "for what it's worth" portion of this letter. Much as I liked these two programs, I didn't want to *not* be able to "END" the menu program, so I wrote a modified Listing 6, to be added to D: MENU after it's been modified by Listing 2. I've enclosed it "for what it's worth." Feel free to do with it anything you please (including throwing it out).

I also noticed one slight flaw in the **CheckWriter** program as published, concerning spacing of the word *and* in the written amount of the check. Deleting Line 702 and changing Line 705 to read:

```
705 MSG$(LEN(MSG$)+1,LEN(M
SG$)+5)=" AND "
```

solved the problem for me.

Again, thank you for publishing an interesting and informative magazine. Keep up the good work.

Sincerely,
Curtis W. Lacey
North Fort Myers, FL

```
20 POSITION 8,1:? #6;"MENU
":POSITION 3,3:? #6;"enter
 checks":POSITION 3,5:? #6
;"SEARCH CHECKS"
30 POSITION 3,7:? #6;"BALA
NCE ACCOUNT":POSITION 3,9:
? #6;"UTILITIES"
40 POSITION 3,11:? #6;"PRI
NT REPORT"
42 POSITION 3,13:? #6;"WRI
TE CHECKS"
44 POSITION 3,15:? #6;"END
"

60 ROW=3:POSITION COL,ROW:
? #6;"=>"
```

```
100 ON CHOICE GOTO 110,120
,130,140,145,147,150
147 RUN "D:CHECKWTR"
150 GRAPHICS 0:END
160 OLDROW=ROW:ROW=ROW+2:I
F ROW=17 THEN ROW=3
190 IF OLDROW=3 THEN POSIT
ION 3,3:? #6;"ENTER CHECKS
":POSITION 3,5:? #6;"searc
h checks"
200 IF OLDROW=5 THEN POSIT
ION 3,5:? #6;"SEARCH CHECK
S":POSITION 3,7:? #6;"bala
nce account"
210 IF OLDROW=7 THEN POSIT
ION 3,7:? #6;"BALANCE ACCO
UNT":POSITION 3,9:? #6;"ut
ilities"
220 IF OLDROW=9 THEN POSIT
ION 3,9:? #6;"UTILITIES":P
OSITION 3,11:? #6;"print r
eport"
225 IF OLDROW=11 THEN POSI
TION 3,11:? #6;"PRINT REPO
RT":POSITION 3,13:? #6;"wr
ite checks"
227 IF OLDROW=13 THEN POSI
TION 3,13:? #6;"WRITE CHEC
KS":POSITION 3,15:? #6;"en
d"
230 IF OLDROW=15 THEN POSI
TION 3,15:? #6;"END":POSIT
ION 3,3:? #6;"enter checks
"
```

Express yourself!

I am writing this letter as an appeal to all Atari 8-bit users and user groups.

Take a look at all of the new software that is being put out for the other 8-bit computer (Commodore), as compared to that for the Atari.

A few years ago, many "computer experts" had Atari dead and buried . . . pre-

Get something Extra!

An Atari 8-bit Extra
from ANALOG Computing

It's a book of some of the best articles and software listings submitted to **ANALOG Computing**—things we just couldn't fit in the monthly magazine pages.

Owners of Atari 8-bit computers will find the **Extra** a must. It gives you games, tutorials, utilities, applications, and more—material you'll want to keep.

$8.95
PLUS $1.50 FOR POSTAGE AND HANDLING

Call toll-free to order the **Extra** today.

1-800-345-8112

When you're looking for the best in Atari—tutorials, games, reviews and programs—look for **ANALOG Computing**.

We're the magazine that always gives you something **Extra**.

Don't miss it!

THE #1 MAGAZINE FOR ATARI COMPUTER OWNERS
ANALOG COMPUTING

maturely. However, Mr. Tramiel provided the needed transfusion, and it appears that Atari is again alive and kicking. But there's not much new third-party software for the 8-bit machines, and there are some great-looking programs out there: Activision-Gamestar—**GFL Championship Football**, **GBA Championship Basketball**; Electronic Arts—**Pegasus**; Epyx —**Summer Games II**, **World Games**, **Winter Games**, **Street Sport Series** and **Sub Battle Simulator**, just to name a few.

Atari users: write these companies and let them know that the Atari 8-bit is alive and well. Let them know that you want third-party support. Write Atari; remind them that the 8-bit computer still exists and there are a lot of users out here. Encourage them to push third-party support for the 8-bit machines.

Steven van Beverhoudt
St. Thomas, V.I.

Star conflict.

I read with interest Steve Panak's review of **Star Raiders II** in issue 47. I was most surprised to read that the review of a follow-up to the great **Star Raiders** should be entrusted to someone who admits to never having played the original version.

It was **Star Raiders** that tipped me into buying a 400 in 1981; I'm sure there were many more like me. The game still offers a tireless challenge after all these years, and it was a brave decision to try to make a follow-up.

However, there are a few points that surprised me. First, after only a few evenings, I was at Admiral status—it took five years to get near a similar level in the original. In an ideal world, the highest grade should be awarded to the player who can achieve the task in minimum time, with minimum damage. In the latest offering, one can gain a higher score and rank by *delaying* the slaughter of alien bases and picking off the onslaught of Zylons as they're created, and running up very high scores on level one. The original required all slaying to be done quickly, using as little energy to reach top grade.

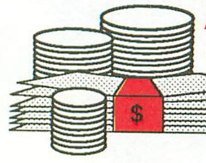
Second, **Star Raiders** (the original) had the joystick response and ultimate playability that only a well-practiced Atarian can understand. Zylons often had to be sought out and destroyed, and one often was attacked from behind *and* in front simultaneously. The new game has far less of this, despite its remarkably improved console graphics.

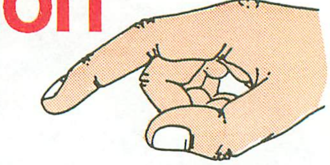
Third—and perhaps least importantly—I really miss those pink, fuzzy photon torpedoes and the two-way intense battle which often ensued.

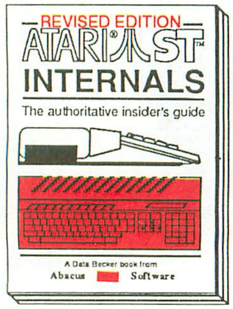
To a new Atari owner, I guess the new game will be more appealing, but can I suggest any prospective buyer has a look—and preferably a feel—of the original before buying the new?

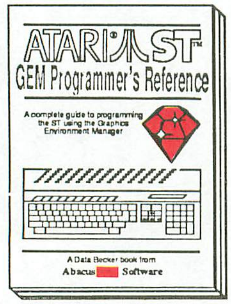
Keep up the good work.
David Rawlings
Aberdeen Scotland, U.K.

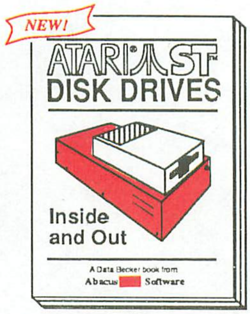
(continued on page 18)

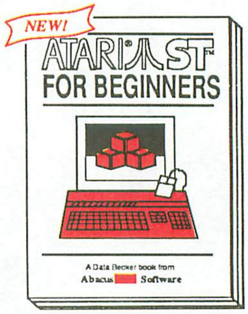


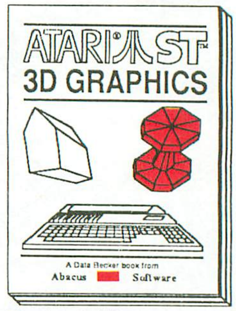





ATARI ST
GEM Programmer's Reference
A complete guide to programming the ST using the Graphics Environment Manager
A Data Becker book from
Abacus Software







The making of AtariWriter Plus

An inside look at the rocky road to stability.

by Frank Cohen

Back in the late 1970s, if you had an interest in microcomputers, you could easily be considered a pioneer. There were no high school computer classes, and personal computers were limited to those few who could afford an Apple II with 16K of memory at $1300.

William Robinson was one of these pioneers. By the time he was thirteen, William was already writing simple BASIC programs for the Commodore Pet. He later went on to learn how the new Atari 800 worked.

School offered him very little in satisfaction or achievement, whereas computers had already given him a job—at a new company called Programma International. Riding his motor scooter through the Los Angeles business district, William would drive thirty miles from his home in Malibu, to work after school.

Those were the days when developing a new product meant writing a 4K game in BASIC. William wrote the products Programma put on the market. The market consisted of a small string of computer specialty stores and direct mail-order sales. Personal computers were still mostly unheard of, but that was changing.

Within less than a year, Programma ceased to exist. Most of the people at Programma went on to establish new, larger companies. David Gordon, half-owner of Programma, later founded the Los Angeles based, multimillion-dollar software and computer book publishing company we know as Datamost.

By that time, William was avoiding high school as much as possible. His friends became his biggest link with the school. And English rapidly became his friends' best subject. . . but not because of their writing skills.

Beginnings.

One of William's first programs was a simple offering that let you type text into a Commodore Pet and print it out onto a computer printer. The concept of a word processor was still limited to mini-computers.

Classroom English assignments given to William's friends meant they would be busy that night, working on their papers. The assignments were usually three pages of typewritten discussion. William found that his friends had completed only two pages of their assignments the night before the papers were due. The solution occurred to him when he found that he could change a printout's line spacing (the amount of space between the lines of text) to make two pages of text fill three pages —and still *look* double spaced.

On another occasion, even less text had been finished for an assignment due the next day. William found that, with certain daisy-wheel printers, he could vary the number of characters per inch. Typically, these printers yield ten characters per inch. By reducing the number to eight, each line of text on a page holds less text, so a two-page document becomes a three-page document. And the amount of text needed to finish a paper grew even lower when headers and footers were added to every page.

The little document-printing program grew and grew, until eventually it became a fairly powerful text editor. Later, William was to meet Pat Ketchum, owner of a small software company called DataSoft. And **Text Wizard 1** was released within a couple of months of their getting together.

The meaning of success.

That small company found, in a matter of months, that it had marketed the leading word processor for the new Atari 800. DataSoft had its first money-making product, after failing in the mini-computer software market and making an unsuccessful foray into the educational market with such products as **Maxwell's Demon** and **Bishop's Square**.

By this time, William was sixteen years old—and receiving royalty checks each month for **Text Wizard**. He began work on **Text Wizard 2**.

The meaning of success was very clear: he bought a new car and moved out of his parent's house. The freedom his little word processor had brought him was unexpected.

Neither DataSoft nor William ever realized the full extent of **Text Wizard**'s success, for Atari was now interested in the word processor. Soon, William would be hard at work on another new product, **AtariWriter**.

The making of AtariWriter Plus *continued*

From muddled beginnings.

Under the terms of the agreement Pat Ketchum had negotiated with Atari, William would develop Atari's new word processor and receive the royalties for his work. DataSoft was the communications link between William and Atari, through Gary Furr, an Atari product manager.

This was the old Atari, full of corporate overhead and extremely difficult to work with. In due course, the company developed a document called the "AtariWriter Internal Design Specification." Over 300 pages thick, the IDS document described in exacting detail *everything* Atari wanted in its new word processor.

The IDS was filled with so many new functions that Mark Rieley, one of DataSoft's programmers, was brought in to develop part of the new program code. Mark and William were already friends and shared a common bond: they weren't certain of the company's stability.

By this time, DataSoft had five programmers working in their Northridge office. These programmers developed DataSoft's first "arcade-style" games. Ron Rosen created **Pacific Coast Highway**, a **Frogger** look-alike; Ralph Burris and James Garron designed the TRS-80 color computer version of **The Sands of Egypt**, later to be written on the Atari 800; Mark Rieley wrote **Shooting Arcade**; and this writer developed **Clowns & Balloons**.

The programming group at DataSoft was very close, bound by a common dislike for the group manager. They were also suspicious about the business dealings behind DataSoft, which led to a very strained situation.

Though William was an outside developer, he began working in DataSoft's office when the **AtariWriter** development schedule began to be pushed back. The friction between DataSoft and William grew more and more intense as the development schedule was delayed, and as Atari consistently came up with more functions to be added.

The internal pressure at DataSoft exceeded the bounds of normalcy the week before the company moved into its new, larger offices in Chatsworth. Four of the six programmers quit. Ron Rosen and Mark Rieley left to start another software company called Everyware, with DataSoft's ex-president Gary Koffler. Frank Cohen went to work for Fox Video games, developing a **M∗A∗S∗H** cartridge, and Ralph Burris became an editor for a technical book publisher.

The **AtariWriter** program was completed shortly after this shakeup, and Atari began producing cartridges. Exactly 80,000 cartridges were produced before the "back doors" were found. William and Mark had added some special effects to **AtariWriter**, which could be seen when certain pins on the joystick ports were shorted together. A space game, color scroll, sound effects demo and other such unusual features (for a word processor, at least) were included.

Atari immediately threatened to sue DataSoft for damages. DataSoft discontinued royalty payments to William, who filed suit against DataSoft for late royalty payments on both **Text Wizard** and **AtariWriter**. A month after his eighteenth birthday, William walked into a local General Motors showroom and drove away with a new 1984 Corvette. He paid cash.

The friends get together.

Led by Gary Koffler, the original DataSoft programmers—Ron Rosen, Mark Rieley and William Robinson—joined forces to start a company called Everywhere. Their impetus was to develop new and original arcade-style games and productivity software for the emerging home computer industry.

The idea was basically sound, but the market was still not ready for software-only development houses. Financial backers of computer companies had seen the recent destruction of the home games market, with the release of the **ET** game cartridge on Atari's VCS 2600. The marriage of retailers, distributors and computer software houses abruptly ended, after the dismal showing of what had promised to be the most successful home computer game yet.

Eventually, the founders of Everywhere went their own ways, developing products independently for established companies and foreign markets. For the most part, the software markets in England and Europe kept these developers alive economically. Atari was sold to the Tramiels shortly after this low period for home computers. Something new was needed to keep the home computer market going.

The Tramiels began development of the new 16-bit line of ST computers, relying heavily on income from the older 8-bit line. The decision was made to release a new version of **AtariWriter**, to renew interest in the established 8-bit machines.

Ron Rosen and William Robinson combined their programming talent in a new company, Microfantasy. The company was involved with the development of games for the Atari and Commodore computers. **Mr. Robot and His Robot Factory**, released through Datamost, was one of their more well known products.

By this time, Ron Rosen had become a formidable negotiator, and landed the **AtariWriter Plus** contract. William was to develop the basic word processor, Ron the mail-merge database, and Stan Kistler the spelling checker.

William was now nineteen, an acknowledged professional in word processing. **AtariWriter Plus** was designed from scratch, and included word count, 200-column editing, double-column printing, and many more advanced functions never before implemented on a home computer.

AtariWriter Plus was developed over eight months, with another four months to integrate the spelling checker and database. The finished product was released on disk, so the space limitations of the original **AtariWriter** cartridge were not a problem.

AtariWriter Plus was completed in the summer of '84 and hit the market in mid-1985. Today, the Atari 8-bit market, extending into several million machines, is still very vital. At this writing, William had begun the new version of **AtariWriter**, which will work with the new 80-column adapter.

The Future of *Text Wizard.*

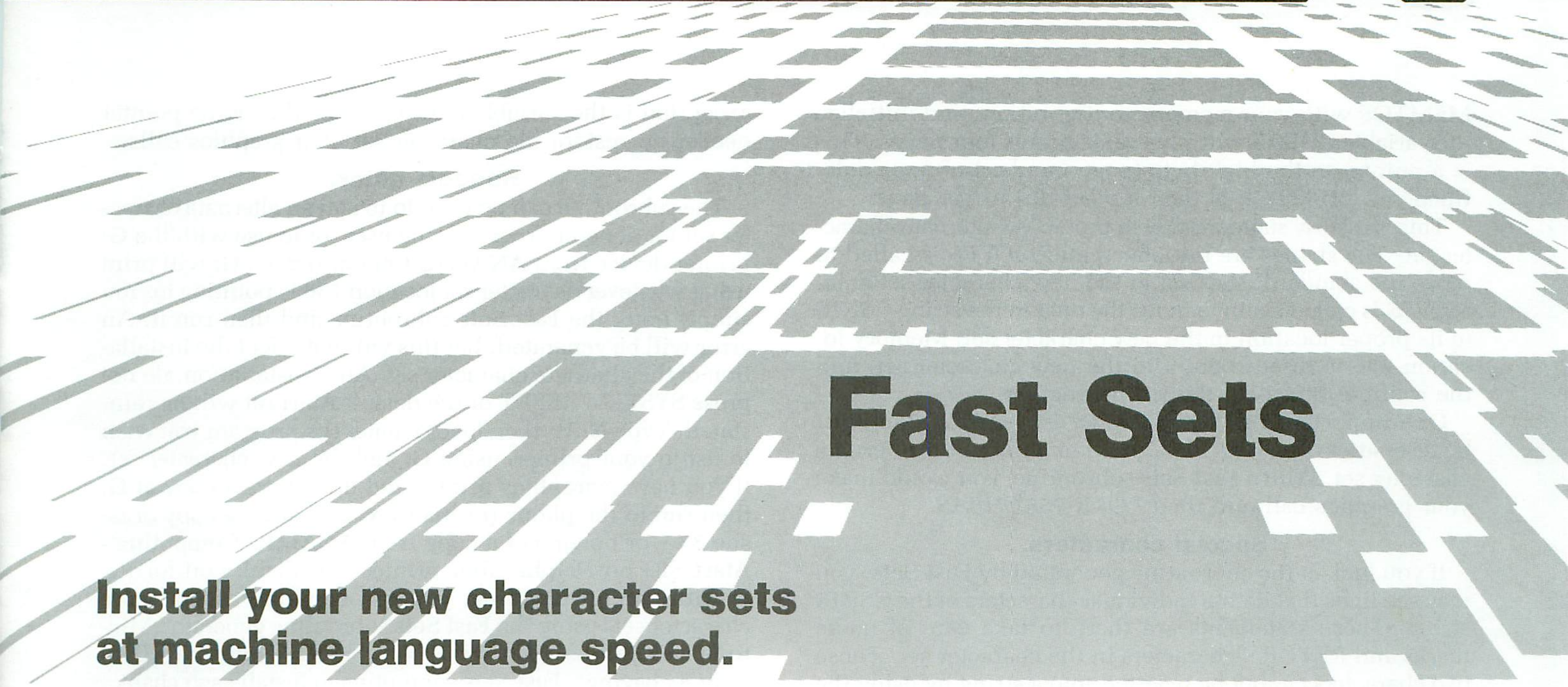
William Robinson founded a company called Regent Software little more than a year ago, developing two word processors for the new ST computers.

Regent Word II, his latest version, could also be considered "Text Wizard 8." William is now working on his new word processor, **Regent Word III** (really "Text Wizard 9"). The new program uses Macintosh fonts and will be a what-you-see-is-what-you-get word processor.

The mark of a good programmer seems to be consistency. The **Text Wizard** word processors all seem to have a basic design philosophy about them: what's easy to use sells. In their time, all of the **Text Wizards** have been easy to use and state-of-the-art. Who knows, by the time he hits twenty-five, William's "Text Wizard 800" might be finished.

*Frank Cohen, author of **Regent Base**, has been publishing software for the Atari line since his first program, **Clowns & Balloons**, in 1982.*

[Readers should note that DataSoft/Intellicreations is now under wholly new management from that described in this article. —Ed.]



Fast Sets

Install your new character sets at machine language speed.

by Darryl W. Howerton

Imagine installing an entire new character set at machine language speed in a BASIC program. With **Fast Sets**, you can generate a subroutine that will install a new character set in your BASIC program in under 1 second.

Standard methods.

The usual ways to install a new character set in BASIC are: to read the information as data and use POKE to put it into memory, or to have the set read into memory from a disk file at program run time. While both of these are acceptable, each has faults. POKEing data in is painfully slow if used for more than a few characters. Reading a disk file requires loading an entire character set from disk, whether you need it or not. The disk file method also requires the main program disk to be in the drive each time the program is run.

A routine not commonly used is to store the redefined characters in a string and use machine language to relocate the data. This is not only very fast, but uses less memory than comparable data statements—and no disk access is required.

Getting started.

Listing 1 is the **Fast Sets** subroutine generator program. Type this in, checking it with **BASIC Editor II** (from issue 47), then save it to disk.

Now type in Listing 2 and save it. When run, Listing 2 will create a disk file called LINES.LST, machine language strings to be merged with Listing 1—specifically, Line 5030 and Lines 32010 to 32050. LINES.LST must be entered into the main code for **Fast Sets**.

Listing 3 is the source code for the string MOVE$, the RAM mover routine used to relocate the standard and redefined character sets in **Fast Sets** and the subroutine it generates. You do *not* need to type this in for **Fast Sets** to work.

Theory of operation.

Fast Sets loads a redefined character set you have created (or borrowed) and makes a BASIC subroutine for you to include in your own programs. The subroutine is completely self-contained and is called with a simple GOSUB command.

Since a character set must begin on the page boundary of a 1K block of memory, storage in a string can be difficult; the string would need to be relocated to a page boundary before it could be used. Methods to move the string itself to a specific memory location tend to be rather cumbersome to use. **Fast Sets** overcomes this by moving just the string data to a page boundary, through the use of a machine language routine stored in the string MOVE$.

MOVE$ is a completely relocatable routine to move RAM. The calling parameter to use MOVE$ in your own programs is: X=USR(ADR(MOVE$),FROM,TO,BYTES), where FROM is the address of the RAM to move, TO is the address where you want the RAM to move, and BYTES is the number of bytes to move. This routine moves memory 1 byte at a time, by starting at the bottom of the FROM location and working its way up. Because of this, the TO location *must* be higher than the FROM location plus the BYTES. Otherwise, you will overwrite the area you're attempting to move. This problem does not occur when you're moving RAM down in memory.

How the subroutine works.

A full character set requires 1024 bytes of RAM. Since a page of memory is 256 bytes, a character set needs four pages reserved for it. In addition, certain combinations of GRAPHICS calls and the clear screen command can cause a portion of memory above MEMTOP to be cleared. For this reason, an extra page of memory is needed to protect the character set from this phenomenon.

The **Fast Sets** subroutine will determine the location of

MEMTOP with your memory configuration, and initialize the variable CHBAS with this value minus four pages. This is a pointer to the high byte (page number) of your redefined set. MEMTOP is then moved down five pages.

This is a fairly standard way to protect an alternate character set. The strings are initialized and MOVE$ is called to move the standard Atari set to the new character set area. MOVE$ is called again to move the data in the string CSET$ to its proper location in the new character set. Memory location 756 is reset to point to the new character set, and the routine returns to the main program.

One important rule to remember is: location 756 *must* be reset after each GRAPHICS call to point to the alternate character set. With a **Fast Sets** subroutine, you would make your graphics call and then *POKE 756,CHBAS*.

Special characters.

If you look at the subroutine generated by **Fast Sets**, you may see lines that define individual characters of the CSET$ string. These statements are there to take care of quote marks and RETURN characters in the character set. These two characters cannot be inserted into a string by using the ESCAPE key as the other "active" keys can.

When **Fast Sets** runs into a quote mark or RETURN code in the character set data, an array variable is set, defining the position in the string and the type of data. A space is substituted for the character in the string CSET$. The necessary lines are generated, so the correct data will be inserted into the string when the subroutine is called.

Using the program.

Fast Sets is self prompting and contains extensive error trapping. By following the screen prompts, you shouldn't have any trouble. It isn't necessary to use the D: prefix on your files, unless you're using a drive other than D1.

The alternate character set you load is displayed at the bottom of the screen. The choices of lines of the character set refer to the lines of characters displayed here. Line 1 would be the top line, Line 2 the second line, etc. Press the number corresponding to your choice. Do *not* press RETURN. If you should happen to make a mistake and need to make changes, press SYSTEM RESET and run the program again.

Using the subroutine.

The subroutine generated by **Fast Sets** should be merged with the program with which you want to use the alternate character set. First load your program, then use *ENTER D:filename.ext* to merge the **Fast Sets** subroutine with your program.

Before you can use the alternate character set, you must *GOSUB lineno*, where *lineno* is the starting line of your subroutine. The alternate character set will be moved and installed.

If your program does a graphics call after the **Fast Sets** subroutine has installed the alternate character set, you must *POKE 756,CHBAS* to restore the alternate character set. Otherwise, the standard Atari set will be re-enabled.

Both the scalar and string variables used by the **Fast Sets** subroutine may be used by your program, as long as the subroutine is called *before* your program uses them. The exception is the variable CHBAS, since this is the pointer needed to restore the character set after graphics calls.

Random notes.

The subroutine can be used to install an alternate character set for general programming uses, or to use with the **G:** printer device from **ANALOG Computing** 35. **G:** will print using whatever character set location 756 is pointing to. You simply enter the **Fast Sets** subroutine and then run it. An error will be generated, but this will not affect the installation of the alternate character set. After installation, *do not* press SYSTEM RESET or the default Atari set will be reinstated. Type *NEW*, then load or enter the program you wish to list to your printer, using **G:** and the new character set. If you have a graphics printer and don't have a copy of **G:** then run to the phone (or mailbox) and order a copy of issue 35—or download a copy from **ANALOG Computing**'s Atari SIG on Delphi! Your printer will thank you for it.

With careful planning, you can combine two or more character sets using the **Fast Sets** subroutine generator. Multiple sets can be used with a Display List Interrupt (DLI) routine, having a **Fast Sets** subroutine to install each character set. The value in CHBAS would need to be stored in the memory location where your DLI expects to find it, instead of in the default CHBAS location of 756. You would also need to remove the portions of the *first* line of the subroutine that dimensions the strings and the line that defines MOVE$. This should be done on all but the *first* **Fast Sets** subroutine your program calls. On all but the *last* **Fast Sets** subroutine your program calls, the statement *POKE 106, CHBAS-1* in the first line of the subroutine should be reentered as *POKE 106,CHBAS*.

Segments of up to four character sets can be stored in the space of one set with the following steps.

(1) In the first **Fast Sets** subroutine called, remove the GRAPHICS 0 call from the first line of the subroutine and the POKE 756,CHBAS from the last line.

(2) In *all* **Fast Sets** subroutines called *after the first*, remove the *first* and *second* line of the subroutine. Also remove the line that relocates the Atari set in memory (the first USR call) and the POKE 756,CHBAS in the last line. These lines contain the initialization portions of the subroutine and are only needed once.

(3) In the *last* subroutine called, make the *last* line: *lineno GRAPHICS 0:POKE 756,CHBAS:RETURN.*

If you did everything right, you now have parts of up to four character sets installed as one.

The **Fast Sets** subroutine is automatically numbered in increments of 10. This can be changed by altering the increment value in Line 2000 from *10* to whatever you would rather use. Line number overflow is checked for, to prevent accidental errors.

Fast Sets supports drives 1, 2 and 8. If your system has more, or if you wish to use a hard disk, the program can be easily modified for your configuration.

Darryl W. Howerton, with his B.S. in Communications, is the Control Room Supervisor for KAIT-TV, an ABC affiliate. He has owned and programmed an Atari 800 for the past four years, first in BASIC and lately with Action! He *also does some assembly language programming with the Atari Assembler/Editor cartridge.*

The two-letter checksum code preceding the line numbers here is *not* a part of the BASIC program. For further information, see the *BASIC Editor II*, in issue 47.

Listing 1.
BASIC listing.

```
OR 1 REM *** FAST SETS SUBROUTINE WRITER
JG 2 REM *** BY: DARRYL W. HOWERTON
XV 3 REM *** SEPTEMBER 1986
NJ 4 REM
WP 25 GOSUB 32000:GOSUB 5000:GOTO 100
YX 50 FOR X=5 TO 18:POSITION 2,X:? SP$;:S
   OUND 0,40,10,18-X:NEXT X:SOUND 0,0,0,0
   :RETURN
WO 100 NUM=1024:FIN=NUM:BEG=OFF:OFFSET=ON
   :GOSUB 50:POSITION 2,6
JL 110 POSITION 4,6:? "SELECT A CHARACTER
    SET TO LOAD":LINE=100:GOSUB 1000
CP 120 ? :? " LOADING=> ";FN$;"<<<<<<<<<<
   <<<<<<<<<<<";
JJ 130 OPEN #1,4,0,FN$:GOSUB 7000:CLOSE #
   1
XL 200 GOSUB 50:POSITION 3,6:? "SELECT A
   NAME FOR YOUR SUBROUTINE":LINE=200:GOS
   UB 1000
HP 220 GOSUB 50:POKE CURSOR,ON:POSITION 2
   ,6:? "STARTING LINE NUMBER? ";:INPUT #
   16,F$:IF F$="" THEN 220
FP 230 POKE CURSOR,OFF:TRAP 6000:LINE=220
   :LINO=VAL(F$)
JP 240 IF LINO>32000 THEN ? :? :? "▣CAN N
   OT BE LARGER THAN 32000!":GOTO 6090
BT 250 IF FLAG THEN 300
AM 260 ? :? " USE:   1. LINE 1":? "
     2. LINE 2":? "           3. LINE 3":? "
           4. LINE 4"
EQ 267 ? "           5. TOP 2 LINES":? "
        6. BOTTOM 2 LINES":? "           7. W
   HOLE CHARACTER SET"
YF 270 ? :? "   ENTER THE number OF YOUR 
   CHOICE ";:GET #2,A
WK 280 IF A=49 THEN NUM=256:FIN=NUM
PK 284 IF A=50 THEN NUM=256:BEG=BEG+NUM:F
   IN=FIN-512:OFFSET=NUM
QN 288 IF A=51 THEN NUM=256:BEG=BEG+512:F
   IN=FIN-NUM:OFFSET=512
XU 290 IF A=52 THEN NUM=256:BEG=BEG+768:O
   FFSET=768
JY 294 IF A=53 THEN NUM=512:FIN=FIN-NUM
UP 298 IF A=54 THEN NUM=512:BEG=BEG+NUM:O
   FFSET=512
PM 300 GOSUB 50:POSITION 3,10:? "INSERT D
   ISK TO WRITE SUBROUTINE TO":POSITION 8
   ,12:? "PRESS return WHEN READY"
EM 302 GET #2,A:IF A<>155 THEN 300
FP 303 GOSUB 50:POSITION 8,10:? "...WRITI
   NG SUBROUTINE...":POSITION 12,12:? "PL
   EASE STAND BY!"
DD 305 TRAP 6000:LINE=300:OPEN #1,8,0,FN$
LJ 310 ? #1;LINO;" MEMTOP=PEEK(106):CHBAS
   =MEMTOP-4:POKE 106,CHBAS-1:GRAPHICS 0:
   DIM MOVE$(43),CSET$(";NUM;")"
FX 320 GOSUB 2000:? #1;LINO;" MOVE$=";CHR
   $(34);MOVE$;CHR$(34)
GY 330 FOR X=BEG TO FIN STEP 64:GOSUB 200
   0:? #1;LINO;" CSET$(";X-OFFSET;")=";CH
   R$(34);
LH 340 FOR J=0 TO 63:K=PEEK(CHBAS+J+X-1):
   POKE CHBAS+J+X-1,255:POKE CHBAS+J+X-1,
   K
XG 345 IF K=34 OR K=155 THEN FIX(0,FIX)=J
   +X-OFFSET:FIX(1,FIX)=K:FIX=FIX+1:K=32
WU 350 PUT #1,K:NEXT J:? #1;CHR$(34):NEXT
    X
NN 360 IF  NOT FIX THEN 400
AU 370 FOR X=0 TO FIX-1 STEP 4:GOSUB 2000
   :? #1;LINO;" ";
GZ 375 FOR J=0 TO 3:IF J+X>FIX-1 THEN J=4
   :GOTO 389
PY 380 ? #1;"CSET$(";FIX(0,X+J);",";FIX(0
   ,X+J);")=CHR$(";FIX(1,X+J);")";
BQ 385 IF J<3 AND J+X<FIX-1 THEN ? #1;":"
   ;
UN 389 NEXT J:? #1
MF 390 NEXT X
XT 400 GOSUB 2000:? #1;LINO;" X=USR(ADR(M
   OVE$),57344,CHBAS*256,1024)"
OA 410 GOSUB 2000:? #1;LINO;" X=USR(ADR(M
   OVE$),ADR(CSET$),CHBAS*256";:IF BEG>1
   THEN ? #1;"+";BEG-1;
VZ 420 ? #1;",";NUM;")":GOSUB 2000:? #1;L
   INO;" POKE 756,CHBAS:RETURN"
LG 430 CLOSE #1
MD 900 GOSUB 50:POSITION 15,10:? "finishe
   d!"
SK 910 POSITION 3,16:? " PRESS ANY KEY TO
    RE-RUN ▌ q=quit "
IG 920 IF PEEK(764)=255 THEN 920
RK 930 IF PEEK(764)=47 THEN POKE 764,255:
   POKE 106,PEEK(208)+4:GRAPHICS 0:TRAP 9
   50:RUN "D:MENU"
PO 940 X=USR(ADR(MOVE$),57344,CHBAS,1024)
   :GOSUB 5100:GOTO 100
OI 950 END
PO 1000 TRAP 6000:? "(DRIVE #1 ASSUMED UN
   LESS SPECIFIED)":POSITION 2,15:? "PRES
   S: return FOR DIRECTORY LISTING":?
LS 1010 FN$="":POKE CURSOR,ON:POKE 764,25
   5:POSITION 2,10:? "ENTER:":? :? "    F
   ILENAME.EXT=> ";:INPUT #16;F$
QQ 1020 IF F$="" THEN POP :GOTO 1100
CA 1030 IF F$(1,2)<>"D:" AND F$(1,3)<>"D1
   :" AND F$(1,3)<>"D2:" AND F$(1,3)<>"D8
   :" THEN FN$="D:":FN$(3)=F$:GOTO 1050
GQ 1040 FN$=F$
LF 1050 IF FN$(2,2)<"1" OR FN$(2,2)>"2" A
   ND FN$(2,2)<>":" AND FN$(2,2)<>"8" THE
   N POKE 195,160:GOTO 6000
EF 1060 POKE CURSOR,OFF:? :RETURN
LZ 1100 POKE CURSOR,OFF:GOSUB 50:POSITION
    6,9:? "DIRECTORY FOR WHICH DRIVE?":PO
   SITION 13,12:? "[1, 2, OR 8]"
BV 1105 GET #2,A:IF A<49 OR A>56 THEN A=4
   9
YH 1110 F$="D :*.*":F$(2,2)=CHR$(A):OPEN
   #1,6,0,F$
HQ 1120 GOSUB 50:POSITION 5,5:? "┌───  DIRE
   CTORY  DRIVE: #";CHR$(A+128);"▌──┐"
ER 1130 FOR X=6 TO 16.5 STEP 0.5:INPUT #1
   ,F$:F$=F$(3):IF F$(3,12)="FREE SECTO"
   THEN 1200
NP 1140 IF X=INT(X) THEN POSITION 5,X:? "
   |";F$(1,8);" ";F$(9,11);"▌";:FLAG=1:G
   OTO 1160
ZR 1150 POSITION 20,INT(X):? F$(1,8);" ";
   F$(9,11);"|":FLAG=0
LS 1160 NEXT X
PY 1170 ? "      └─── more-PRESS: return ───
   ┘":POKE 764,255
NX 1180 IF PEEK(764)<>255 THEN POKE 764,2
   55:GOTO 1120
SD 1190 GOTO 1180
NK 1200 IF FLAG THEN POSITION 20,INT(X):?
    "               |":FLAG=0
OG 1210 ? "      └─ finished-PRESS: return ─
   ┘";:CLOSE #1
RO 1220 IF PEEK(764)<>255 THEN POKE 764,2
   55:GOTO LINE
OZ 1230 GOTO 1220
```

```
NO 2000 LINO=LINO+10:IF LINO>32767 THEN F
   LAG=1:LINE=220:POKE 195,200:GOTO 6000
AD 2010 RETURN
XH 5000 MEMTOP=PEEK(106):CHBAS=MEMTOP-4:P
   OKE 106,CHBAS-1:POKE 208,CHBAS:GRAPHIC
   S 0:SETCOLOR 2,0,4:SETCOLOR 1,0,12
EL 5010 CHBAS=CHBAS*256:CURSOR=752:LET ON
   =0:OFF=1:OPEN #2,4,ON,"K:":POKE CURSOR
   ,OFF:?
PP 5020 DIM F$(15),FN$(15),A$(OFF),FIX(OF
   F,128),SP$(37),DLI$(12)
EF 5040 X=USR(ADR(MOVE$),57344,CHBAS,1024
   ):DLIST=PEEK(560)+PEEK(561)*256+4:POKE
    CURSOR,OFF
EX 5050 POKE DLIST+2,7:POKE DLIST+3,6:POK
   E DLIST+5,13:POKE DLIST+6,13:POKE DLIS
   T+20,141:POKE 54286,192
FL 5060 X=INT(ADR(DLI$)/256):J=ADR(DLI$)-
   X*256:POKE 512,J:POKE 513,X:POKE 756,N
   SET+2
CF 5070 ? "↖          analog presents♦":?
   "____FAST SETS____     SUBROUTINE WRITER"
   :? "        by♦ darryl w. howerton"
NF 5080 FOR J=0 TO 3:POSITION 4,19+J:K=J+
   1:IF J=2 THEN K=0
KO 5090 FOR X=0 TO 31:? "Ɛ";CHR$(X+(K-(J=
   3))*32);:NEXT X:NEXT J
BK 5100 POSITION 10,10:? "clearing arrays
   ...":FOR X=0 TO 1:FOR J=0 TO 128:FIX(X
   ,J)=0:NEXT J:NEXT X
AI 5110 RETURN
UU 6000 GOSUB 50:POKE CURSOR,OFF:TRAP 600
   5:CLOSE #1
AL 6005 POSITION 3,10:ERR=PEEK(195):POP
NV 6010 IF ERR=170 THEN ? "CAN'T FIND FIL
   E=> ";FN$;"!"
DF 6020 IF ERR=144 THEN ? "NO DISK IS IN
   THE DRIVE!":? "(OR DISK IS WRITE PROTE
   CTED)"
UH 6030 IF ERR=160 THEN ? "DRIVE NOT AVAI
   LABLE ON YOUR SYSTEM!"
MM 6040 IF ERR=167 THEN ? "FILE REQUESTED
    IS LOCKED!"
UO 6050 IF ERR=169 THEN ? "DIRECTORY IS F
   ULL! use another disk."
VX 6060 IF ERR=162 THEN ? "DISK IS FULL!
   use another disk."
LP 6070 IF ERR=18 THEN ? "ILLEGAL VALUE!!
   "
XO 6080 IF ERR=200 THEN ? "SUBROUTINE LIN
   E NUMBER OVERFLOW!!":? " use smaller l
   ine number at prompt"
WD 6090 POSITION 6,16:? " PRESS ANY KEY T
   O CONTINUE ":POKE 764,255
RL 6100 IF PEEK(764)<>255 THEN POKE 764,2
   55:GOTO LINE
PN 6110 GOTO 6100
BC 7000 TRAP 7030:ADRHI=INT(CHBAS/256):AD
   RLO=CHBAS-ADRHI*256
KY 7010 IOCB=848:POKE IOCB+2,7:POKE IOCB+
   4,ADRLO:POKE IOCB+5,ADRHI:POKE IOCB+8,
   0:POKE IOCB+9,4
SC 7020 I=USR(ADR("hhh⊠LVd"),16)
AO 7030 RETURN
OO 32000 MEMTOP=PEEK(106):CHBAS=MEMTOP-4:
   POKE 106,CHBAS:NSET=CHBAS:GRAPHICS 0:D
   IM MOVE$(43),CSET$(256)
GP 32060 X=USR(ADR(MOVE$),57344,CHBAS*256
   ,1024)
PD 32070 X=USR(ADR(MOVE$),ADR(CSET$),CHBA
   S*256+768,256)
NT 32075 FOR X=CHBAS*256+512 TO CHBAS*256
   +519:POKE X,0:NEXT X
EJ 32080 RETURN
```

●

Listing 2.
BASIC listing.

```
WT 1 REM *** PROGRAM TO CREATE LINES
IG 2 REM *** 5030 AND 32010 THRU 32050
AE 3 REM *** IN A FILE CALLED LINES.LST
JM 4 REM *** WHICH IS TO BE MERGED WITH
GF 5 REM *** THE MAIN FAST SETS PROGRAM
NL 6 REM
CS 10 GRAPHICS 0:? "PUT A FORMATTED DISK
   IN DRIVE #1 AND":? "PRESS  RETURN";:OP
   EN #1,4,0,"K:":GET #1,A
AF 20 ? "↖WRITING FILE..."
YS 30 CK=0:CKSUM=0:CLOSE #1:OPEN #1,8,0,"
   D:LINES.LST":RESTORE
GP 40 FOR X=1 TO 28:FOR I=1 TO 16:READ J:
   PUT #1,J:CK=CK+J:IF CK>1000 THEN CK=CK
   -1000
PD 50 NEXT I:READ CKSUM:IF CK<>CKSUM THEN
    ? "ERROR IN DATA...":? :? "CHECK WITH
    BASIC EDITOR II AND RE-RUN":GOTO 70
NZ 60 NEXT X
YA 70 CLOSE #1:END
JP 1000 DATA 53,48,51,48,32,83,80,36,61,3
   4,32,34,58,83,80,36,849
RJ 1010 DATA 40,51,55,41,61,34,32,34,58,8
   3,80,36,40,50,41,61,646
ZI 1020 DATA 83,80,36,58,68,76,73,36,61,3
   4,72,141,10,212,173,208,67
UA 1030 DATA 0,141,9,212,104,64,34,155,51
   ,50,48,49,48,32,77,79,220
HE 1040 DATA 86,69,36,61,34,104,104,133,2
   11,104,133,210,104,133,213,104,59
XE 1050 DATA 133,212,104,133,214,104,170,
   208,2,198,214,160,0,177,210,145,443
MY 1060 DATA 212,200,208,4,230,211,230,21
   3,202,208,242,198,214,16,238,96,365
ZT 1070 DATA 34,155,51,50,48,50,48,32,67,
   83,69,84,36,40,49,41,302
VZ 1080 DATA 61,34,0,0,12,24,0,24,48,0,0,
   12,62,102,110,252,43
LH 1090 DATA 204,0,0,31,51,126,108,204,24
   8,0,0,31,57,112,96,204,515
TA 1100 DATA 248,0,0,60,54,102,102,204,24
   8,0,0,63,48,124,96,192,56
BB 1110 DATA 252,0,0,63,48,124,96,192,192
   ,0,0,31,48,96,110,204,512
FC 1120 DATA 252,0,34,155,51,50,48,51,48,
   32,67,83,69,84,36,40,612
PS 1130 DATA 54,53,41,61,34,0,51,51,126,1
   02,204,204,0,0,63,12,668
FW 1140 DATA 24,24,48,252,0,0,3,3,6,6,204
   ,120,0,0,51,54,463
LE 1150 DATA 124,120,216,204,0,0,48,48,96
   ,96,192,252,0,0,51,55,965
PN 1160 DATA 127,107,198,198,0,0,51,59,12
   6,126,220,204,0,0,30,51,462
BM 1170 DATA 102,102,204,120,0,34,155,51,
   50,48,52,48,32,67,83,69,679
RI 1180 DATA 84,36,40,49,50,57,41,61,34,0
   ,62,51,102,124,192,192,854
KS 1190 DATA 0,0,30,51,102,102,216,108,0,
   0,62,51,102,124,216,204,222
VL 1200 DATA 0,0,31,48,124,14,60,240,0,0,
   63,12,24,24,48,48,958
TC 1210 DATA 0,0,51,51,102,102,204,252,0,
   0,51,51,102,102,124,48,198
PC 1220 DATA 0,0,51,51,107,127,238,198,0,
   34,155,51,50,48,53,48,409
VJ 1230 DATA 32,67,83,69,84,36,40,49,57,5
   1,41,61,34,0,51,51,215
CJ 1240 DATA 60,60,204,204,0,0,51,51,60,2
   4,48,96,0,0,63,6,142
TC 1250 DATA 28,48,96,252,0,0,12,12,24,24
   ,0,48,0,24,24,24,758
MR 1260 DATA 24,24,24,24,24,0,120,24,24,2
   4,24,120,0,0,8,28,250
```

```
ML 1270 DATA 54,99,0,0,0,0,0,0,0,0,0,255,
   0,34,155,0,847
DO 1280 REM * 448 BYTES
```

Listing 3.
Assembly listing.

```
; MOVE$ RAM MOVER SUBROUTINE
; BY D.W. HOWERTON
; CALL FROM BASIC WITH THE FORM:
; X=USR(ADR(MOVE$),FROM,TO,#BYTES)
; WHERE: FROM=MEM. LOCATION TO MOVE
;        TO=TARGET LOCATION
;        #BYTES=NO. OF BYTES TO MOVE
;
        *=   $0600      ;RELOCATABLE
FROM    =    $D2        ;ZERO PGE STORAGE
TO      =    $D4        ;FOR LOC USED IN
NUM     =    $D6        ;SUBROUTINE
        PLA             ;UNUSED-DISCARD
        PLA             ;HI BYTE OF FROM
        STA  FROM+1     ;STORE IT
        PLA             ;LOW BYTE OF FROM
        STA  FROM       ;STORE IT
        PLA             ;HIGH BYTE OF TO
        STA  TO+1       ;STORE IT
        PLA             ;LOW BYTE OF TO
        STA  TO         ;STORE IT
        PLA             ;HI BYTE #BYTES
        STA  NUM        ;STORE IT
        PLA             ;LO BYTE #BYTES
        TAX             ;X IS COUNTER
        BNE  START      ;GO AHEAD
        DEC  NUM        ;PREVENT MOVING
        ;               EXTRA PAGE OF MEM
START   LDY  #0         ;USE FOR INDEXING
MOVE    LDA  (FROM),Y   ;BYTE TO MOVE
        STA  (TO),Y     ;MOVE IT
        INY             ;INCREMENT INDEX
        BNE  NEXT       ;IF<>0 A PAGE
        ;               HASN'T BEEN MOVED
        INC  FROM+1     ;INCREMENT THE
        INC  TO+1       ;HIGH BYTES
NEXT    DEX             ;-LO BYTE COUNTER
        BNE  MOVE       ;MOVE ANOTHR BYTE
        DEC  NUM        ;-HI BYTE COUNTER
        BPL  MOVE       ;IF POS DO MORE
        RTS             ;RETURN TO BASIC
```

EXTENDED HOURS

MON. - FRI. 9A.M.-9P.M. CST   SAT. 11A.M.-5P.M.CST

ATARI 520ST / ATARI 1040ST

ATARI 520ST SYSTEM PACKAGE - CALL

INCLUDING RGB OR MONOCHROME MONITOR,
MOUSE, DISK DRIVE,BASIC,
TOS ON ROM, AND RF MODULATOR
FULL MANUFACTURER'S WARRANTY APPLIES

ATARI 1040ST SYSTEM PACKAGE - CALL

INCLUDING RGB OR MONOCHROME MONITOR,
MOUSE, DOUBLE-SIDED DISK DRIVE,BASIC
TOS ON ROM, AND BUILT-IN POWER SUPPLY,
FULL MANUFACTURER'S WARRANTY APPLIES

SUPRA 20 MEG HARD DRIVE ONLY $559

ATARI SF 314 DISK DRIVE

DOUBLE SIDED/ I MEGABYTE STORAGE

CALL

ST MODEM PACKAGE

*AVATEX 1200HC MODEM
*ST MODEM CABLE
*ST TALK TELECOMMUMICATIONS PACKAGES

$145

OKIMATE 20 COLOR PRINTER & ST PLUG N' PRINT

$199

EPSON

LX 86	219
FX 86E	399
EX 800	479
EX 1000	629
FX 286E	569
LQ 800.	519
LQ IOOO	729
LQ 2500.	1149

STAR MICRONICS

NX-10	CALL
NL-10	CALL
NP-10	CALL
NX-15	329
ND-15	449
NR-15	539
POWERTYPE	189
NB 24/15 (24 WIRE)	CALL

PANASONIC

PANASONIC 1080i	199
PANASONIC 1091i	259
PANASONIC 1092i	349
PANASONIC 3131	279
PANASONIC 3151	409
PANASONIC 1592	419
PANASONIC 1595	539

OKIDATA

OKIDATA 182	249
OKIDATA 192	369
OKIDATA 193+	559
OKIDATA 292E	449
OKIDATA 293E	599

CANON

CANON A-50	279
CANON A-55	389
CANON A-60	349

SEIKOSHA

SP-180	169
SP-1000A	199
SP-1200	209

BROTHER

BROTHER 1009	139
BROTHER 1509	379
BROTHER 1709	489
BROTHER HR 35	749
TWINWRITER 5	839

ST ADVENTURES

Hacker	17.95
Hacker II	25.95
Sundog	24.95
Black Cauldron	25.95
Apshai Trilogy	25.95
Universe II	44.95
The Pawn	29.95
Starglider	29.95
Dungeonmaster	24.95
Alternate Reality	24.95
Tass Times	25.95
Mercenary	24.95
Autoduel	32.95
Ogre	25.95
Defender of Crown	32.95
S.D.I.	32.95
Balance of Power	32.95
Ultima III or IV	38.95
Portal	32.95
Kings Quest 1,2 or 3	32.95
Space Quest	32.95
Tracker	29.95
Colonial Conquest	25.95
Roadwar 2000	25.95
Wizard's Crown	25.95
Rings of Zilfin	25.95
Sinbad	32.95
Golden Path	29.95
Phantasie 1,2 or 3	25.95

ST ACCESSORIES

Flip'n' File II- Micro	19.95
Dustcovers	Call
3.5 Drive Clean Kit	16.95
Mouse Pad	8.95
Mouse House	6.95
6 Way Surge Protector	19.95
Anti-Glare Screen	19.95
6 FT SF 354 Cable	19.95

ST TELECOMM

ST Talk Ver 2.0	18.95
BB/ST	31.95
Flash	27.95
I.S Talk	31.95

MICHTRON

BBS 2.0	48.95
Business Tool	31.95
Calendar	18.95
Cornerman	31.95
DOS Shell	24.95
Echo	24.95
M-Disk +	24.95
Major Motion	24.95
Mi-Term	31.95
Michtron Utilities	37.95
Cards	24.95
The Animator	24.95
Time Bandits	24.95
Mighty Mail	31.95
Personal Money Mgr	31.95
Pinball Factory	24.95
Financial Future	24.95
Eight Ball	18.95
Hard Disk Backup	24.95
Match-point	24.95
Karate Kid II	24.95
GFA Basic	48.95
Trimbase	62.95
Shuttle 2	24.95
M-Cache	24.95
GFA Draft	62.95
Journey to the Lair	32.95
GFA Compiler	48.95
Realtizer.	149.95

ST PRINT UTILITIES

Rubber Stamp	24.95
Printmaster Plus	24.95
Megafont ST	24.95
Art Gallery 1 or 2	18.95
Typesetter Elite	31.95
PM Interface	18.95
Certificate Maker	31.95
Library 1/Certificate Mkr	21.95

ST WORD PROCESSORS

Regent Word II	48.95
Wordwriter ST	48.95
Thunder	24.95

ST EDUCATIONAL

Decimal Dungeon	24.95
Fraction Action	24.95
Kinderama	24.95
Read & Rhyme	24.95
Math Wizard	24.95
Animal Kingdom	24.95
Speller Bee	31.95
Kid Talk	31.95
Math Talk	31.95
First Shapes	31.95
Winnie the Pooh	15.95
ADVANCED ORDERS	
Read-A-Rama	31.95
Aesop Fables	31.95
Magical Myths	31.95
All About America	31.95
Adv. of Sinbad	31.95
Arabian Nights	31.95
Wiztype	25.95
Buzzword	29.95
1st Letters and Words	31.95
Math Talk Fractions	31.95

ST UTILITIES

Music Studio	32.95
Time Link	31.95
Middiplay	32.95
Micro Cookbook	32.95
CZ Droid	69.95
EZ Track	39.95
Write 90	18.95
Fast	31.95
Labelmaster Elite	27.95
Video Wizard	39.95
K-Switch	27.95
Desk Cart	72.95

CASIO KEYBOARDS

CZ 101	Call
CZ 230S	Call

DESKTOP PUBLISHING

Drawrite	99.95
Publishing Partner	99.95

HIPPOPOTAMUS

Call for items and prices

INFOCOM ST

Forever Voyaging	29.95
Bally Hoo	25.95
Bureaucracy	25.95
Cutthroats	25.95
Deadline	31.95
Enchanter	19.95
Hitchiker	19.95
Infidel	28.95
Leather Goddess	25.95
Moonmist	25.95
Planetfall	25.95
Seastalker	25.95
Sorcerer	28.95
Spellbreaker	31.95
Starcross	31.95
Suspect	28.95
Suspended	31.95
Trinity	25.95
Wishbringer	11.95
Witness	25.95
Zork I	25.95
Zork II or III	28.95
Hollywood Hijinxs	25.95

ST LANGUAGES

Personal Pascal	49.95
Mark Williams C.	114.95
Metacomco Pascal	69.95
Macroassembler	49.95
Lattice C	99.95
Cambridge Lisp	139.95
Modula II	49.95
Metacommco Make	49.95
Modula II Developer'	Call
Fast Basic	89.95
Fortran 77.	124.95
LDW Basic	44.95

ST DATABASES

DB Man	96.95
Regent Base	57.95
Data Manager ST	48.95
Zoomracks II	96.95
Superbase Gem	94.95

ST ARCADE GAMES

Winter Games	24.95
Rogue	24.95
Super Huey	25.95
Mean 18	25.95
Famous Course Disk	14.95
Leader Board	25.95
Brattacus	32.95
Silent Service	25.95
Flight Simulator II	33.95
Champ. Wrestling	24.95
World Games	24.95
Video Vegas	24.95
Strip 'Poker	25.95
Little Computer Peo	11.95
Gato	31.95
Joust	20.95
Star Raiders	20.95
Chessmaster 2000	29.95
Tenth Frame	25.95
Shanghai	25.95
3-D Helicopter	33.95
Micro League Baseball	39.95
WWF Micro Wrestling	39.95
Skyfox	29.95
Super Cycle	24.95
ST Pool	22.95
Indoor Sports	32.95
F-15 Strike Eagle	27.95
High Roller Simulator	32.95
Two on Two Basketball	25.95
Deep Space	29.95
Arena	22.95
ST Karate	22.95
The Wanderer	25.95
Champ Baseball '86	25.95
GFL Football	25.95
Harrier Srike	31.95
Space˜ Stat/Protector	25.95
Sub Battle Simulator	24.95
Arctic Fox	26.95
RPV	24.95
Crystal Castles	20.95
Gridiron Football	42.95
ST Wars	24.95
Hardball	24.95
Psion Chess	38.95

ST BUSINESS

VIP Professional	Call
Swiftcalc ST	48.95
Isgur Portfolio	124.95
Financial Cookbook	32.95
DAC Payroll	32.95
DAC Easy Accounting	44.95
Dollars and Sense	64.95
Home Accountant	31.95
Sylvia Porter	48.95
A-calc Prime	39.95
Logistix Jr.	64.95
Logistix Sr.	89.95

ABACUS

Textpro	32.95
Datatrieve	32.95
Forth MT	32.95
Paintpro	32.95
Assempro	39.95
Powerplan	49.95
Abacus Books	Call

ST GRAPHICS

Degas Elite	Call
Easy Draw	48.95
Cad 3-D	36.95
Graphic Artist	124.95
Paintworks	25.95
1st Cadd	31.95
Aegis Animator	48.95
Art I /Aegis Animator	21.95
Neochrome	34.95
Stereo Cad 3-D	67.95

DISKETTES

Sony 3.5 SS/DD (10 PK)	13.95
Sony 3.5 DS/DD (10 PK)	18.95

Note: Buy Diskettes at these low prices when added to any other order.

No surcharge for Master Card

MasterCard

EST. 1982
ComputAbility™

VISA

No surcharge for Visa

To Order Call Toll Free
800-558-0003

P.O. Box 17882, Milwaukee, WI 53217
ORDER LINES OPEN
Mon.-Fri. 11 a.m. - 7 p.m. CST © Sat. 12 p.m. - 5 p.m. CST

TELEX NUMBER 9102406440
(ANSERBACK = COMPUT MILW UQ)

For Technical Info, Order
Inquiries, or for Wisc. Orders
414-357-8181

ORDERING INFORMATION: Please **specify system.** For fast delivery send cashier's check or money order. Personal and company checks allow 14 business days to clear. School P.O.'s welcome. **C.O.D. charges are $3.00.** In Continental U.S.A. include $3.00 for software orders. 4% shipping for hardware, minimum $4.00. MasterCard and Visa orders please include card #, expiration date and signature. WI residents please include 5% sales tax. HI, AK, FPO, APO, Puerto Rico and Canadian orders, please add 5% shipping, **minimum $5.00.** All other **foreign orders add 15% shipping, minimum $10.00.** All orders shipped outside the Continental U.S.A. are shipped first class insured U.S. mail. If foreign shipping charges exceed the minimum amount, you will be charged the additional amount to get your package to you quickly and safely. All goods are new and include factory warranty. Due to our low prices all sales are final. **All defective returns must have a return authorization number.** Please call (414) 357-8181 to obtain an R.A. # or your return will not be accepted. Prices and availability subject to change without notice.

CIRCLE #104 ON READER SERVICE CARD

EXTENDED HOURS

MON. - FRI. 9A.M.-9P.M. CST SAT. 11A.M.-5P.M. CST

ATARI 800/ XL/ XE COMPUTERS

ATARI 130XE COMPUTER

WITH 2 FREE PROGRAMS

(OUR CHOICE)

$134

ATARI 1050 DISK DRIVE

WITH DOS 2.5 INCLUDED

$129

ATARI 130 XE PACKAGE

INCLUDES

- *130 XE COMPUTER
- *1050 DISK DRIVE
- *1027 PRINTER
- *ATARIWRITER +
- *MILLIPEDE
- *HOME FILING MANAGER
- *PAINT
- *DEFENDER

$399

NX-10 PRINTER & SUPRA 1150 INTERFACE

$259

AVATEX 1200 HC MODEM & P.R. CONNECTION

WITH ATARI MODEM CABLE

$179

MISCELLANEOUS HARDWARE

P.R CONNECTION.	62.95	SUPRA 1150 INTERFACE	59.95
US DOUBLER	49.95	850 INTERFACE	CALL
ANIMATION STATION	49.95	GRAPHIC AT INTERFACE	39.95
ATARI XM 301 MODEM	CALL	AVATEX 1200 HC MODEM	125.00

PANASONIC 1080i PRINTER & SUPRA 1150 INTERFACE

$249

PANASONIC 1091i PRINTER & SUPRA 1150 INTERFACE

$309

PANASONIC 1092i PRINTER & SUPRA 1150 INTERFACE

$379

ACCESSORIES

6 Way Surge Protector	19.95
Monitor Cable	9.95
Printer Stand	14.95
Swivel/Tilt Monitor Stand	19.95
Anti-Glare Screen	19.95
Teakwood Disk Holders	19.95
Disk Notcher	5.95
Blank Disks	Call
Printer Ribbons	Call
Dust Covers	Call
Modem Cable	14.95
Printer Cable	14.95
1000 Sheet Printer Paper	19.95

XLENT SOFTWARE

Megafont II.	16.95
Xlent Word Processor	18.95
Page Designer	18.95
Megafiler	18.95
Picture Disk	14.95
PS Interface	18.95
Rubber Stamp	18.95
Typesetter	21.95

BRODERBUND

Karateka	19.95
Print Shop	28.95
Bank Street Writer	32.95
Graphic Lib 1,2,or 3	16.95
Print Shop Compan	22.95

ELECTRONIC ARTS

Age of Adventure	10.95
Archon	10.95
Archon II.	10.95
Financial Cookbook	10.95
Mail Order Monsters.	10.95
Movie Maker	10.95
Music Construction	10.95
One on One	10.95
Pinball Construction.	10.95
Racing Destruction	10.95
7 Cities of Gold	10.95
Super Boulder Dash.	10.95
Touchdown Football	10.95
Chessmaster 2000	25.95
Chickaumauga	23.95
Lords of Conquest	22.95
Ogre	25.95
Rommel/Toburk	26.95
Starfleet 1	33.95
Ultima 1	27.95
Ultima III	31.95
Ultima IV	38.95

DATASOFT

Alternate Reality/City	19.95
Alt. Reality/Dungeon	24.95
Gunslinger	18.95
Video Title Shop	21.95
221 Baker Street	18.95
The Mercenary	18.95
Theatre Europe	21.95
Data Disk/Mercenary	11.95

SSI

Carrier Force	37.95
Wargame Constr. Set	19.95
Broadsides	24.95
Computer Ambush	37.95
Fighter Command	39.95
Field of Fire	24.95
Imperium Galacticum	24.95
Warship	39.95
Kampgruppe	37.95
Battle of Antietam	32.95
USAAF	37.95
Phantasie	27.95
Wizard's Crown	27.95
Gettyburg	39.95
Panzer Grenider	24.95
Rebel Charge	32.95
Battlecruiser	39.95

ACTIVISION

Cross Country Race	11.95
Hacker	11.95
Mindshadow	16.95
Music Studio	22.95
Ghostbusters	11.95

JOYSTICKS

(NOTE: Prices good when with any other purchase)

Wico Super 3-Way	19.95
Unite Joystick	5.95
Starfighter	8.95
Tac 3	10.95
Slik Stick	7.95
Epyx 500XJ	12.95

MICROPROSE

Silent Service	22.95
F-15 Strike Eagle	22.95
Decision in Desert	25.95
Kennedy Approach	16.95
Crusade in Europe	25.95
Conflict in Vietnam	25.95
Top Gunner	16.95

OPTIMIZED SYSTEMS

Basic Xe.	44.95
Action	44.95
Mac 65 XL.	44.95
Basic XL	34.95
All Tool Kits	19.95

WE CARRY 100'S OF PROGRAMS FOR THE 8 BIT ATARI COMPUTERS. IF YOU DON'T SEE IT,

CALL

MISCELLANEOUS

Flight Simulator 2.	33.95	Print Driver Const. Set	16.95
Universe.	57.95	TAC	25.95
Strip Poker.	21.95	Jupiter Mission 1999	32.95
Micro League Baseball	25.95	Blue Max 2001	19.95
Harcout/Brace SAT	59.95	Championship Loderunner.	19.95
General Mgr. MLB.	25.95	Home Accountant	30.95
Fight Night	18.95	Apshai Trilogy	14.95
Hardball.	18.95	Summer Games	14.95
Triple Pak	14.95	World Karate Championship.	19.95
Leader Board	24.95	R Tlime 8 Cart	54.95
Tourn Disk/Leader Bd.	14.95	Halley's Project	10.95
Executive Disk/Leader Bd	14.95	Mastertype	25.95
Tenth Frame	27.95	Word Attack	32.95
Infiltrator.	19.95	Math Blaster	32.95
Bop'n' Wrestle	19.95	Spell It.	32.95
SuperHuey	16.95	DLM Teaching	CALL
Home Planetarium	27.95	Spinnaker	CALL
Atariwriter Plus	29.95	Hardball	18.95
MLB Boxcore/Stat	16.95	Ace Of Aces.	18.95
SPY vs SPY 1&2	10.95	Ultima 2.	28.95
Video Vegas	20.95	Mrca Mach II.	20.95
Blazing Paddles	22.95	The Pawn	25.95
		Trailblazer	19.95

INFOCOM

Hitchiker	17.95
Wishbringer	11.95
Ballyhoo	25.95
Moonmist	22.95
Leather Goddess	22.95
Hollywood Hijinx	22.95
Zork Trilogy	39.95

SYNAPSE

Syncalc	32.95
Synfile	32.95
Loderunner's Rescue.	19.95
Syncalc Templates	16.95
Essex	25.95
Mindwheel	25.95
Brimstone	25.95

MasterCard

No surcharge for Master Card

To Order Call Toll Free

800-558-0003

EST. 1982

ComputAbility™

P.O. BOX 17882, Milwaukee, WI 53217
ORDER LINES OPEN
Mon,-Fri. 9a.m.-9p.m. CST Sat. 11a.m.-5p.m. CST

TELEX NUMBER 9102406440

(ANSERBACK • COMPUT MILW UQ)

VISA

No surcharge for Visa

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

414-357-8181

CIRCLE #104 ON READER SERVICE CARD

Reader comment

continued from page 8

Plaudits for *MICROMOD-Turbobase.*

I am currently a subscriber to **ANALOG Computing**, and have been since June of 1985. I enjoy the publication, as evidenced by my continued subscription.

My purpose for this letter is threefold: (1) to give plaudits to **MICROMOD Turbobase** software; (2) to attest to the "business" power of the Atari 130XE; and (3) to encourage "support" of 8-bits.

Atari suffers the reputation of a "toy," something for the kids to play with. Unfortunately, in my area, Atari is doing very little to change this image—much to the chagrin of the Huntsville Atari Users Group (H.A.U.G.) Though there are several excellent business applications on the market, Atari doesn't appear to capitalize on their existence.

I have many of them in my own library, including **SynCalc**, **SynTrend**, **SynStock**, **VisiCalc**, **Networth**, **Peachtrees** (G/L, AR, AP), **Family Finances** and, last but not least, **MICROMOD Turbobase**.

I believe I can truthfully say I am proficient in all of them. I have been looking for the one "system" that would be capable of "doing it all"—a system that could be used by a small business. The rational is that most "Mom and Pop" businesses cannot afford the $10,000-plus price tag for one of the "big guys"—or even for a clone of one. Well, I believe I now have just such a system in **Turbobase**.

My first experience with **MICROMOD** goes back to September 1986, with my purchase of **MICROMOD 3.0**. I was immediately impressed with the program, and equally disappointed with the manual and bugs. Although the manual was well written, I felt it was *not* written for the general public—i.e., it assumed too much knowledge on the part of the user.

I contacted Mr. Steve Bolduc and expressed my opinions to him. He was most gracious, receptive and supportive. My comments must have not been unique, for, shortly thereafter, Mr. Bolduc removed **MOD 3.0** from the market.

In February of 1987, I received a copy of the **MICROMOD-Turbobase Quick Course and Cookbook**, and a demo disk of **Turbobase**. Mr. Bolduc listened very well indeed. **QCC** does all he claims, and more. It's not a manual *per se*, but a well written "walk-through" a new user could utilize to set up his books without even looking at the manual.

On April 9th, I got my copy of **MICROMOD-Turbobase** with the new manual. *Outstanding!* Mr. Bolduc *et al.* have outdone the industry, even the "big guys."

The program does all he claims, and is more than adequate for a small to medium business application, depending, of course, on the type of business. It's certainly suitable for a small store or contractor, with a limited inventory. (I am now using the system to keep the books for a 100-unit condominium.)

The manual is something else again—*terrific!* It is extremely well written, if at times a little verbose. Anyone can understand the instructions.

I feel Mr. Bolduc has gone beyond the "typical" manual, in that he not only gives examples of applications, but fully explains the logic behind the method, to allow the user to "design" specific applications. This is something not readily available in "off the shelf" programs.

In the support area, **MICROMOD** has no equal. I won't mention those who've been nonsupportive. However, I will say, on the few occasions I contacted Mr. Bolduc, his response was immediate and *positive* (no excuses). **MICROMOD** does live up to its claim of full product support.

J.J. Moniz, Treasurer
Huntsville Atari Users Group (AL)

*For information on **MICROMOD-Turbobase**, you can contact MicroMiser Software, 1635-A Holden Ave., Orlando, FL 32809 — (305) 857-6014. And watch for a review in these pages. —Ed.*

Keyboard contacts.

I purchased a 130XE last October. When reading the issue 52 (March 1987) **Database Delphi** regarding its keyboard, I became very concerned. The term *graphite contacts* stirred up unpleasant memories of the Atari 5200 joysticks.

The fire buttons, keypad and START/PAUSE/RESET buttons use small, round, black pieces of conductive material, which a technician told me were "graphite contacts." Used regularly, they function properly for several months, then gradually lose their conductivity. Cleaning sometimes—but not always—helps. I repaired four of the sticks for a neighbor last year (her grandchildren abuse the machine), by gluing pieces of aluminum-sensing foil over each and every contact. It's a slow, tedious fix (especially when your eyesight balks at anything close up), but seems to work.

The **Database Delphi** comments ("the bottom of the key is made of a U-shaped piece of conductive rubber that bridges two pads on the circuit board to make a key closure") brought visions of a 130XE keyboard full of "5200-type" contacts.

So, throwing caution to the wind, I disassembled my 130XE (and, later, a 600XL) keyboard. What I discovered was a great relief. Either I misunderstood the comments in **ANALOG**, or Atari has recognized and corrected the problem.

Each key (both console and keyboard) contained a flexible rubber insert with no conductive coating. The method of key closure, in both the 600XL and 130XE, is essentially similar. There is a folded-over plastic "circuit board" with foil patterns on each side. An insulator sheet of plastic (with holes to allow conductivity—one for each key) lies in between. Depressing a key forces foil patterns on the top and bottom "boards" to make contact, resulting in a key closure.

The only problem I can foresee is tarnishing (or possible damage?) of the circuit board foils. I did notice contacts on the 130XE's connector cable were tarnished, upon removal from the mother board. To forestall any future problems, I cleaned them with a Freon TF/alcohol solution.

I don't know if all 130XE keyboards are identical to this one, but feel relieved. This incident also helped me realize that keyboard disassembly, should it become necessary, isn't as tough as I'd imagined—only potentially hazardous to that flexible printed circuit board.

Bill Hicks
Newport News, VA

*Matthew J.W. Ratcliff, our **Database Delphi** oracle, tells us that there are several variations on the Atari's keyboards. His local dealer has seen three types, which vary depending upon their origin of manufacture (Taiwan, Korea, etc.)*

One more thing—remember Matt's Rule of Computing: if it ain't broke, don't fix it. You could get into very hot water—and invalidate your warranty. —Ed.

Updates.

*Our apologies for any confusion the recent review (issue 53) of **The Learning Phone** may have caused. It seems that Plato discontinued their Homelink service shortly before we went to press, so there's really no point in running out to get a Plato access cartridge. If you own one, you'll be pleased to know that Atari will swap it for another of your choice.*

*Apologies also to Matthew Ratcliff for omitting his name in issue 54's table of contents, for his review of **Super 3D Plotter**. (Greg: if you're trying to remember writing the review, forget it.) —Ed.*

TRADE SECRETS

Part 1—A survival guide for the novice programmer.

by Clayton Walnum

If only I had known. . .

When I think of all the hours I wasted back in the "good old days," trying to recover lost work or tracking down program bugs that were, because of my inexperience at the time, harder to find than a squadron of black cats at midnight, I feel as if a huge portion of my life has been squandered. Of course, that time wasn't *really* wasted: every mistake was a lesson learned, every stumble yielded a new piece of wisdom to add to my ever-enlarging repertoire of programming skills. But I still can't get over the feeling that if I had only known a few simple rules, if only someone had told me. . .

If you're new to programming—or even if you've been at it for a year or so—you're undoubtedly becoming familiar with a certain mental dizziness caused by walking in intellectual circles. You're spending a lot of time backtracking, redoing things that could have been done right the first time around, if—at the risk of repeating myself—only you had known.

Well, today's your lucky day. Mark it on your calendar! I'm going to let you in on some techniques that'll make your programming efforts less frustrating, maybe even enjoyable. Don't scoff. Your mental health is valuable. Following good advice can keep you from a visit to that infamous rest home for retired programmers, the Institute for the Incredibly Nervous.

Endless saves.

First of all, when you're programming or doing *anything* on a computer, *save your work frequently*. And don't just sit there humoring me—don't nod your head and grin and give me that I-must-have-heard-that-one-a-million-times look. I didn't put that in italics because I think slanted lettering is hip. Ignore this piece of advice, and you might as well sign the Institute's guest list right now (and bring a lunch and a change of socks, because you're going to be there for a while).

When you're working on a program, you should save your work at least every half an hour. If you don't, you *will* get burned. Even experienced programmers forget, and that's when the demon bites.

You see, there's this little guy with horns, a pointed tail and a severe skin problem, who watches over the shoulders of all laboring programmers. At the exact moment this visitor from the nether realms sees that you've made some crucial, unsaved changes, he lures some other creature (usually a cat or a small child) over to the plug, then stands back to watch the fun.

Don't believe me? Okay, don't save your work. You'll see.

The demon has other methods, too. He likes to point POKE commands to your computer's operating system or change the address of a USR call, so your computer will do its famous Yale trick. (Lock up. Get it?) I know all this is true, because I've found those very mistakes in my programs, and *I* never make mistakes like that.

Take it from me. If you're not willing to spend a little time saving your work, don't even bother starting on your next program. You'll never finish it. The demon bites.

Endless versions (well, a few, anyway).

Not only should you save your work frequently, but you should retain older versions of your program, rather than writing over them with new ones. One good way to do this is use a version number as the file's extension. For instance, MYPROG.V01 would be the first version of MYPROG. After making some changes, the program would then be saved as MYPROG.V02, and so on.

Don't trust me on this one either? Okay, but you'd be surprised how a few changes can make the run of an almost finished program look like an excursion into the Twilight

Zone. Many times, finding these problems can be so tough it's easier to give it up and try again. That's when you're going to wish you had the previous version lying around.

On many larger computer systems, handling version numbers is built in. Also, you may have used word processors on your Atari that will automatically preserve an older file by changing its extension to .BAK (for backup) before saving the current version. You can add an "auto-backup" capability to DOS 2.0 by using Gary Domrow's excellent **DOS Mods** from issue 39. This modification will preserve as many old versions of a program as you like, and will automatically load the most current version when you're ready to get back to work. It saves the programmer from having to remember the most recent version number, making the entire process painless. I highly recommend it.

Tape vs. disk.

Some of you aren't going to take kindly to this, but it has to be said. If you want to be a programmer, get rid of that tape machine and get a disk drive. Cassettes are about as reliable as Charles Manson. If I had a nickel for every program I lost to the hazards of those accursed ribbons of iron filings, I'd be in Hawaii right now, sipping Piña Coladas on the beach. You just can't trust tapes! Even if you save the darn program out three times, you're not safe. Best I can figure is that the demon uses cassette tape for dental floss.

There's also the matter of speed. If you're going to save your work often (you are, right?), and you're stuck with a tape system, you can spend a lot of time staring at the screen, listening to an endless stream of data stampeding from your computer to an instrument that would serve you better as a bookend or a doorstop. That high whining sound makes people grit their teeth—but it'll *kill* dogs. If your program is large, you can expect to listen to the howling of the poor, dying beasts for up to twenty minutes.

Line numbering.

Most BASIC programming guides show programs numbered in intervals of 10. That's great for a final version, but when you're developing a program, you're going to need more space between lines than that—unless you've got your program planned very thoroughly.

When I start work on a new program, I usually number the lines in increments of 100. That is, the first line of my program will be *100*, the second *200*, and so on. If I later discover I have to insert a large block of code somewhere, I have plenty of room. There's nothing worse than manually renumbering a program section because you need to accommodate an unexpected piece of code. Not only is it annoying, but it's dangerous. What if you have some references to those lines elsewhere in the program? You better get them all!

Numbering by *100*s solves a lot of problems, but it also creates one. Suppose the program becomes so large that the line numbers exceed the maximum of *32767*? The only way out in this case is to renumber the entire program. But don't even *think* about doing it by hand! It's ridiculously meticulous and time consuming, and I guarantee if you try it, your program (unless it's very small) will never run again. There's no way you'll ever get all those GOTOs and GOSUBs changed properly.

Nope, renumbering is a perfect job for your computer to perform on its own. There are many commercial and public domain programs that can handle this task (issue 27 contains a program called **Instant Renumberer** that'll do the job).

No BASIC programmer should be without a renumbering utility. No matter how large an increment you originally used for your numbering, you'll find the program closing in on you as you add more and more code between existing lines. That's when you whip out your renumber utility and restore the numbering to your original increment.

Once the program is complete, you can renumber it a final time, using a line increment of ten to give it that professional, finished look.

But, if you plan to use a renumberer, you *must* be aware of certain restrictions. Basically, you should avoid any indirect line references—that is, any line that's referenced using a variable. For instance:

```
1000 GOTO LINE*10
```

will leave any renumber utility scratching its head. As a general rule, you should avoid indirect line references (except one particular type I'll be mentioning later), whether you plan to renumber or not. They cause more problems than they're worth. If you can't avoid it, and you do renumber the code, you'll have to recalculate all the references by hand (shudder).

A REMarkable idea.

As your programs emerge from their humble, one- or two-dozen-line beginnings to many-hundred-line masterpieces, finding specific sections of code will become difficult. For this reason, you should use REM statements prodigiously. Yes, I know they take up a lot of memory, but, in today's 48K to 128K machines, that's usually not a problem. And if they *do* become a problem, you can always delete some of them later.

When you're trying to read through a 10-page printout of a program, you can save huge amounts of time by having the major sections of your programs easily identifiable. This is one of the reasons why the good people who designed BASIC gave us REM statements. And don't just allocate one REM line for your "label"; use several. A single REM is like a popsicle stick in a forest—you'll never see it. Which of the following do you like?

```
1000 REM SAVE GAME
1010 OPEN #1,8,0,"GAME.DAT"
1020 ? #1;SCORE:? #1;LIVES
1030 CLOSE #1
```

or

```
1000 REM **********************
1010 REM *     SAVE GAME      *
1020 REM **********************
1030 OPEN #1,8,0,"D:GAME.DAT"
1040 ? #1;SCORE:? #1;LIVES
1050 CLOSE #1
```

Remember, you can always tighten up your program later by removing the REMs, but while you're working, it's a great boon to be able to find code segments fast. Unless you're running out of memory, I'd even leave them in the finished program. You never know when you or somebody else might want to do an enhancement.

The house that Jack built.

Of course, marking off sections of code with REM statements only works well if you impose some sort of structure on your program. When Jack put up his house, he decided how many rooms he wanted before he bought the lumber and nails. He also decided what kind of rooms there'd be: bedroom, kitchen, living room.

When you first begin to think about a programming project, the thought of all those hundreds of program lines is intimidating, to say the least. Worse than intimidating, it's confusing. It's one of those cases where you can't see the forest because of the trees (I must have forests on the brain today).

Think of each program as a series of general steps, and worry about the details later. Outline on paper the steps you must take to complete your program.

All programs contain three general steps:

I. INITIALIZATION
II. MAIN PROGRAM
III. END

Each of these steps can, in turn, be broken up further, and may end up looking something like this:

I. INITIALIZATION
    A. DRAW TITLE SCREEN
    B. DIM AND INITIALIZE VARIABLES
    C. REDEFINE CHARACTER SET
    D. DRAW MAIN SCREEN
II. MAIN PROGRAM
    A. GET PLAYER'S INPUT
    B. SEPARATE VERB AND NOUN
    C. PERFORM COMMAND
III. END
    A. DRAW ENDING SCREEN
    B. SAVE GAME VARIABLES IF NECESSARY
    C. CLEAR SCREEN

The above is a general outline for a text adventure game.

Each of the new steps shown can be reduced to even greater detail. But the point is: you don't have to worry about the whole project at once. And you don't necessarily have to code the program in the order shown in the outline. For example, you can go ahead and do the dimensioning and initialization of the variables before you design the title screen.

Nothing is cast in stone. You won't be through with each step until you've completed the program. For instance, as you get into the game outlined above, you'll inevitably add more variables, which means, of course, they must be initialized. *That* means a quick step back to the initialization step. Each section of the program will evolve, but at least you'll be focusing on one thing at a time. If you've used REM statements as I've suggested, it'll only take a second to find a program section that requires modification.

The point here is to pick one element of the program and work on it until it's as complete as possible, then move on to something else. Changes in one section will inevitably lead to changes in another. That's okay. Make the required changes, then turn your attention back to the task at hand.

Sensible monikers.

When you're writing your program, do everything possible to make the process easy on yourself. A great way to make your program readable and easy to follow is to use variable names that give some clue to the function they're serving. For instance, which of the following is clearest to you?

```
1000 A=P*(1+R)^N
```

or

```
1000 AMOUNT=PRINCIPLE*(1+RATE)^PERIODS
```

This is the formula for calculating your bank balance after a given period of time, including compounded interest. The first version is okay, since at least we used variable names that hinted at what they were. But the second version is instantly readable, and, since the variable names are longer, there's less chance of accidentally using the same one twice. There are only twenty-six possibilities if you want to stick to single-letter names (I arrived at that figure only after much thought and deliberation). You'll need more variables than that for a large program.

Another type of moniker.

Giving major program sections "labels" is another way you can make your programs more readable—and thus easier to debug. The statement *GOSUB SOUND1* means more to you as the programmer than does *GOSUB 1000.*

Using this technique can save you endless hours of tracing through program listings. When you're trying to track down a problem and you encounter GOSUB 1000, you're going to have to find Line 1000 and figure out what that section of code is doing—unless you have a great memory. By using GOSUB SOUND1, you know immediately what that subroutine does, and, assuming the problem doesn't lie in that section of code, you can move along to the next statement.

Of course, just like every programming technique, this leads to new problems. For one, if the problem you're trying to track down *is* in the section of code labeled *SOUND1*, you're going to have to find out what line number the label represents. This isn't too tough if you initialize all the labels on the same program line; you can just look it up. What I usually do when employing this technique is keep each label and its associated value on a piece of paper for quick reference.

By far the worst problem, though, is that you're using indirect line references, and you know what that means, right? Your renumber utility is going to throw a tantrum. There's a way around this problem: make sure each section of code referenced by a label is easily identifiable by REM statements, as I described above. Then, when you renumber the program, ignore the renumberer's complaints (we may hope that your renumber utility will continue renumbering even when there are indirect references in the code; most of them grumble a bit, but then go on their way).

Once the program's been renumbered, list it to the screen and make note of the new line numbers for each labeled subroutine. Then bring up the line containing the labels' initialization and change them to their new values. Presto! All indirect references within the program have been resolved.

Need an example? Look at this program segment:

```
1000 REM ***********************
1100 REM *     SAVE GAME       *
1200 REM ***********************
```

```
1300 OPEN #1,8,0,"D:GAME.DAT"
.
.
.
2000 REM ***********************
2100 REM *      LOAD GAME      *
2200 REM ***********************
2300 OPEN #2,4,0,"D:GAME.DAT"
.
.
.
5000 SAVEGAME=1300:LOADGAME=2300
```

Now, let's say we renumber the program, and it comes out looking like this:

```
110 REM ***********************
120 REM *      SAVE GAME      *
130 REM ***********************
140 OPEN #1,8,0,"D:GAME.DAT"
.
.
.
210 REM ***********************
220 REM *      LOAD GAME      *
230 REM ***********************
240 OPEN #2,4,0,"D:GAME.DAT"
.
.
.
510 SAVEGAME=1300:LOADGAME=2300
```

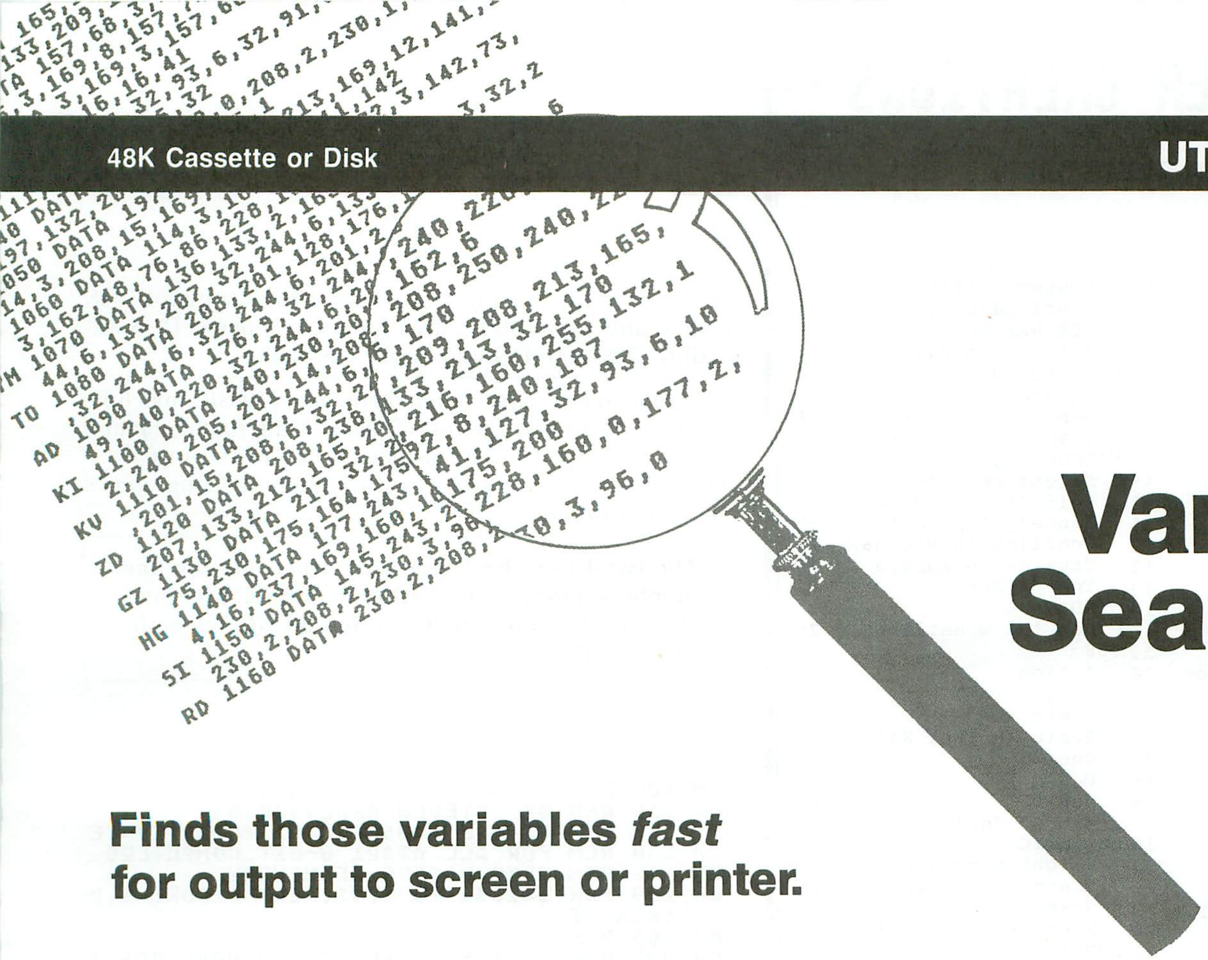
Just find the line that contains the initialization of the labels (in this case, Line 510) and change the values to the new line numbers:

```
510 SAVEGAME=140:LOADGAME=240
```

Now all references in the program will be correct.

Till next time.

This concludes our first excursion into survival techniques for the BASIC programmer. Next month, we'll look at many more ways to keep those guys in the white coats from banging on your door. With any luck, we'll put the Institute right out of business. See you then. ▣



Variable Searcher

Finds those variables *fast* for output to screen or printer.

by Steven Anderson

All BASIC programs use variables—or at least 99 percent of them do—and checking through the program to find them, for any editing reason, can be tedious work.

One of the secret powers of the Atari is its ability to use long variable names. Some programmers like to change variable names used in a program to longer, easier-to-understand names; other programmers want to make long names shorter. **Variable Searcher** can find and cross-reference all variables in seconds, and output to either the screen or a printer. It writes both the variable names and the line numbers in which each variable appears; even variables entered without line numbers are shown.

Variable Searcher is an all machine language program entered by a BASIC loader program. It uses page 6 of memory (0600-06FF hexadecimal, or 1536-1791 decimal). Here, it is free from BASIC and from the RESET key.

To execute the **Searcher**, type in direct mode: *X=USR (1536)*. To send output to a printer, enter *POKE 203, 80* before the USR. To redirect to the screen, enter *POKE 203,69*.

The **Searcher** works by first finding the Variable Name Table. The starting address of this table is in locations 130 and 131. To find the value with BASIC, type: *PRINT PEEK (130)+256*PEEK(131)*.

The ending address is in locations 132 and 133. Once located, the **Searcher** will find and write each variable name, store its token number, then search through BASIC memory for each token number. If it finds any matching token numbers, it then checks for a REM, DATA or PRINT before the token, because the data following REM, DATA or PRINT may have the same value as the token—but not as a variable. If there is no REM, DATA or PRINT, **Searcher** writes the line number where the variable was found. It continues this process until it reaches the end of the Variable Name Table, then returns to BASIC.

It does not check for repeats while searching, so a line with a variable used more than once may appear each time the variable is found.

All variables entered are stored in the Table, even those entered in direct mode—they will be output with no line numbers following them.

Type in the BASIC loader program. This creates the machine language for **Searcher**. Be extra careful on the data lines. Save a copy, then run the loader. Once the *READY* prompt appears, the machine language file is in place and ready to use.

Test it on the loader to be sure it was entered correctly. To use **Searcher** on a different program, type *NEW*, then type or load the new program and use X=USR (1536) to institute **Searcher**.

If the BASIC program affects any memory between 1536 and 1791, *do not* run the BASIC program until after using **Variable Searcher**. Changing this memory will crash the **Searcher**—and maybe the whole program. But, should this memory area be changed, you'll need to reload the loader program and run it, to fix the **Searcher**.

If your program is long and has many variables, all the variables and lines may not fit on the screen without scrolling. You can pause the action with CTRL-1, and resume it by pressing CTRL-1 again.

Variable Searcher automatically sets the left border margin to *0*. This will allow the output to be neat on the screen. **Searcher**, however, does not set the TABs, so altering them may cause the output to be messy. In such a case, use the default tab setting for best results.

The **Variable Searcher** works in a way very similar to the LVAR command in the OSS BASIC XL and BASIC XE cartridges. It works on any 8-bit Atari with or without DOS booted, and works all right with most versions of DOS, including DOS 2.0, DOS 2.5 and DOS 3.

Steven Anderson has studied advanced BASIC, machine language, Logo and Pascal. He bought his first computer, an 800XL, in 1984—disregarding his school's strange decision to stock Commodore 64s. He's interested in graphics and sound programs.

The two-letter checksum code preceding the line numbers here is *not* a part of the BASIC program. For further information, see the *BASIC Editor II*, in issue 47.

Listing 1.
BASIC listing.

```
YM 100 REM * VARIABLE SEARCHER *
CD 110 REM BY  STEVEN ANDERSON *
FT 120 REM * TYPE X=USR(1536) TO ACTIVATE
SN 130 REM FOR ALL ATARI 8-BIT COMPUTERS
XO 140 REM USES PAGE 6 OF MEMORY.
JO 200 FOR I=1536 TO 1791:READ D:POKE I,D
   :NEXT I
PV 205 POKE 203,69
FV 207 POKE 204,58:POKE 205,88:POKE 206,1
   55
NT 210 END
EB 1000 DATA 165,130,133,0,165,131,133,1,
   169,127,133,209,162,48,169,203
DJ 1010 DATA 157,68,3,104,133,82,157,69,3
   ,157,75,3,169,8,157,74
VO 1020 DATA 3,169,3,157,66,3,32,86,228,1
   60,0,177,0,16,16,41
IM 1030 DATA 127,32,93,6,32,91,6,230,209,
   32,111,6,32,91,6,32
OL 1040 DATA 93,6,230,0,208,2,230,1,165,0
   ,197,132,208,219,165,1
IG 1050 DATA 197,133,208,213,169,12,141,1
   14,3,208,15,169,155,162,11,142
EB 1060 DATA 114,3,162,0,142,72,3,142,73,
   3,162,48,76,86,228,165
YM 1070 DATA 136,133,2,165,137,133,3,32,2
   44,6,133,207,32,244,6,133
TO 1080 DATA 208,201,128,176,121,32,244,6
   ,32,244,6,32,244,6,201,2
AD 1090 DATA 176,9,32,244,6,201,155,208,2
   49,240,220,32,244,6,201,20
KI 1100 DATA 240,230,201,27,240,226,201,2
   2,240,205,201,14,208,10,162,6
KV 1110 DATA 32,244,6,202,208,250,240,227
   ,201,15,208,6,32,244,6,170
ZD 1120 DATA 208,238,197,209,208,213,165,
   207,133,212,165,208,133,213,32,170
GZ 1130 DATA 217,32,230,216,160,255,132,1
   75,230,175,164,175,192,8,240,187
HG 1140 DATA 177,243,72,41,127,32,93,6,10
   4,16,237,169,160,164,175,200
SI 1150 DATA 145,243,208,228,160,0,177,2,
   230,2,208,2,230,3,96,0
RD 1160 DATA 230,2,208,2,230,3,96,0
```

●

Window Graphics

Use player-missile graphics for joystick-controlled windowing.

by Howard Green

Have your ever wished that your Atari had the capability to display only a small portion of a graphics screen at a time? Imagine having a joystick-controlled graphics window which could be moved around the playfield to expose only the graphics directly "underneath" it. This technique is not only possible, but rather easy to implement in any BASIC program. If you know anything about player-missile graphics, you should have no problems adding **Window Graphics** to your own games and graphics displays.

Typing it in.

The best way to get started with **Window Graphics** is to type in the demo program shown in Listing 1, then run it. The program requires a joystick in port 1. This demonstration should give you a better feel for what **Window Graphics** are and how they can be used.

The program is intentionally uncomplicated. The demo allows only horizontal movement of the graphics window, but, with a little knowledge of player-missile graphics, vertical movement of the window is easily achieved.

How it works.

The secret to **Window Graphics** lies in location 623 of your computer's memory, otherwise known as the "Priority Selection Register." This register allows you to select which screen objects will be "in front" of others. These priorities are normally selected by setting one of the first 4 bits in this memory location. There's a different order of priority for each of these 4 bits (see the table following). For example: by setting bit 0, you can instruct the computer to give priority to the players, the playfield and the background, in that order.

If we set more than one of these first 4 bits, we get some very interesting results. If location 623 is POKEd with the number 5 (4+1), both bits 3 and 0 are set simultaneously. This causes what's known as a "priority conflict" to occur. Several conflicts occur in this case, but the one that the following demo program takes advantage of is the conflict which occurs between player 0 and playfield register 0.

Priority Table.

BIT	DECIMAL	PRIORITY CAUSED
0	1	PL 0-3, PF 0-3, BAK
1	2	PL 0-1, PF 0-3, PL 2-3, BAK
2	4	PF 0-3, PL 0-3, BAK
3	8	PF 0-1, PL 0-3, PF 2-3, BAK

PL=Player PF=Playfield BAK=Background

When ANTIC receives a conflicting priority signal, it simply displays the areas of overlap in black. So, in our demo, when player 0 overlaps the word *ANALOG*, the previously hidden letters (playfield register 0) turn black.

The theory behind why **Window Graphics** works is admittedly a bit complicated, but you really don't need to fully understand the theory in order to create some impressive displays with this technique. All you need to do is follow these simple steps:

(1) Create a display using color 0 and set color 0 to the same hue and luminance as the background. This can be done in text modes 1 and 2, as well as graphics modes 3, 5 and 7.

(2). Set up and define the graphics for player 0 in any shape you wish. This player will be your "Graphics Window." If you aren't familiar with player-missile graphics, consult previous **ANALOG Computing** articles for more information.

(3) By programming your game or display program to move player 0 over your "hidden" display, you can cause the parts of the display directly underneath player 0 to appear.

Window Graphics *continued*

(4) If you wish, you can use player 2 to frame your window as I did in the demo.

It would take pages to completely explore the potential of **Window Graphics**. I'll leave some of that exploring up to you.

There are many exciting uses for this technique, only a few of which are discussed in this article. If you're interested in further experimentation with this effect, you might want to consult *Mapping The Atari* by Ian Chadwick for further information about the priority register. Try POKEing location 623 with other values and see what you can come up with. Good luck and have fun. I'm looking forward to seeing some spectacular results.

Howard Green is an undergraduate student at Marquette University, where he's studying Civil Engineering. He plans to specialize in Construction Management and will soon begin work as a co-op student for Opus Corporation in Minneapolis. When he isn't programming, he uses the Atari to write reports for school.

The two-letter checksum code preceding the line numbers here is *not* a part of the BASIC program. For further information, see the *BASIC Editor II*, in issue 47.

Listing 1.
BASIC listing.

```
UL 10 REM WINDOW GRAPHICS DEMO
OK 20 REM BY HOWARD GREEN
UX 30 REM FOR ANALOG COMPUTING
IE 40 DIM HM(15)
LJ 50 GRAPHICS 2:POKE 704,9:POKE 706,14:P
   OKE 710,130:POKE 709,14:POKE 752,1
UH 60 POKE 708,130:POKE 712,130
WI 70 POSITION 4,1:? #6;"A.N.A.L.O.G."
TF 80 POSITION 3,6:? #6;"window graphics"
ZR 90 POSITION 4,7:? #6;"demonstration"
YQ 100 ? "   PRESS START TO RETURN TO BAS
   IC"
CY 110 A=PEEK(106)-8:POKE 54279,A
WW 120 START=256*A+512:POKE 559,46:POKE 5
   3277,3:POKE 623,5
CC 130 FOR X=START+20 TO START+35:POKE X,
   126:NEXT X
NH 140 START=START+256
NN 150 FOR X=START+19 TO START+36:POKE X,
   255:NEXT X
LO 160 H=48:POKE 53256,3:POKE 53258,3
RO 170 FOR N=1 TO 15:HM(N)=0:NEXT N
DA 180 HM(7)=1:HM(11)=-1
AD 190 REM MAIN MOVE ROUTINE
BG 200 S=STICK(0):H=H+HM(5)
SC 210 IF H>175 THEN H=175
FP 220 IF H<48 THEN H=48
VL 230 POKE 53248,H:POKE 53250,H
SC 240 IF PEEK(53279)<>6 THEN 200
ZE 250 POKE 53248,0:POKE 53250,0:GRAPHICS
    0:END
```

●

Mouse Maker
(version 3.5)

Everything you wanted to know about your ST's mouse.

by Saveen V. Reddy

The mouse. No, not the furry one you see in pet stores, but the one hooked up to your ST. Indeed, it's quite useful —a lot faster than a joystick and easier to hold than a touch tablet. And you can even change the way it appears onscreen, with **Mouse Maker**.

Make your own mouse cursor images, save or load them, and even save them in a form you can use from ST BASIC. You can build up a library of shapes to use in your programs. If you don't want to define your own shapes, this article will show you how to use the ones built into GEM.

Setting up.

Before you use this program, create a folder named CURSOR on the disk in drive A, then enter BASIC. Type in the SETINGUP.BAS program, save it and run it. You need only do this once. The program sets up a very important file used by **Mouse Maker**. This file will reside in the CURSOR folder, along with the data files created by **Mouse Maker**, so they don't clutter up the root directory.

Typing it in.

Mouse Maker should be typed exactly as it appears in the listing. However, near the beginning of the program, the variable OFF should temporarily be set to *0*. It determines whether GEM is turned on or not. Once you have saved the program, run it to check for bugs. If there are any, make sure you get them fixed. If ST BASIC tries to print an error message while GEM is turned off, you will have to reboot. Once the program works properly, change OFF to equal *1* as printed in the listing. When you run **Mouse Maker**, you must load it from drive A. **Mouse Maker** works properly only in medium resolution.

How to draw the cursor.

First, select your color from the menu at the bottom right of the screen. The menu box labeled *BACKGROUND* is the color green (color index 3). The ones labeled *BLACK* and *WHITE* are self explanatory. Just select the color you want to draw with, go to the grid, and press the left button. The background color isn't actually present in the display. I didn't allow other colors to be used for the mouse cursor images, because I wanted the *images* to be usable in all resolutions.

For those with monochrome monitors.

Mouse Maker was written using a color monitor. It does not work in monochrome. With a few adjustments, however, it *should* work in high resolution. Most of the modifications require the adjustment of Y-coordinates and use of different fill patterns, instead of color registers 2 and 3. It will require some degree of work on your part to do this.

Functions.

SAVE, LOAD, BASIC SAVE.

The SAVE and LOAD functions permit you to save and load the cursor image files. You will not be asked for a filename to save or load, but for a number. These numbers correspond to the numbers listed by the DIR function. When using the SAVE function, you will also be asked for a filename.

The BASIC SAVE function saves the image in a BASIC program form which you can merge with your own programs. Before you select this function, you should set the action point of the mouse. You do this by pressing the right mouse button. If it is not set, it defaults to *0,0* (upper left corner). You will be asked for a filename. Don't type in an extension—the extension *.BAS* will automatically be added. Then you'll be asked for the starting line number. The increment is preset to ten. When you merge the file into

When you want to talk Atari

XL/XE HARDWARE

INTERFACES

ICD	
P:R Connection	$59.99
Printer Connection	44.99
Supra	
Microprint	39.99
1150	39.99
1151 (1200 XL)	39.99
Xetec	
Graphix Interface	39.99
Atari	
850 Interface	109.00

COMPUTERS

Atari 130XE **$129**

65 XE	89.99
800 XL	68.99
800 XL Package (1050 Drive, 1025 Printer, Joystick & Diskettes)	329.00

MEMORY

Axlon	
32K Board	19.99
48K Board	Call
128K RAM Disk	Call

MODEMS

Atari	
SX212 300/1200 (ST)	99.99
835	24.99
XMM301	44.99
Anchor	
VM520 300/1200 Direct Connect (ST)	129.00
Volksmodem 1200	99.99
Supra	
1200 AT	139.00
1200 ST	139.00

ST HARDWARE

1040 Monochrome System **$719**

Includes: 1040ST, 1 mb RAM with 3½" drive built-in, 192K ROM with TOS, Basic, power supply and monochrome monitor.

1040 RGB/Color System	879.00
520 Monochrome System (Includes: 520ST computer with modulator, disk drive, mouse, logo, Basic and monochrome monitor)	call
520 RGB/Color System	769.00
SM124 Monochrome Monitor	159.00
SM1224 Color Monitor	329.00
1040 Computer (no monitor)	589.00

Call on the new Atari: 520ST FM/PC compatible/Laser Printer/Daisywheel.

DRIVES

Atari 354 SS/DD Disk Drive (ST) **$139**

Atari	
1050 SS/DD Drive (XL/XE)	$139.00
SHD204 20 Meg Hard Drive (ST)	599.00
AA314 DS/DD Disk (ST)	219.00
Indus	
GT Disk Drive (XL/XE)	179.00
Supra	
20 Meg Hard Drive (XL/XE)	679.00
20 Meg Hard Drive (ST)	549.00
30 Meg Hard Drive (ST)	759.00

PRINTERS

Atari	
1020 XL/XE	$29.99
1025 XL/XE	129.00
XDM121 XL/XE	209.00
XMM801 XL/XE	199.00

Atari XMM804 ST **$189**

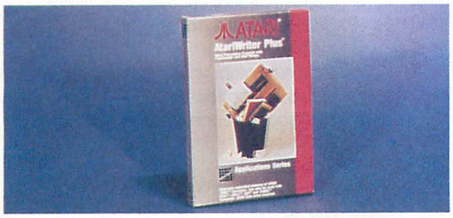
Brother	
M-1109 100 cps (80 col.)	209.00
Citizen	
MSP-10 (80 col.)	289.00
Premier 35 cps Daisywheel	489.00
C.Itoh	
Prowriter 1550P (132 col.)	249.00
Prowriter 8510SP	Call
Epson	
LX86, (120 cps)	199.00
FX86E, FX286E (240 cps)	Call
EX800, EX1000	Call
LQ800, LQ1000	Call
Juki	
5510 Color Dot	349.00
6000 Letter Quality	189.00
6100 10 cps Daisywheel	369.00
NEC	
ELF360 20 cps	389.00
3000 Series Spinwriter 35 cps	779.00
Pinwriter 560-660-760	Call
Okidata	
Okimate 20	129.00
ML-182, ML-192, ML-193	Call
ML-292, ML-293	Call
Panasonic	
KX-1080i 120 cps 80 col.	209.00
KX-1091i 180 cps 132 col.	299.00
Star Micronics	
NX-10 120 cps 80 col.	209.00
NX-15 120 cps 132 col.	379.00
Toshiba	
P321 216 cps 24 pin	479.00

COMPUTER MAIL ORDER

.......you want to talk to us.

XL/XE SOFTWARE

Access	
Leaderboard Golf	$25.99
Executive Disk	14.99
Accolade	
Fight Night	19.99
Hardball	19.99
Atari	
Various Titles from	$3.49
Batteries Included	
Paperclip w/Spellpack	37.99
B-Graph	37.99
Broderbund	
Printshop	26.99
Graphics Library 1-2-3 (ea.)	16.99
Cygnus	
Starfleet I	32.99
Datasoft	
221 Baker St.	21.99
Electronic Arts	
Touchdown Football	12.99
Chessmaster 2000	29.99
Epyx	
World Karate Championship	19.99
Infocom	
Zork III	24.99
Microprose	
Top Gunner	17.99
F-15 Strike Eagle	21.99
Origin Systems	
Ultima 4	36.99
Roklyn	
Gorf/Wizard of Wor/AT Deluxe	
Anti-Sub/Journey to Planet (ea.)	3.99
Strategic Simulations	
Gemstone Warrior	12.99
Sublogic	
Flight Simulator II	33.99
X-Lent	
Typesetter	21.99



AtariWriter Plus **$34**

ACCESSORIES

Maxell	
MD1-M SS/DD 5¼"	$8.49
MD2-DM DS/DD 5¼"	9.49
MF1-DDM SS/DD 3½"	12.49
MF2-DDM DS/DD 3½"	18.49
Sony	
MD1D SS/DD 5¼"	7.99
MD2D DS/DD 5¼"	9.49
MFD-1DD SS/DD 3½"	12.99
MFD-2DD DS/DD 3½"	19.99
Allsop Disk Holders	
Disk File 60-5¼"	9.99
Disk File 30-3½"	9.99
Curtis	
Emerald	39.99
Safe Strip	19.99
Universal Printer Stand	14.99
Tool Kit	22.99
ICD (XL/XE)	
Sparta DOS Contruction Set	28.99
US Doubler/Sparta DOS	48.99
Real Time Clock	49.99
Rambo XL	28.99
Multi I/O Board 256K	169.00
Multi I/O Board 1 Meg	289.00

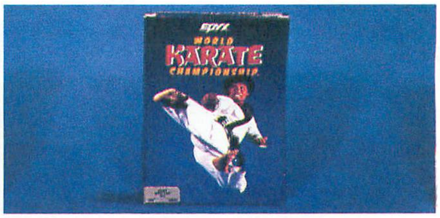
ST SOFTWARE

Access Leaderboard Golf **$24.99**

Activision	
Music Studio	34.99
Antic	
Cad 3-D	30.99
Atari	
Star Raiders	19.99
Batteries Included	
Degas Elite	48.99
Thunder	25.99
Cygnus	
Starfleet I	35.99

ST SOFTWARE

Ditek	
STA Accounts	$149.00
Electronic Arts	
Chessmaster 2000	29.99



Epyx World Karate Championship **$23.99**

Championship Wrestling	25.99
Haba	
Habawriter	32.99
Hippopotamus	
Hippo Backgammon	27.99
Infocom	
Zork Trilogy	42.99
Hitchhiker's Guide	29.99
Michtron	
Major Motion	25.99
Microprose	
Silent Service	23.99
F-15 Strike Eagle	29.99
Mindscape	
High Roller	34.99
OSS	
Personal Pascal	46.99
Paradox	
Wanderer (3-D)	25.99
Polarware	
Crimson Crown	26.99
Psygnosis	
Arena	25.99
Sierra On Line	
Winnie The Pooh (Educ.)	18.99
Sublogic	
Flight Simulator II	33.99
Timeworks	
Wordwriter	49.99
VIP	
Professional (GEM)	144.99
X-Lent	
Typesetter	27.99

In the U.S.A. and in Canada

Call toll-free: 1-800-233-8950.

Outside the U.S.A. call 717-327-9575 Telex 5106017898

Educational, Governmental and Corporate Organizations call toll-free 1-800-221-4283

CMO. 477 East Third Street, Dept. B707, Williamsport, PA 17701

ALL MAJOR CREDIT CARDS ACCEPTED.

POLICY: Add 3% (minimum $7.00) shipping and handling. Larger shipments may require additional charges. Personal and company checks require 3 weeks to clear. For faster delivery use your credit card or send cashier's check or bank money order. Pennsylvania residents add 6% sales tax. All prices are U.S.A. prices and are subject to change and all items are subject to availability. Defective software will be replaced with the same item only. Hardware will be replaced or repaired at our discretion within the terms and limits of the manufacturer's warranty. We cannot guarantee compatibility. All sales are final and returned shipments are subject to a restocking fee.

CIRCLE #107 ON READER SERVICE CARD

CIRCLE #108 ON READER SERVICE CARD

your own program, you must give the beginning line number a LABEL and add a RETURN statement at the end. This way, it can be called whenever you need it. Remember, every time a FORM_ALERT is called, the mouse is changed back to its original shape. So you must reset it every time you use a FORM_ALERT box.

CLEAR IMAGE.

This will clear the current mouse image data.

DISK FUNCTIONS.

This function will display a FORM_ALERT box with three choices: DIR, RENAME and CANCEL.

Selecting DIR will print out the numbers and names of all mouse cursor image files on the disk in drive A. Files that don't exist are listed as "********".

By the way, this information doesn't come directly from the disk directory. Since there is no way to execute the DIR command from a BASIC program, the program simulates it by writing all data filenames into a disk file. When you run **Mouse Maker**, the "directory" file is loaded into a string array. Every time something is done to change the names, the altered contents of the string array are written back into the file. The string array is thus kept up to date.

So, the DIR function works without accessing the disk drive, thus no wait is required for reading the directory. The RENAME option allows you to rename any file you choose. If you wish to delete a file, first rename it to "********" and then go to the desktop and delete it.

CUSTOM FUNCTION.

The box labeled *CUSTOM FUNCTION* has no use. It was provided in case you need an extra function not available in the program. If you do activate it, you will see a FORM_ALERT box. Selecting either of the two choices presented returns you back to the main program.

If you need some sort of special function in the program (such as scrolling the cursor image), then write a subroutine to perform the task, and then use CUSTOM FUNCTION to call it when needed.

You needn't worry about the mouse's X,Y-coordinates; everything is already set up. Just change the FORM_ALERT box to meet your specifications, and add whatever you want. Instructions on how to use the FORM_ALERT boxes are given later in this article.

Because this program turns GEM off for speed, you need to turn it on during the time you add something to it. This way, if an error occurs, you won't have to reboot. To automatically have GEM on at all times, look toward the beginning of the program for the variable OFF. Change it to the value *0*. Once you've finished adding whatever you wanted and have made it completely bug-proof, give it its previous value of *1*.

The subroutines are listed in Table 1.

FORM_ALERT.

The FORM_ALERT box is a great boon to ST BASIC users. It allows you to have a professional looking, easy-to-use, foolproof way to get a response from the user. The calling of the subroutine is simple.

(1) Set the string alrt$ to the proper form for the FORM_ALERT box.

(2) If you want one of the choices to be selected by pressing RETURN, set the variable CHO to the one desired. A value of *0* means none of them can be selected in such a fashion.

(3) Call the subroutine ALERT.

(4) Check the value of CHOICE to see which one is selected.

(5) Do whatever the choice indicates.

Here is the form the alrt$ should take:

```
alrt$="[symbol][message][choices]"
```

The symbol is a number, *1* through *3*. A value of *1* displays an exclamation point, *2* a question mark, *3* a stop sign.

A quick look through the program will reveal that all the FORM_ALERT messages have the format of: [symbol][|message][choices]. An extra "|" character is included just before the message text begins. I do this because I like to see some space between the top of the box and the text inside it.

Table 1.

SUBROUTINE NAME	FUNCTION
BLK	Draws a box in the main grid. The variables *BX* and *BY* determine its location.
BC	Sets the fill color for the box. The variable *FC* represents the fill color.
BOX1	Draws the low- and medium-resolution pixels in the image.
DESTROY	Erases all the text in the text area.
HIDECURSOR	Hides mouse cursor.
SHOWCURSOR	Shows mouse cursor.
ALERT	Displays a *FORM_ALERT* box. The string *alrt$* contains the message and symbols displayed (see below).
MOUSE	Returns the mouse X,Y-coordinates and indicates if a button is pressed. *MSX,MSY* are the X,Y-coordinates. *MSB* indicates the button status: 0=no button pressed, 1=left button pressed, 2=right button pressed.

The message cannot have too many characters, or the program crashes. The number of characters you can fit into each line is resolution-dependent. To separate the message into different lines add a "|" character between the areas to be separated.

You can have up to three different choices. Again, don't make them too long. The choices must be separated by the "|" character. Here's an example:

```
alrt$="[1][|Sample Message]
       [choice1|choice2]"
```

Using the preset images.

In order to use the preset mouse images, you must first call the GEMSTUFF subroutine in this program. It needs to be called only once. It sets up the variables used by the GEMSYS command. Next, poke GINTIN with the number of the image you want. The list of image numbers is below. Finally, use the statement GEMSYS(78), and a preset image will appear as your mouse cursor.

0	The arrow (default).
1	Cursor.
2	The bee image.
3	A hand with the index finger extended.
4	An open hand.
5	Cross hair.

6 . . . . A very thick cross hair.
7 . . . . An outline of the cross hair.

If you come up with anything great for the custom function, tell me. If you use Delphi, just drop me a message. My username is RHODAN. I'll welcome any suggestions.

Saveen V. Reddy is a student at Man High School in Man, West Virginia. He owns a 1040ST and just began learning C.

Listing 1.
ST BASIC listing.

```
10 ' setting up
20 open "O",#1,"a:\cursor\names.fil"
30 for i=1 to 50
40 print #1,"********"
50 next i:close #1
60 end
```

●

ST CHECKSUM DATA.
(see page 38)

```
10 data  855, 567, 762, 268, 157, 724
, 3333
```

●

Listing 2.
ST BASIC listing.

```
10 ' MOUSE MAKER ver. 3.5
20 ' by Saveen V. Reddy
40 randomize 0:onn=0:off=1
50 on error goto  4040
70 textx=100:texty=100:cho=1:poke syst
ab+24,off
80 dim cursor(32),filename$(1,8),text$
(1,40),fn$(1,12),fileline$(1,80)
90 dim a$(1,100),finame$(12),alrt$(80)
,namesfile$(50),oldc(4,3)
130 gosub CHANGECOLORS
140 fullw 2:clearw 2:gosub WELCOME:cle
arw 2
150 gosub HIDECURSOR
160 gosub BOXDRAW
170 color 1,1,1,1,1:gosub SETUP
180 textcolor=1:gosub DIRECTION:gosub
GETFILE
220 MAIN:
230 gosub SHOWCURSOR
240 gosub READKEY
250 goto MAIN
290 READKEY:
300 gosub MOUSE:bx=int((msx-1)/16):by=
int((msy+1)/8-3)
310 if bx<0 then bx=0
320 if by<0 then by=0
330 if bx>15 then bx=15
340 if by>15 then by=15
350 if msb=2 then ax=bx:ay=by
360 if msb=1 and (msx>258 or msy>150)
then goto SELECTION
370 if msb=1 then fc=q:gosub BC:gosub
HIDECURSOR:gosub BOK:gosub SHOWCURSOR
380 return
420 SELECTION:
430 if msx>261 and msy>141 and msx<355
 and msy<151 then q=3
440 if msx>356 and msy>141 and msx<407
 and msy<151 then q=1
450 if msx>408 and msy>141 and msx<510
 and msy<151 then q=0
460 if msx>261 and msy>151 and msx<312
 and msy<161 then gosub DATALOAD
470 if msx>312 and msy>151 and msx<359
 and msy<161 then gosub DATASAVE
480 if msx>360 and msy>151 and msx<455
 and msy<161 then gosub DISKPUT
490 if msx>456 and msy>151 and msx<510
 and msy<161 then gosub QUITPROGRAM
500 if msx>261 and msy>161 and msx<370
 and msy<171 then gosub INSTRUC
510 if msx>371 and msy>161 and msx<510
 and msy<171 then gosub FUNCTION1
520 if msx>261 and msy>171 and msx<370
 and msy<181 then gosub IMAGECLEAR
530 if msx>371 and msy>171 and msx<510
 and msy<181 then gosub DISKFUNCTIONS
540 goto 380
580 TRANSLATE:
590 for ii=0 to 15:count=15:tot=0
600 for i=0 to 15
610 bx=i:by=ii:gosub LOCATE
620 if uc=0 then tot=tot+(2^count)
630 count=count-1
640 next i:cursor(ii+1)=tot:next ii
650 for ii=0 to 15:count=15:tot=0
660 for i=0 to 15
670 bx=i:by=ii:gosub LOCATE
680 if uc=1 then tot=tot+(2^count)
690 count=count-1
700 next i:cursor(ii+17)=tot:next ii
710 return
750 BC:
760 poke contrl,25:poke contrl+2,0:pok
e contrl+6,1
770 poke intin,fc:vdisys(1):return
810 BLK:
820 poke contrl,11:poke contrl+2,2:pok
e contrl+6,0
830 poke contrl+10,1:poke ptsin,(16*bx
)+1:poke ptsin+2,(8*by)+23
840 poke ptsin+4,(16*bx)+15   :poke pt
sin+6,(8*by)+29
850 vdisys(1):return
890 MP:
900 poke contrl,5:poke contrl+2,0:poke
 contrl+6,0
910 poke contrl+10,15:vdisys(1):mx=pee
k(intout)
920 my=peek(intout+2):return
960 LOCATE:
970 poke contrl,105:poke contrl+2,1:po
ke contrl+6,0
980 poke ptsin,(16*bx)+3:poke ptsin+2,
(8*by)+25
990 vdisys(1):uc=peek(intout+2):return
1030 MB:
1040 poke contrl,124:poke contrl+2,0:p
oke contrl+6,0
1050 vdisys(1):but=peek(intout):return
1090 SBLK:
1100 zc=c3:fc=zc:gosub BC
1110 poke contrl,11:poke contrl+2,2:po
ke contrl+6,0
1120 poke contrl+10,1:poke ptsin,(16*b
x)+5:poke ptsin+2,(8*by)+25
1130 poke ptsin+4,(16*bx)+10   :poke p
tsin+6,(8*by)+27
1140 vdisys(1):return
1180 BOXDRAW:
1190 fc=3:gosub BC
1200 poke contrl,11:poke contrl+2,2:po
ke contrl+6,0
1210 poke contrl+10,1:poke ptsin,1:pok
e ptsin+2,22
1220 poke ptsin+4,(16*15)+18:poke ptsi
n+6,(8*19)+36:vdisys(1)
1230 gosub BOXING
1240 return
1280 DISKPUT:
1290 restore 1340
```

```
1300 gosub FIXTEX
1310 alrt$="[3][|Save cursor in BASIC
form?][Yes|No]":gosub ALERT
1320 if choice=2 then return
1330 gosub HIDECURSOR
1340 data Please wait while the cursor
 data,is being processed
1350 data The data work is finished.,P
lease type in the filename you
1360 data want to save cursor as.  Do
not,type the drive or extender.
1370 data Enter begining line number.
1400 for i=1 to 2:read text$:gotoxy 32
,i:? text$:next i:gosub HIDECURSOR
1410 gosub TRANSLATE
1420 for i=1 to 2:gotoxy 32,i:? space$
(36):next i
1430 for i=1 to 4:read text$:gotoxy 32
,i:? text$:next i:poke systab+24,onn
1440 gotoxy 32,i:line input filename$:
poke systab+24,off
1450 filename$=filename$+".bas"
1460 for i=1 to 5:gotoxy 32,i:? space$
(36):next i
1470 read text$:gotoxy 32,1:? text$:po
ke systab+24,onn
1480 gotoxy 32,2:input linenum:poke sy
stab+24,off
1490 for i=1 to 5:gotoxy 32,i:? space$
(36):next i
1500 open "O",#1,filename$:print #1,st
r$(linenum)+" '"
1510 print #1,str$(linenum+10)+" '/"+s
tring$(45,"-")+"/"
1520 print #1,str$(linenum+20)+" '":li
nenum=linenum+30
1530 for i=1 to 32  step 4
1540 filen$=str$(linenum+(10*(i-1)))+"
 data "+str$(cursor(i))
1550 filen$=filen$+","+str$(cursor(i+1
))+","+str$(cursor(i+2))
1560 filen$=filen$+","+str$(cursor(i+3
))
1570 print #1,filen$
1580 next i:ln=linenum
1590 print #1,str$(ln+320)+" '"
1600 print #1,str$(ln+330)+" poke cont
rl,111"
1610 print #1,str$(ln+340)+" poke cont
rl+2,0"
1620 print #1,str$(ln+350)+" poke cont
rl+6,37"
1630 print #1,str$(ln+360)+" poke inti
n,"+str$(ax)+":' action point x"
1640 print #1,str$(ln+370)+" poke inti
n+2,"+str$(ay)+":' action point y"
1650 print #1,str$(ln+380)+" poke inti
n+4,1"
1660 print #1,str$(ln+390)+" poke inti
n+6,0:' mask color index"
1670 print #1,str$(ln+400)+" poke inti
n+8,1:' data color index"
1680 print #1,str$(ln+410)+" for i=10
to 72 step 2
1690 print #1,str$(ln+420)+" read zzt:
poke intin+i,zzt:next i"
1700 print #1,str$(ln+430)+" vdisys(1)
"
1710 close #1
1720 gotoxy 32,0:? space$(36):gosub SH
OWCURSOR
1730 return
1770 DIRECTION:
1780 mode=3:gosub WRITEMODE
1790 tex=0:gosub TEXT:textx=(33*8):the
ight=9:gosub TEXTHEIGHT
1800 texty=149:a$=" background black
white":gosub TEXTPRINT
1810 texty=159:a$=" load  save  BASIC
save  Quit":gosub TEXTPRINT
1820 texty=169:a$=" instructions  cust
om function":gosub TEXTPRINT
1830 texty=179:a$=" clear image   disk
 functions          ":gosub TEXTPRINT
1840 theight=9:gosub TEXTHEIGHT
1850 tex=0:gosub TEXT:mode=1:gosub WRI
TEMODE
1860 return
1900 TEXT:
1910 poke contrl,106:poke contrl+2,0:p
oke contrl+6,1
1920 poke intin,tex:vdisys(1):return
1960 DOT1:
1970 poke contrl,11:poke contrl+2,2:po
ke contrl+6,0
1980 poke contrl+10,1:poke ptsin,(dx):
poke ptsin+2,(dy)
1990 poke ptsin+4,(dx+1):poke ptsin+6,
(dy)
2000 vdisys(1):return
2040 DOT2:
2050 poke contrl,11:poke contrl+2,2:po
ke contrl+6,0
2060 poke contrl+10,1:poke ptsin,(dx):
poke ptsin+2,(dy)
2070 poke ptsin+4,(dx):poke ptsin+6,(d
y)
2080 vdisys(1):return
2120 INSTRUC:
2130 alrt$="[1][INSTRUCTIONS:|Left but
ton|selects function.|Right button|"
2140 alrt$=alrt$+"sets action point][o
k]":gosub ALERT
2150 return
2190 FUNCTION1:
2200 alrt$="[1][|Custom function][opti
on1|option2]"
2210 gosub ALERT
2220 return
2260 TEXTPRINT:
2270 poke contrl,8:poke contrl+2,1:pok
e contrl+6,len(a$)+1
2280 poke ptsin,textx:poke ptsin+2,tex
ty
2290 for m=1 to len(a$)
2300 poke intin+(m-1)*2,asc(mid$(a$,m,
1))
2310 next m:poke intin+(m-1)*2,0:vdisy
s(1):return
2350 SETUP:
2360 poke systab+2,1:gosub GEMSTUFF
2370 poke contrl,22:poke contrl+2,0:po
ke contrl+6,1
2380 poke intin,2:vdisys(1)
2390 linef 0,132,258,132
2400 linef 5,136,120,136:linef 120,164
,5,164:mode=2:gosub WRITEMODE
2410 linef 120,136,120,164:linef 5,136
,5,164:gosub FIXTEX
2420 color 1,2,1,1,1:fill 2,134:tex=1:
gosub TEXT
2440 textcolor=0:gosub TEXCOL
2450 textx=17*8:texty=21*8:a$="image":
gosub TEXTPRINT
2460 texty=22*8:a$="area":gosub TEXTPR
INT:mode=1:gosub WRITEMODE
2480 linef 258,0,258,190:linef 259,0,2
59,190
2490 linef 4,136,4,164:linef 121,136,1
21,164
2500 linef 258,118,620,118
2510 linef 258,108,620,108:textx=260
2530 color 1,2,1,1,1:fill 260,112
2540 linef 263,0,263,105:linef 264,0,2
64,105
2550 linef 263,105,273,100:linef 273,1
```

```
00,570,100
2560 linef 570,100,580,95:linef 580,95
,580,0
2580 linef 581,95,581,0
2590 color 1,1,1,1,1:fill 582,0:fill 5
82,150
2600 gosub BARS
2610 gosub PLSETUP
2630 plx=261:ply=141:plx1=355:ply1=151
:gosub POLYLINE
2640 plx=356:ply=141:plx1=407:ply1=151
:gosub POLYLINE
2650 plx=408:ply=141:plx1=510:ply1=151
:gosub POLYLINE
2660 plx=261:ply=151:plx1=312:ply1=161
:gosub POLYLINE
2670 plx=312:ply=151:plx1=359:ply1=161
:gosub POLYLINE
2680 plx=360:ply=151:plx1=455:ply1=161
:gosub POLYLINE
2690 plx=456:ply=151:plx1=510:ply1=161
:gosub POLYLINE
2700 plx=261:ply=161:plx1=370:ply1=171
:gosub POLYLINE
2710 plx=371:ply=161:plx1=510:ply1=171
:gosub POLYLINE
2720 plx=261:ply=171:plx1=370:ply1=181
:gosub POLYLINE
2730 plx=371:ply=171:plx1=510:ply1=181
:gosub POLYLINE
2740 return
2780 TEXTHEIGHT:
2790 poke contrl,107:poke contrl+2,0:p
oke contrl+6,1
2800 poke intin,theight:vdisys(1):retu
rn
2840 FIXTEX:
2850 poke contrl,106:poke contrl+2,0:p
oke contrl+6,1:poke intin,0
2860 vdisys(1):poke contrl,107:poke co
ntrl+2,0:poke contrl+6,1
2870 poke intin,9:vdisys(1):poke contr
l,22:poke contrl+2,0
2880 poke contrl+6,1:poke intin,1:vdis
ys(1):return
2950 DATASAVE:
2960 gosub FIXTEX
2970 alrt$="[3][|Save cursor form data
 file?][Yes|No]":gosub ALERT
2980 if choice=2 then return
3000 gosub HIDECURSOR
3010 gotoxy 32,0:? "Wait while cursor
data"
3030 gotoxy 32,1:? "is being processed
."
3050 gosub HIDECURSOR:gosub TRANSLATE
3060 gotoxy 32,0:? space$(21)
3070 gotoxy 32,1:? space$(21)
3080 gosub FIXTEX:color 1
3090 poke systab+24,onn:gotoxy 32,0
3100 ? "Which data file(1-50,0 cancels
)"
3110 gotoxy 36,1:input datfil$:datafil
e=val(datfil$)
```

ATARI USERS' GROUPS

Modesto Atari Computer Club (MACC)
P.O. Box 3811, Modesto, CA 95352
Meetings; BBS; newsletter. President: Robert Johnstone.

San Luis Obispo Atari Computer Enthusiasts
P.O. Box 4156, San Luis Obispo, CA 93403
Meetings; BBS; newsletter: *Slo-Pokes*.
President: Mark McMurry.

Conejo Valley Atari Users' Group (CVAUG)
587 "F" North Ventu Park Rd., Suite 123,
Thousand Oaks, CA 91320
Meetings; BBS; newsletter. President: Grant Peterson.

Pikes Peek Poke Atari Computer Enthusiasts (P∧3ACE)
P.O. Box 9282, Colorado Springs, CO 80935
Meetings; newsletter. President: Eric Kowalski.

Gainesville Atari Group (GAG)
6733-B SW 5th Place, Gainesville, FL 32607-1706
Meetings; newsletter. President: Darin L. Delegal.

Louisiana Atari Computer Enthusiasts (LACE)
6925 Oak Hill Circle, Shreveport, LA 71106
Meetings; BBS; newsletter. President: Mike Hinz.

Frederick Atari Computer Enthusiasts (FACE)
P.O. Box 300, Keedysville, MD 21756
Meetings; BBS; newsletter: *Current Notes*.
President: John Maschmeier.

Battle Creek Atari Users Group (BCAUG)
22554 Pine Lake Rd., Battle Creek, MI 49017
Meetings; newsletter. President: Chuck Steele.

Warrensburg Atari Computer Owners (WACO)
P.O. Box 199, Warrensburg, MO 64093
Meetings; BBS; newsletter: *W.A.C.O. Wramblings*.
President: Les Lynam.

Brooklyn Atari Society for Information
& Communication (BASIC)
c/o Peter J. Fazio,
2724 East 23rd St., Brooklyn, NY 11235
Meetings; newsletter: *Dateline: Atari*.

Long Island Atari Computer Enthusiasts (LIACE)
P.O. Box 382, Bellport, NY 11713
Meetings; newsletter. President: Mike Ferrara.

Keystone Area Atari Computer Enthusiasts (KACE)
740 Horner St., Johnstown, PA 15902
Meetings; BBS; newsletter.

Tri-Cities Atari Support Club
P.O. Box 1148, Petersburg, VA 23803
Meetings; newsletter: *TASC Times*.
President: John Tarnaski.

Milwaukee Area Atari Users' Group (MILATARI)
P.O. Box 19858-0858
Meetings; SIGs; BBS; newsletter. President: Ron Friedel.

ATTENTION USERS' GROUPS

If you would like your organization to be listed here, send information (and newsletter, if appropriate) to **ANALOG Computing** Group Listing, P.O. Box 23, Worcester, MA 01603. **ANALOG Computing** is not responsible for errors.

```
3130 if datafile=0 then gosub DESTROY
3140 if datafile=0 then gosub SHOWCURS
OR:poke systab+24,off:return
3150 if datafile<1 or datafile>50 then
 gosub dlalert:goto 3090
3170 finame$="a:\cursor\crsrdata."+dat
fil$
3180 poke systab+24,off
3200 open "O",#1,finame$:close #1:open
 "O",#1,finame$
3210 for i=1 to 32
3220 print #1,str$(cursor(i)):next i
3230 close #1
3250 gotoxy 32,0:gosub DESTROY
3260 gotoxy 32,0:? "Type the filename
for this file."
3270 poke systab+24,onn
3280 gotoxy 32,1:input quest$
3290 namesfile$(val(datfil$))=quest$
3300 poke systab+24,off:gosub DESTROY:
gosub PUTFILE
3310 gosub SHOWCURSOR
3320 return
3360 DATALOAD:
3370 alrt$="[3][|Load cursor form data
][Yes|No]":gosub ALERT
3380 if choice=2 then return
3400 gosub HIDECURSOR
3410 gosub fixtex:color 1:gosub BOXER:
gosub BARS
3420 gotoxy 32,0:? "Which data file(1-
50,0 cancels)"
```

```
3430 gotoxy 36,1
3440 poke systab+24,onn:input datfil$
3450 datafile=val(datfil$):poke systab
+24,off
3470 if datafile=0 then gosub DESTROY:
gosub SHOWCURSOR
3480 if datafile=0 then poke systab+24
,off:return
3490 if datafile<1 or datafile>50 then
 gosub DLALERT:goto 3430
3510 finame$="a:\cursor\crsrdata."+dat
fil$
3520 open "I",#1,finame$:close #1:open
 "I",#1,finame$
3530 for i=1 to 32:input #1,datastuff$
3540 cursor(i)=val(datastuff$):next i
3550 close #1
3570 gotoxy 32,0:? space$(36)
3580 gotoxy 32,1:? space$(36)
3590 gosub HIDECURSOR:gosub TRANSLATE2
3600 gosub DESTROY:gosub SHOWCURSOR
3610 return
3630 TRANSLATE2:
3640 fc=0:gosub BC:gab=0
3650 for ii= 1 to 16:ct=15
3660 for i=0 to 15
3670 if cursor(ii)=0 then goto 3690
3680 if cursor(ii)>((2^ct)-1) then gos
ub PLOTTER
3690 ct=ct-1:next i:next ii
3700 fc=1:gosub BC:gab=16
3710 for ii=17 to 32:ct=15:zz=ii-16
```

```
3720 for i=0 to 15
3730 if cursor(ii)=0 then goto 3750
3740 if cursor(ii)>((2^ct)-1) then gos
ub PLOTTER
3750 ct=ct-1:next i:next ii:return
3790 WRITEMODE:
3800 poke contrl,32:poke contrl+2,0:po
ke contrl+6,1
3810 poke intin,mode:vdisys(1):return
3850 QUITPROGRAM:
3860 alrt$="[3][|Quit Program?][YES|NO
]"
3870 gosub ALERT
3880 if choice=2 then return
3890 for i=1 to 4:cindex=i-1
3900 red=oldc(i,1):green=oldc(i,2)
3910 blue=oldc(i,3)
3920 gosub SETCOLOR:next i
3930 poke systab+24,onn
3940 end
3980 TEXCOL:
3990 poke contrl,22:poke contrl+2,0:po
ke contrl+6,1
4000 poke intin,textcolor:vdisys(2):re
turn
4040 ERASEAREA:
4050 for i=0 to 7
4060 gotoxy 32,i:? space$(36):next i:r
esume MAIN
4100 ALERT:
4110 gosub SHOWCURSOR
4120 gem#=addrin:poke gintin,cho
4130 alrt$=alrt$+chr$(0)+chr$(0)
4140 poke gem#,varptr(alrt$):gemsys(52
)
4150 choice=peek(gintout):cho=1
4160 return
4200 SHOWCURSOR:
4210 poke contrl,122:poke contrl+2,0:p
oke contrl+6,1:poke intin,0
4220 vdisys(1):return
4260 HIDECURSOR:
4270 poke contrl,123:poke contrl+2,0:p
oke contrl+6,0:vdisys(1):return
4310 IMAGECLEAR:
4320 alrt$="[3][|Erase cursor image?][
Yes|No]":gosub ALERT
4330 if choice=2 then return
4340 gosub BOXER:gosub BARS:gosub SHOW
CURSOR:return
4380 DLALERT:
4390 alrt$="[1][|Number was not in pro
per range][OK]:gosub ALERT:return
4430 DESTROY:
4440 color 1,0,1,1,1:poke contrl,11:po
ke contrl+2,2:poke contrl+6,0
4450 poke contrl+20,1:poke ptsin,266:p
oke ptsin+2,22:poke ptsin+4,571
4460 poke ptsin+6,120:vdisys(1):return
4490 BOXER:
4500 fc=3:gosub BC
4510 poke contrl,11:poke contrl+2,2:po
ke contrl+6,0
4520 poke contrl+10,1:poke ptsin,+1:po
ke ptsin+2,22
4530 poke ptsin+4,(16*15)+18:poke ptsi
n+6,(8*19)
4540 vdisys(1):gosub BOXING:return
4580 GEMSTUFF:
4590 a#=gb
4600 control=peek(a#)
4610 global=peek(a#+4)
4620 gintin=peek(a#+8)
4630 gintout=peek(a#+12)
4640 addrin=peek(a#+16):return
4680 MOUSE:
4690 poke contrl,124:poke contrl+2,0:p
oke contrl+6,0:vdisys(1)
```

```
4700 msb=peek(intout):msx=peek(ptsout)
:msy=peek(ptsout+2):return
4740 BARS:
4750 color 1,1,1,1,1:qt=0
4760 lt=15:for i=1 to 16:linef lt,0,lt
,128:lt=lt+16:next i
4770 for i=0 to 16:linef 0,qt,254,qt:q
t=qt+8:next i:return
4810 POLYLINE:
4820 poke contrl,6:poke contrl+2,5:pok
e contrl+6,0
4830 poke ptsin,plx:poke ptsin+2,ply:p
oke ptsin+4,plx1
4840 poke ptsin+6,ply:poke ptsin+8,plx
1:poke ptsin+10,ply1
4850 poke ptsin+12,plx:poke ptsin+14,p
ly1:poke ptsin+16,plx
4860 poke ptsin+18,ply:vdisys(1):retur
n
4900 PLSETUP:
4910 poke contrl,17:poke contrl+2,0:po
ke contrl+6,1:poke intin,0
4920 vdisys(1):return
4960 BOK:
4970 gosub BLK
4980 dx=(2*bx)+15:dy=by+161:gosub DOT1
:dx=bx+60:gosub DOT2
4990 return
5030 BOXING:
5040 color 3,3,3,1,1
5050 poke contrl,11:poke contrl+2,2:po
ke contrl+6,0
5060 poke contrl+10,1:poke ptsin,15:po
ke ptsin+2,161
5070 poke ptsin+4,120:poke ptsin+6,184
5080 vdisys(1):return
5120 PLOTTER:
5130 by=(ii-1)-gab:bx=(i):gosub BLK
5140 dx=(2*bx)+15:dy=by+161:gosub DOT1
5150 dx=bx+60:gosub DOT2
5160 cursor(ii)=cursor(ii)-(2^ct):retu
rn
5200 WELCOME:
5210 color 1,1,1,1,1:poke contrl,11:po
ke contrl+2,2:poke contrl+6,0
5220 poke contrl+10,1:poke ptsin,1:pok
e ptsin+2,22:poke ptsin+4,609
5230 poke ptsin+6,189:vdisys(1)
5240 mode=2:gosub WRITEMODE
5250 color 1,2,1,2,2:pcircle 300,90,20
0
5260 pcircle 0,0,170
5270 pcircle 610,0,200
5280 for i=1 to 20
5300 xfactor=int(rnd(1)*630)
5310 yfactor=int(rnd(1)*180)
5320 radi=int(rnd(1)*150)
5330 color 1,2,1,1,int(rnd(1)*5)
5340 pcircle xfactor,yfactor,radi
5350 next i:color 0,1,1,1,1
5360 theight=42:gosub TEXTHEIGHT:tex=4
:gosub TEXT
5370 textx=50:texty=60:a$="M O U S E
M A K E R  ver. 3.5":gosub TEXTPRINT
5380 theight=11:gosub TEXTHEIGHT:tex=1
:gosub TEXT
5390 textx=120:texty=80:a$=" By  Savee
n V. Reddy ":gosub TEXTPRINT
5400 theight=9:textx=160:texty=100:a$=
"Please wait for program to begin"
5410 gosub TEXTHEIGHT:gosub TEXTPRINT
5420 mode=1:gosub WRITEMODE:gosub fixt
ex
5430 for i=1 to 2100:next i
5440 return
5480 GETFILE:
5490 open "I",#1,"\cursor\names.fil":c
lose #1
```

```
5500 open "I",#1,"\cursor\names.fil"
5510 for i=1 to 50
5520 input #1,quest$
5530 namesfile$(i)=quest$
5540 next i
5550 close #1
5560 return
5600 PUTFILE:
5610 open "O",#1,"\cursor\names.fil":c
lose #1
5620 open "O",#1,"\cursor\names.fil"
5630 for i=1 to 50
5640 print #1,namesfile$(i)
5660 next i
5670 close #1
5680 return
5720 DISKFUNCTIONS:
5750 alrt$="[1][|Select a function][Di
r|Rename|Cancel]":gosub ALERT
5760 if choice=3 then return
5770 if choice=1 then goto DIRECTORY
5780 if choice=2 then goto RENAME
5790 return
5810 DIRECTORY:
5820 'osub fixtex
5830 for i=1 to 50 step 10
5840 for ii=0 to 9
5850 gotoxy 32,ii:print (i+ii);"-";nam
esfile$(i+ii)
5860 next ii
5870 gotoxy 32,10:? " Press any key to
 continue "
5880 woo=inp(2):gotoxy 32,10:? "
                                "

5890 gosub destroy
5900 next i:return
5920 RENAME:
5930 gotoxy 36,1:? "RENAME FILE"
5940 gotoxy 32,2:? "Type the number of
 the file"
5950 gotoxy 32,3:? "you wish to rename
 (1-50)"
5955 poke systab+24,onn
5960 gotoxy 35,5:input datfil$:df=val(
datfil$)
5970 if df=0 then gosub DESTROY:return
5980 if df<1 or df>50 then gosub DLALE
RT:goto 5960
5990 gosub DESTROY
6000 gotoxy 32,0:? "The current for fi
le #";df
6010 gotoxy 32,1:? namesfile$(df)
6020 gotoxy 32,3:? "Replace with"
6030 gotoxy 32,4:input quest$:poke sys
tab+24,off
6040 alrt$="[1][|Replace "+namesfile$(
df)+"|with "+quest$+"][Yes|No]
6050 gosub ALERT
6060 if choice=2 then gosub DESTROY:re
turn
6070 gosub DESTROY:namesfile$(df)=ques
t$:gosub PUTFILE:return
6080 CHANGECOLORS:
6090 for i=1 to 4:cindex=i-1
6100 gosub INQCOLOR
6110 oldc(i,1)=red
6120 oldc(i,2)=green
6130 oldc(i,3)=blue
6140 next i
6160 restore 6170
6170 data 7,7,7
6180 data 0,0,0
6190 data 7,0,0
6200 data 0,6,0
6210 for i=1 to 4:cindex=i-1
6220 read newc1,newc2,newc3
6230 red=int(newc1*(1000/7))
6240 green=int(newc2*(1000/7))
6250 blue=int(newc3*(1000/7))
6260 gosub SETCOLOR
6270 next i:return
6280 SETCOLOR:
6290 poke contrl,14
6300 poke contrl+2,0
6310 poke contrl+6,4
6320 poke intin,cindex
6330 poke intin+2,red
6340 poke intin+4,green
6350 poke intin+6,blue
6360 vdisys(1):return
6370 INQCOLOR:
6380 poke contrl,26
6390 poke contrl+2,0
6400 poke contrl+6,2
6410 poke intin,cindex
6420 poke intin+2,0
6430 vdisys(1)
6440 red=peek(intout+2)
6450 green=peek(intout+4)
6460 blue=peek(intout+6)
6470 return
```

●

ST CHECKSUM DATA.

(see page 38)

```
 10 data  901, 918, 419, 677, 247, 159
, 131, 969, 693, 814, 5928
 160 data  345, 720, 290, 96, 918, 340
, 678, 570, 729, 474, 5160
 320 data  480, 836, 843, 963, 75, 201
, 358, 906, 318, 326, 5306
 450 data  305, 854, 875, 808, 415, 79
1, 49, 76, 694, 421, 5288
 580 data  910, 99, 978, 704, 386, 268
, 892, 89, 996, 722, 6044
 680 data  407, 286, 886, 345, 762, 57
1, 707, 969, 555, 841, 6329
 840 data  243, 739, 822, 437, 200, 86
0, 420, 563, 855, 125, 5264
 1030 data  868, 614, 234, 125, 11, 42
2, 39, 939, 708, 611, 4571
 1190 data  156, 424, 390, 181, 212, 4
47, 649, 33, 248, 208, 2948
 1320 data  474, 732, 135, 977, 11, 80
1, 529, 641, 996, 733, 6029
 1440 data  387, 920, 9, 915, 530, 12,
 321, 278, 654, 957, 4983
 1540 data  397, 344, 723, 351, 917, 3
17, 791, 778, 882, 429, 5929
 1640 data  887, 603, 262, 210, 207, 4
41, 65, 698, 989, 845, 5207
 1770 data  948, 166, 722, 431, 725, 5
84, 953, 956, 983, 851, 7319
 1900 data  195, 562, 351, 158, 308, 2
7, 771, 471, 132, 281, 3256
 2060 data  0, 262, 479, 653, 857, 58,
 831, 864, 682, 244, 4930
 2220 data  831, 11, 393, 320, 291, 71
4, 903, 390, 621, 290, 4764
 2380 data  316, 786, 163, 581, 694, 8
46, 881, 828, 409, 273, 5777
 2500 data  60, 732, 65, 339, 726, 336
, 784, 610, 106, 635, 4393
 2630 data  70, 72, 44, 60, 78, 81, 65
, 67, 63, 73, 673
 2730 data  69, 848, 65, 569, 277, 536
, 873, 319, 885, 499, 4940
 2950 data  790, 562, 959, 588, 12, 74
6, 253, 646, 116, 119, 4791
 3080 data  930, 643, 564, 987, 87, 32
6, 520, 351, 690, 788, 5886
 3210 data  423, 547, 689, 590, 12, 73
2, 698, 168, 325, 82, 4266
 3320 data  836, 780, 106, 572, 24, 71
, 560, 326, 365, 658, 4298
```

// Mouse Maker *continued*

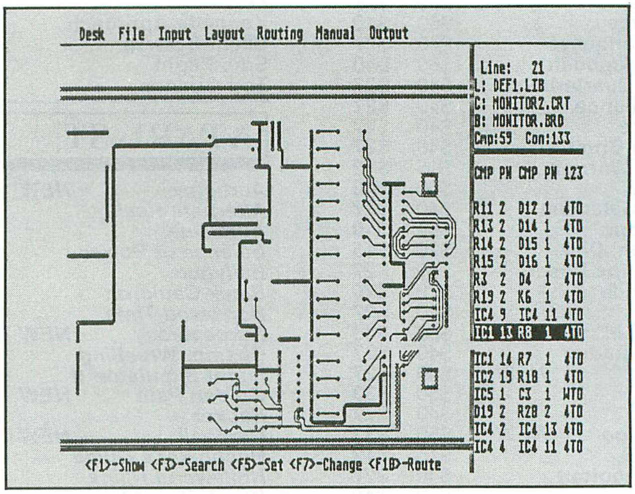
```
 3470 data  860, 507, 77, 357, 775, 4,
 458, 700, 143, 146, 4027
 3590 data  815, 601, 844, 28, 532, 81
5, 436, 448, 345, 365, 5229
 3700 data  641, 417, 435, 440, 344, 8
40, 919, 301, 593, 335, 5265
 3860 data  988, 270, 587, 286, 263, 8
55, 104, 749, 138, 518, 4758
 3990 data  312, 662, 809, 384, 428, 3
39, 78, 613, 524, 437, 4586
 4150 data  268, 836, 117, 847, 483, 6
3, 707, 937, 682, 569, 5509
 4340 data  459, 580, 225, 648, 298, 8
25, 949, 404, 165, 440, 4993
 4520 data  411, 561, 639, 841, 33, 63
6, 556, 592, 942, 695, 5906
 4680 data  399, 29, 957, 154, 777, 24
3, 429, 829, 474, 315, 4606
 4840 data  802, 738, 165, 684, 267, 7
36, 28, 756, 398, 479, 5053
 5030 data  481, 140, 431, 525, 761, 7
17, 621, 249, 685, 939, 5549
 5160 data  329, 590, 294, 482, 840, 5
76, 238, 231, 382, 45, 4007
 5300 data  319, 326, 829, 551, 413, 1
27, 737, 263, 720, 53, 4338
 5400 data  431, 147, 492, 133, 461, 6
00, 416, 323, 56, 407, 3466
 5530 data  193, 356, 416, 466, 639, 4
20, 334, 61, 274, 361, 3520
 5670 data  421, 471, 574, 289, 500, 6
03, 132, 475, 998, 241, 4704
 5830 data  116, 37, 550, 440, 527, 75
6, 343, 371, 457, 951, 4548
 5940 data  510, 417, 965, 10, 152, 28
1, 346, 692, 74, 754, 4201
 6030 data  206, 113, 250, 959, 367, 3
36, 268, 742, 574, 127, 3942
 6130 data  788, 666, 472, 811, 770, 7
92, 777, 266, 495, 852, 6689
 6240 data  221, 991, 752, 63, 849, 91
8, 38, 55, 756, 317, 4960
 6340 data  874, 539, 494, 850, 925, 4
7, 51, 758, 784, 103, 5425
 6440 data  779, 53, 32, 850, 1714
```

●

Some program listings reproduced in **ANALOG Computing** may contain "strange" characters not shown on the keyboards of earlier Atari models. These are special characters which use the CTRL, ESC and "ATARI LOGO" (inverse) keys. Shown below is a list of these characters and the keystrokes used to get them.

```
♥ --- CTRL ,
├ --- CTRL A
▕ --- CTRL B
┘ --- CTRL C
┤ --- CTRL D
┐ --- CTRL E
╱ --- CTRL F
╲ --- CTRL G
◢ --- CTRL H
▗ --- CTRL I
◣ --- CTRL J
▝ --- CTRL K
▘ --- CTRL L
▔ --- CTRL M
▁ --- CTRL N
▖ --- CTRL O
♣ --- CTRL P
┌ --- CTRL Q
─ --- CTRL R
┼ --- CTRL S
● --- CTRL T
▄ --- CTRL U
▎ --- CTRL V
┬ --- CTRL W
┴ --- CTRL X
▌ --- CTRL 'Y
└ --- CTRL Z
␛ --- ESC ESC
↑ --- ESC CTRL UP-ARROW
↓ --- ESC CTRL DOWN-ARROW
← --- ESC CTRL LEFT-ARROW
→ --- ESC CTRL RIGHT-ARROW
♦ --- CTRL .
♠ --- CTRL ;
↰ --- ESC SHIFT CLEAR
◀ --- ESC BACK S
▶ --- ESC TAB
◙ --- INVERSE CTRL ,
┫ --- INVERSE CTRL A
▐ --- INVERSE CTRL B
┛ --- INVERSE CTRL C
┣ --- INVERSE CTRL D
┓ --- INVERSE CTRL E
◤ --- INVERSE CTRL F
◥ --- INVERSE CTRL G
◤ --- INVERSE CTRL H
▛ --- INVERSE CTRL I
◥ --- INVERSE CTRL J
▙ --- INVERSE CTRL K
▟ --- INVERSE CTRL L
▇ --- INVERSE CTRL M
▀ --- INVERSE CTRL N
▜ --- INVERSE CTRL O
▣ --- INVERSE CTRL P
▛ --- INVERSE CTRL Q
▬ --- INVERSE CTRL R
▦ --- INVERSE CTRL S
◙ --- INVERSE CTRL T
▀ --- INVERSE CTRL U
▐ --- INVERSE CTRL V
▦ --- INVERSE CTRL W
▦ --- INVERSE CTRL X
▐ --- INVERSE CTRL Y
▙ --- INVERSE CTRL Z
⇧ --- ESC DELETE
⇩ --- ESC INSERT
⇦ --- ESC CTRL TAB (CLR)
⇨ --- ESC SHIFT TAB (SET)
█ --- INVERSE SPACE
▇ --- INVERSE _
◇ --- INVERSE CTRL .
♤ --- INVERSE CTRL ;
‖ --- INVERSE |
⍓ --- ESC CTRL 2
◁ --- ESC CTRL BACK S
▷ --- ESC CTRL INSERT
```

PC Board Designer


Desk File Input Layout Routing Manual Output
<F1>-Show <F3>-Search <F5>-Set <F7>-Change <F10>-Route

```
Line:   21
L: DEF1.LIB
C: MONITOR2.CRT
B: MONITOR.BRD
Cmp:59  Con:133

CMP PN CMP PN 123

R11 2  D12 1  4TO
R13 2  D14 1  4TO
R14 2  D15 1  4TO
R15 2  D16 1  4TO
R3  2  D4  1  4TO
R19 2  K6  1  4TO
IC4 9  IC4 11 4TO
IC1 13 R8  1  4TO
IC1 14 R7  1  4TO
IC2 19 R18 1  4TO
IC5 1  C3  1  WTO
D19 2  R28 2  4TO
IC4 2  IC4 13 4TO
IC4 4  IC4 11 4TO
```

ABACUS SOFTWARE
P.O. Box 7219 Dept. 1B
Grand Rapids, MI 49510
(516) 241-5510
High or medium resolution $195.00

by Matthew J.W. Ratcliff

The **PC Board Designer** is a software package that will assist you in the creation of printed circuit boards for your electronic hardware designs. This program will help you go from a schematic to a working prototype board with minimal effort, making tedious hand drawing and taping techniques things of the past.

My complaints about the **PC Board Designer** in its previous release were primarily the cost and the limited size of the circuit boards one could design. The price has been cut in half, and the newest version of the software (a free upgrade to all owners of the earlier version) supports 6.3-by-4.5-inch boards. This is only a minor improvement over the previous 6.3-by-3.9-inch size. The board is limited by the size of the display, since it is not a scrollable GEM window. If you can live with the small size of the boards, you'll find the **PC Board Designer** very powerful and easy to use.

Among the stated features of the **PC Board Designer** is that it can handle up to 250 components, with up to 1100 connections for a single design— although you would probably never fit that much into a 6-by-4-inch space. It handles single- or double-sided PC boards, and can automatically route connections to your layout. No matter how good you are at designing boards, a great deal of "manual" routing will be required as well, for anything but the simplest boards. You'll find it quite easy to lay out your components on the board, perform a test route, adjust positioning, and perform another auto route.

The **PC Board Designer** makes it very easy to modify your component layout, allowing you to pick up and "drag" a component anywhere on the board, clicking to rotate by 90-degree increments if necessary. As you drag a component around the board, its connections to other components are displayed continuously, with "rubber-band lines" to help you make the best placement for autorouting the signals. When the optimum layout is defined (a minimum of incomplete connections is achieved), then the remainder of the connections are completed in the manual mode. This is as simple as pointing and clicking with the mouse.

You may select routing at 45- or 90-degree angles. Two different trace widths are available. The wider will generally be used for power supply and ground lines. When you're ready to produce a prototype board, your completed board can be sent to an Epson FX80 or compatible printer, double sized. It can be reduced by 50 percent and used to create a transparent master for photo etching.

Note that the Gemini SG10 or earlier printers are not supported. The graphics output software uses Epson's programmable line feed, which is in 216ths of an inch. The Gemini's programmed in 144ths of an inch. Abacus is developing drivers for other printers, and pen plotters made by Hewlett-Packard and Epson. Support for NEC and Toshiba printers is currently available also, in the form of stand-alone load and print utilities.

In addition to outputting the printed circuit board, you can print a final layout of the pinholes and silk screen masks for the board. You can also output a list of components and the wire net (connection list).

In designing a board, you'll begin with a completed schematic. You may need to add components to the definition library, if they're not there. A fairly complete library comes on the program disk. There's a separate component definition editor program for adding to, changing, or creating a library.

Next, you'll need to enter a component and connection list. Since there's no separate window for these functions, you're limited by the width of the work area reserved to the right of the display. Component names in your design can only be three characters long (e.g., *IC1*, *R8K*, etc.), and their names in the library can only be six.

While entering information into these lists, you can call up a component list in the circuit board display area. I find it very annoying that components are not sorted alphabetically, however, making them difficult to find. And, with a limitation of six characters, their names can't be very descriptive.

(continued on page 74)

TEVEX Computer Software 1-800-554-1162

SSI

		LIST PRICE	OUR PRICE
Battalion Commander		$40	$27
Battle of Antietam		$50	$33
Battlecruiser	*NEW*	$60	$40
Broadsides		$40	$27
Carrier Force		$60	$40
Colonial Conquest		$40	$27
Computer Ambush		$60	$40
Computer Quarterback		$40	$27
Cosmic Balance		$40	$27
Field of Fire		$40	$27
50 Mission Crush		$40	$27
Gemstone Warrior		$15	$12
Gettysburg		$60	$40
Imperium Galactum		$40	$27
Kampfgruppe		$60	$40
Kamp. Scen. Disk		$20	$15
Knights of the Desert		$40	$27
Mech Brigade		$60	$40
Nam		$40	$27
Operation Market Garden		$50	$33
Panzer Grenadier		$40	$27
Phantasie	*NEW*	$40	$27
Questron		$50	$33
Rails West		$40	$27
Rebel Charge	*NEW*	$50	$33
Reforger 88		$15	$12
Six Gun Shootout		$40	$27
U.S.A.A.F.		$60	$40
War in Russia		$80	$53
Wargame Constr Set		$30	$20
Warship	*NEW*	$60	$40
Wizard's Crown		$40	$27

BRØDERBUND

	LIST PRICE	OUR PRICE
Champ. Lode Runner	$35	$23
Lode Runner	$35	$23
Lode Runner's Rescue	$30	$20
Print Shop	$45	$30
Print Shop Companion	$35	$23
PS Grap. Lib. 1,2,3	$25	$17

EPYX

		LIST PRICE	OUR PRICE
Summer Games		$20	$15
Temple Apshai Trilogy		$20	$15
World Karate Champ.		$30	$20
EPYX Joystick	*NEW*	$20	$15

DKG

	LIST PRICE	OUR PRICE
Clash of Wills	$40	$30
Great War 1914	$40	$30
Metz/Cobra	$40	$30
Objective Kursk	$40	$30
Surrender at Stalingrad	$40	$30

MICROPROSE

	LIST PRICE	OUR PRICE
Conflict in Vietnam	$40	$27
Crusade in Europe	$40	$27
Decision in the Desert	$40	$27
F-15	$35	$23
Kennedy Approach	$25	$17
Silent Service	$35	$23
Solo Flight	$25	$17
Top Gunner	$25	$17

ATARI ST

		LIST PRICE	OUR PRICE
Acrticfox	*NEW*	$40	$27
Alternate Reality		$50	$33
Auto Duel		$50	$33
Balance of Power		$50	$33
Ballyhoo		$40	$27
Black Cauldron		$40	$27
Borrowed Time		$50	$33
Burueacracy	*NEW*	$40	$27
Champ. Wrestling		$40	$27
Flight Simulator II		$50	$33
Golden Path	*NEW*	$45	$30
Hacker II		$40	$27
Hardball	*NEW*	$40	$27
Hitchhiker's Guide		$30	$20
Hollywood Hijinx		$40	$27
Kings Quest II		$50	$33
Kings Quest III		$50	$33
Leader Board		$45	$30
L.B. Tournament Disk		$20	$15
Leather Goddesses		$40	$27
Mean 18		$45	$30
Mercenary		$40	$27
Mercenary 2nd City		$20	$15
Moonmist		$40	$27
Ogre		$40	$27
Pawn		$45	$30
Phantasie		$40	$27
Phantasie II		$40	$27
Roadwar 2000	*NEW*	$40	$27
S.D.I.		$50	$33
Shanghai	*NEW*	$40	$27
Silent Service		$40	$27
Skyfox		$45	$30
Space Quest		$50	$33
Star Fleet I		$55	$37
Starglider		$45	$30
Sundog		$40	$27
Tass Times		$40	$27
Tenth Frame	*NEW*	$40	$27
2 on 2 Basketball	*NEW*	$40	$27
Ultima II		$60	$40
Ultima III		$60	$40
Universe II		$70	$46
Winter Games		$40	$27
World Games		$40	$27
Zork Trilogy	*NEW*	$70	$47

INFOCOM

		LIST PRICE	OUR PRICE
Ballyhoo		$40	$27
Burueacracy	*NEW*	$40	$27
Fooblitzky		$40	$27
Hitchhikers Guide		$25	$17
Hollywood Hijinx	*NEW*	$35	$23
Leather Goddesses		$35	$23
Moonmist		$35	$23
Zork Trilogy		$60	$40
Invisiclues		$8	$6

ELEC. ARTS

		LIST PRICE	OUR PRICE
Age of Adventure		$15	$12
Archon or Adept		$15	$12
Autoduel	*NEW*	$50	$33
Chessmaster 2000		$40	$27
Chickamauga		$35	$23
Lords of Conquest		$33	$22
Mail Order Monsters		$15	$12
Movie Maker		$15	$12
Music Construction Set		$15	$12
M.U.L.E.		$15	$12
Ogre	*NEW*	$40	$27
One on One		$15	$12
Racing Destr. Set		$15	$12
Rommel	*NEW*	$40	$27
Star Fleet I		$50	$33
Touchdown Football		$15	$12
Ultima III-Origin Sys.		$50	$33
Ultima IV-Origin Sys.		$60	$40

ETC.

		LIST PRICE	OUR PRICE
Alternate Reality		$40	$27
Fight Night		$30	$20
Flight Simulator II		$50	$33
FS II Scen Disks (each)		$20	$15
Guderian	*NEW*	$30	$20
Hacker		$15	$12
Hardball		$30	$20
Leader Board		$40	$27
L.B. Tournament Disk		$20	$15
Mercenary		$30	$20
Mercenary-2nd City		$15	$12
Micro League Baseball		$40	$27
MLB 1986 Team Disk		$20	$15
Music Studio		$35	$23
Never Ending Story		$30	$20
Pawn	*NEW*	$45	$30
Spy vs Spy I & II		$15	$12
Triple Pack		$20	$15
221-B Baker Street		$30	$20
221-B Case Library		$15	$12
Universe		$90	$70

Tevex Blank Disks

Box of 10, 5 1/4" DSDD only $12

Same Day Shipping

We ship every order the same day it's placed. Just call before 3:30 and we'll ship your order via UPS. U.S. Mail service is also available.

COMING SOON

FOR THE 8-BIT SERIES

Shiloh - SSI
Dungeon - Datasoft

FOR THE ST

Bard's Tale - Elec Arts
Breach - Omnitrend
Defender of the Crown
Gunship - Microprose
Rings of Zilfin - SSI
Shard of Spring - SSI
Wizard's Crown - SSI

CALL TOLL-FREE 1-800-554-1162

Write for FREE CATALOG

Open 9-8 Mon. - Fri. 10-5 Sat.

Georgia residents call 404-441-3045. ORDERING AND TERMS: C.O.D. orders welcome. When ordering by phone use VISA or MASTERCARD. When ordering by mail send money order. Include phone number. SHIPPING: Add $3.00 for shipping and handling charge. Georgia residents add 4% sales tax. Shipping for Canadian orders is $4.00 or 5% of order. APO & FPO orders add $3.00 or 5% of order. Shipping for all other foriegn orders is $10.00 or 15% of order. .All prices subject to change.

VISA

MASTERCARD

TEVEX
4205 First Avenue, Suite 100
Tucker, GA 30084
404-934-5059

CIRCLE #111 ON READER SERVICE CARD

M/L Editor

For use in machine language entry

by Clayton Walnum

M/L Editor provides an easy method to enter our machine language listings. It won't allow you to skip lines or enter bad data. For convenience, you may enter listings in multiple sittings. When you're through typing a listing with **M/L Editor**, you'll have a complete, runnable object file on your disk.

There is one hitch: it's for disk users only. My apologies to those with cassette systems.

Listing 1 is **M/L Editor**'s BASIC listing. Type it in and, when it's free of typos, save a copy to disk, then run it.

On a first run, you'll be asked if you're starting a new listing or continuing from a previously saved point. Press S to start, or C to continue.

You'll then be asked for a filename. If you're starting a new listing, type in the filename you want to save the program under, then press RETURN. If there's already a file by that name on the disk, you'll be asked if you wish to delete it. Press Y to delete the file, or N to enter a new filename.

If you're continuing a file, type in the name you gave the file when you started it. If the program can't find the file, you'll get an error message and be prompted for another filename. Otherwise, **M/L Editor** will calculate where you left off, then go on to the data entry screen.

Each machine language program in **ANALOG Computing** is represented by a list of BASIC data statements. Every line contains 16 bytes, plus a checksum. Only the numbers following the word *DATA* need be considered.

M/L Editor will display, at the top of the screen, the number of the line you're currently working on. As you go through the line, you'll be prompted for each entry. Simply type the number and press RETURN. If you press RETURN without a number, the default is the last value entered.

This feature provides a quick way to type in lines with repetitions of the same number. As an added convenience, the editor will not respond to the letter keys (except Q, for "quit"). You must either enter a number or press RETURN.

When you finish a line, **M/L Editor** will compare the entries' checksum with the magazine's checksum. If they match, the screen will clear, and you may go on to the next line.

If the checksums *don't* match, you'll hear a buzzing sound. The screen will turn red, and the cursor will be placed back at the first byte of data. Compare the magazine listing byte by byte with your entries. If a number's correct, press RETURN.

If you find an error, make the correction. When all data's valid, the screen will return to grey, and you'll be allowed to begin the next line.

Make sure you leave your disk in the drive while typing. The data is saved continuously.

You may stop at any time (except when you have a red screen) by entering the letter Q for byte #1. The file will be closed, and the program will return you to BASIC. When you've completed a file, exit **M/L Editor** in the same way.

When you've finished typing a program, the file you've created will be ready to run. In most cases, it should be loaded from DOS via the L option. Some programs may have special loading instructions; be sure to check the program's article.

If you want the program to run automatically when you boot the disk, simply name the file AUTORUN.SYS (make sure you have DOS on the disk).

That's **M/L Editor**. Use it in good health.

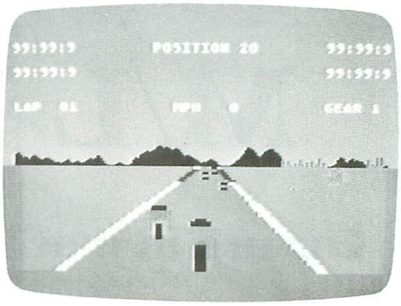
The two-letter checksum code preceding the line numbers here is *not* a part of the BASIC program. For further information, see the *BASIC Editor II*, in issue 47.

Listing 1.
BASIC listing.

```
AZ 10 DIM BF(16),N$(4),A$(1),B$(1),F$(15)
   ,F1$(15)
LF 11 DIM MOD$(4)
BN 20 LINE=1000:RETRN=155:BACKSP=126:CHKS
   UM=0:EDIT=0
GO 30 GOSUB 450:POSITION 10,6:? "Start or
    Continue? ";:GOSUB 500:? CHR$(A)
ZG 40 POSITION 10,8:? "FILENAME";:INPUT F
   $:POKE 752,1:? " "
FE 50 IF LEN(F$)<3 THEN POSITION 20,10:?
   "  ":GOTO 40
NF 60 IF F$(1,2)<>"D:" THEN F1$="D:":F1$(
   3)=F$:GOTO 80
KL 70 F1$=F$
TN 80 IF CHR$(A)="S" THEN 120
FD 90 TRAP 430:OPEN #2,4,0,F1$:TRAP 110
HQ 100 FOR X=1 TO 16:GET #2,A:NEXT X:LINE
   =LINE+10:GOTO 100
WM 110 CLOSE #2:OPEN #2,9,0,F1$:GOTO 170
VT 120 TRAP 160:OPEN #2,4,0,F1$:GOSUB 440
   :POSITION 10,10:? "FILE ALREADY EXISTS
   !!":POKE 752,0
ZU 130 POSITION 10,12:? "ERASE IT? ";:GOS
   UB 500:POKE 752,1:? CHR$(A)
VH 140 IF CHR$(A)="N" OR CHR$(A)="n" THEN
    CLOSE #2:GOTO 30
QG 150 IF CHR$(A)<>"Y" AND CHR$(A)<>"y" T
   HEN 130
BH 160 CLOSE #2:OPEN #2,8,0,F1$
IE 170 GOSUB 450:POSITION 10,1:? "NOW ON 
   LINE: ";LINE:CHKSUM=0
GH 180 L1=3:FOR X=1 TO 16:POSITION 13*(X<
   10)+12*(X>9),X+2:POKE 752,0:? "BYTE #"
   ;X;": ";:GOSUB 310
KH 190 IF EDIT AND L=0 THEN BYTE=BF(X):GO
   TO 210
FY 200 BYTE=VAL(N$)
OZ 201 MOD$=N$
BU 210 POSITION 22,X+2:? BYTE;"  "
YZ 220 BF(X)=BYTE:CHKSUM=CHKSUM+BYTE*X:IF
    CHKSUM>9999 THEN CHKSUM=CHKSUM-10000
MS 230 NEXT X:CHKSUM=CHKSUM+LINE:IF CHKSU
   M>9999 THEN CHKSUM=CHKSUM-10000
IG 240 POSITION 12,X+2:POKE 752,0:? "CHEC
   KSUM: ";:L1=4:GOSUB 310
EW 250 IF EDIT AND L=0 THEN 270
QM 260 C=VAL(N$)
SY 270 POSITION 22,X+2:? C;"   "
IL 280 IF C=CHKSUM THEN 300
DI 290 GOSUB 440:EDIT=1:CHKSUM=0:GOTO 180
LW 300 FOR X=1 TO 16:PUT #2,BF(X):NEXT X:
   LINE=LINE+10:EDIT=0:GOTO 170
FV 310 L=0
LG 320 GOSUB 500:IF A=ASC("Q") AND X=1 AN
   D  NOT EDIT THEN 420
PO 330 IF A<>RETRN AND A<>BACKSP AND (A<4
   8 OR A>57) THEN 320
DX 331 IF A=RETRN AND N$="" THEN N$=MOD$
TD 335 IF A=RETRN AND L=0 AND X>1 THEN 35
   0
JR 340 IF ((A=RETRN AND  NOT EDIT) OR A=B
   ACKSP) AND L=0 THEN 320
DW 350 IF A=RETRN THEN POKE 752,1:? " ":R
   ETURN
GG 360 IF A<>BACKSP THEN 400
SA 370 IF L>1 THEN N$=N$(1,L-1):GOTO 390
AS 380 N$=""
RE 390 ? CHR$(BACKSP);:L=L-1:GOTO 320
BB 400 L=L+1:IF L>L1 THEN A=RETRN:GOTO 35
   0
WX 410 N$(L)=CHR$(A):? CHR$(A);:GOTO 320
KN 420 GRAPHICS 0:END
YT 430 GOSUB 440:POSITION 10,10:? "NO SUC
   H FILE!":FOR X=1 TO 1000:NEXT X:CLOSE
   #2:GOTO 30
FD 440 POKE 710,48:SOUND 0,100,12,8:FOR X
   =1 TO 50:NEXT X:SOUND 0,0,0,0:RETURN
MY 450 GRAPHICS 23:POKE 16,112:POKE 53774
   ,112:POKE 559,0:POKE 710,4
XR 460 DL=PEEK(560)+256*PEEK(561)+4:POKE
   DL-1,70:POKE DL+2,6
HW 470 FOR X=3 TO 39 STEP 2:POKE DL+X,2:N
   EXT X:FOR X=4 TO 40 STEP 2:POKE DL+X,0
   :NEXT X
ZW 480 POKE DL+41,65:POKE DL+42,PEEK(560)
   :POKE DL+43,PEEK(561):POKE 87,0
AC 490 POSITION 2,0:? "analog ml editor":
   POKE 559,34:RETURN
WZ 500 OPEN #1,4,0,"K:":GET #1,A:CLOSE #1
   :RETURN
```

●

Speed King



MASTERTRONIC
7311B Grove Road
Frederick, MD 21701
48K Disk $9.99

by Matthew J.W. Ratcliff

Speed King is a smooth-scrolling motorcycle racing simulation. It pits you against nineteen other world-class motorcycle riders on one of ten famous racetracks, which you can select.

The screen photos on the box promise an exciting motorcycle version of the ever-popular **Pole Position** from Atari. It looks promising, but falls apart when you hit the track—frequently.

The program comes in both Atari and Commodore versions, on the same side of the same disk. The Atari version is autobooting, while Commodore owners just load and run. (I have no idea how they achieve such a neat trick; I didn't think the disk formats were at all compatible.)

You can change your racetrack selection at the title screen by pressing the OPTION key. The tracks are shown in a nicely done graphics window at the top left of the display. They're presented in an overhead view, showing your racing direction and all the treacherous curves you'll *never* make. You can hit the SPACE BAR to preview any of the tracks, or RETURN to practice them. The ESCAPE key allows you to set up a 2-, 4- or 6-lap race. When you're ready to crash and burn, pressing START, with a joystick in port 2, gets you into the game.

You begin in the middle of a racetrack, behind nineteen other pros, with a rear overhead view of your cycle and rider. You must slap your way through six—count 'em, six—gears on your way up to a top speed of 250 miles per hour. As the joystick is moved left or right, the rider leans into the turn. The lateral control of the cycle is poor to nil, making this game virtually impossible to enjoy.

If you make no attempts to control your machine, the program forces you to stay within the general vicinity of the track (like **Pole Position**), with *no loss of speed* (if you keep jamming on the accelerator fire button), contrary to the game's description. If you ride out on the lawn long enough, the game decides you don't know what you're doing, and you crash. (**Pole Position** has road signs to keep you in line, or to crash against; also, when you hit the track's edge, you slow down and, realistically, gain increased traction—but not in this imitation.) At different difficulty levels in the game, the *only* apparent change is in the amount of time the program will let you ride your machine on the lawn before you crash. It's fairly easy to get the hang of taking the curves, even at high rates of speed, but passing other riders seems virtually impossible.

The optimum play strategy for **Speed King** appears to be to match the other riders' speed. As you slowly catch up to the rear of the pack, tapping the fire button speed control and downshifting helps slow your cycle. Approaching the pack at more than a mile or two per hour faster than the rest is always deadly, since lateral movement is so difficult. Apparently, all the other pros are fairly evenly matched, because it seems that all nineteen riders are stacked in a very tight pack when it comes time to pass them.

As you begin to pass, it becomes immediately apparent that this game is an exercise in futility. If you go too slowly, trying to achieve some "bite" on the track to move left or right, some clown mows you down from behind—and keeps on going (a really accurate simulation!) If you go too fast, you nail someone because you can't steer out of his way (you crash and burn; he keeps on going). By closely duplicating the others' speed, you can gradually weave in and out of the pack.

The graphics and sound effects are pretty well done. The scrolling and 3D racetrack effects are almost as well done as those of Atari's **Pole Position**. This game could be as addictive too, if it had more playability.

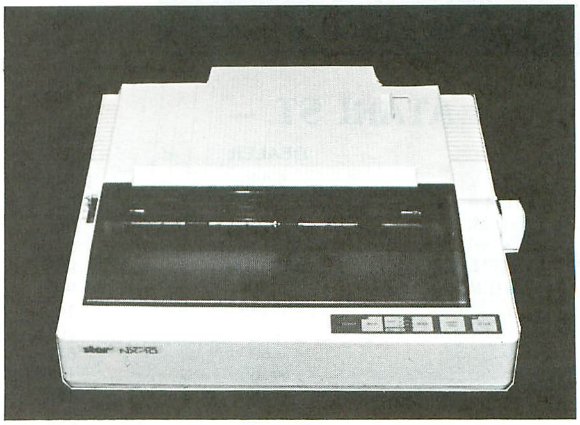
Contrary to Steve Panak (**ANALOG Computing** 48's **Panak strikes!**), I liked **Kikstart**. On the 3D smooth-scrolling screen, you could ride your cycle up and down, and around objects—and even jump ramps, in an effort to accumulate forty items to complete a treasure and participate in a final drag race. This game had good joystick response and control. Although it was an old game in a new format, I found it rather fun to play and worth the affordable price of $9.99.

That was why I was hopeful **Speed King** would be a much faster, more action-oriented game. But, in coming out with a fast action game, Mastertronic has lost all *control* in this game. You just can't do much of anything with it. The company had better come up with some more documentation and original approaches (have you ever noticed how many different *motorcycle* games they put out?) if they would like to stay in the software business. ■

Matthew Ratcliff is an Electrical Engineer in St. Louis, Missouri. When not using his spare time to write articles, he's president of ACE St. Louis and a remote SYSOP on the Gateway City BBS, (314) 647-3290.

NX-10 Printer

STAR MICRONICS, INC.
200 Park Avenue, Suite 3510
New York, NY 10166
(212) 986-6770
$349.00



by Greg Knauss

It's been said thousands of times. . . At the risk of being redundant, I'll say it again: word processing is one of the most powerful and useful functions available to Atari owners. And this is all fine and good —until you realize you need to carry the silly computer around to show anybody what you've written. Enter printers.

There are literally dozens of printers available for Atari computers, one of which is the Star Micronics **NX-10**. From excellent quality to low price, to complete documentation, the **NX** is the best all-around value I've run into in a long time.

First, and probably most important, is print quality. This is where most inexpensive printers fall down—a low price usually guarantees very "computerlike" printing: letters made up of little dots. Though that's acceptable for program listings, friendly letters and personal printouts, it's almost never used in professional correspondence or business letters.

Although the **NX-10** does have a good draft mode—a slight improvement over its most famous predecessors, the Gemini 10X and the Epson 80—it solves the quality problem with a hot item among new printers: Near Letter Quality (NLQ). NLQ is an attempt by a dot-matrix printer to imitate a daisy wheel or typewriter. The **NX** NLQ mode surpasses any other I've seen at the price. It is excellent, completely suitable for all but the strictest requirements.

Though the NLQ mode is the highlight of Star's new printer, they have by no means left out other important features. Its printing capabilities are more than rounded out by: italics, pica, elite, subscript and superscript fonts; condensed, expanded, proportional, emphasized and bold printing; underlining; reverse linefeeds; two IBM, twelve international (including ancient Greek) and infinite downloadable character sets; single-through quad-density graphics; and a whole slew of other graphic and positioning commands. Graphics dumps are possible, and easily accomplished with any program written for Epson or Gemini printers. The **NX** leaves no stone unturned; it can be configured to *anybody's* preferences.

It's also easy to use. The **NX** is reported to be 100-percent Epson compatible, so it should have no problem working with any program written for Epson printers. The CTRL-J command that has caused so much trouble for Star owners in the past has been changed to match Epson's 1/216th-inch linefeeds. Also, when listing programs, the **NX** will convert all inverse characters to italics, making the printout much easier to read.

Along with NLQ, the printer also has another feature new to inexpensive printers: push-paper feeding. By pushing the paper past the printhead, it prevents the waste of a whole sheet of paper in starting at the top of the page. This is the sort of thing you don't appreciate unless you have never had it; and it is appreciated. As would be expected, the **NX** accepts both tractor- and friction-fed paper.

While all that's mentioned above makes the **NX** a great printer, Star went even further. They made features that are optional or not available on most printers standard on the **NX**. A 5K buffer is built in. There's a dip switch, to change from a regular zero to a slash zero. The **NX-10** has an extensive control panel, and an extra fuse is even included! There are dozens of pleasant surprises waiting in the **NX** box.

The manual is excellent. It speaks to the beginner with topics ranging from where to put the printer, to how to print in BASIC, to how to create your own character sets. In one appendix, all the function codes are comprehensively detailed for quick reference. If you stop in at your dealer, ask to see the manual. Check out Appendix D to get an idea of this printer's capabilities.

While the manual is excellent overall, I have to mention that it doesn't have an index, and the table of contents only has page numbers for the main subject of each chapter, not the individual items covered. These are annoying oversights, to say the least.

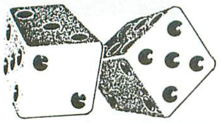
The **NX** does have a few other down points, too. On previous Star printers, the draft-quality character set was a copy of the Epson draft mode. The **NX** uses rounder letters for these printouts, to make them look better. While this is generally successful, some of the characters, especially the uppercase *A*, *G*, *Q* and the asterisk, look sort of funny, with *A* being the worst.

All in all, the feature-packed **NX-10** is an excellent value for almost anyone who needs a printer, either casually or professionally. Star has a winner here, a combination of quality and price that will be hard to beat.

Greg Knauss is a freshman at the University of California, San Diego. He misses home, and Kelly, terribly. He's involved in journalism and drama, as well as computers.

LOWEST PRICES ANYWHERE — BEST SERVICE — WE ARE YOUR ATARI CONNECTION

Lucky Seven



**30% OFF LIST
ON
ST SOFTWARE**

ATARI ST

At **7% Over Dealer** Cost
Plus **FREE SHIPPING**

FULL MANUFACTURER'S WARRANTIES
IMMEDIATE **REPLACEMENT** ON
ALL **DOA** DEFECTS

GUARANTEE

We Guarantee our Prices to be the **LOWEST** Anywhere
If You See any Item Advertised in this Issue for Less than our Price
We Will **MATCH** that Price - and Give you **FREE** Shipping

– ATARI ST –

	DEALER	YOU PAY
520ST	$ 330	$ 353
520ST MONO SYSTEM	600	642
520ST COLOR SYSTEM	750	802
1040ST	515	551
1040ST MONO SYSTEM	750	802
1040ST COLOR SYSTEM	900	963
SF354	135	145
SF314	188	201
SHD204	600	642
SM804	165	177
SM124	135	145
SC1224	285	305
SX212	75	80
PS300	425	455

(Computer System includes ST-CPU, Disk Drive and Monitor.)

CERTIFIED CHECK OR MONEY ORDER ONLY!
NO CREDIT CARDS OR COD AT THESE PRICES

MS. EM	$70
MS. EM PLUS 5¼" IBM DRIVE	275
MAC SAC PLUS	$120

CABLES

PRINTER	12
MODEM	12
DISK DRIVE - 6'	15
COMP MONITOR	20

DISKS

3.5 SS, DD (Box 10)	14
3.5 DS, DD (Box 10)	19

DISK HOLDERS

FLIPSORT MICRO (HOLDS 60, 3.5" DISKS)	9

JOYSTICKS

CS30	6
WICO MAX	9
WICO BOSS	13
WICO BAT	17
WICO 3-WAY	20
SLIKSTIK	6
KRAFT SWITCh	9
TAC2	11
TAC 3	13
TAC 5	14
500XJ	13
QUICKSHOT 2	9
SUNCOM ECONO	5

ACCESSORIES

REALTIZER COLOR DIGITIZER	$140
SOUND DIGITIZER NAVARONE	105
1 MEG ST UPGRADE (NO RAM)	55
CLOCK CARTRIDGE	35
INTERNAL CLOCK	
520	45
1040	55
256K MEM CHIP	4
SUR SUPP 6 OUT	20

AC STRIP 6 OUT	15
PRINTER STAND	10
MONITOR SWITCH (SELECT EITHER MONO OR COLOR MONITOR)	50
RGB MON ADAPTER (USE 3rd PARTY RGB MONITOR ON ST)	50
ATARI ST MOUSE	40
MOUSE MAT	8

COVERS

520	6
1040	6
354/314	6
520-3 PC	12
1040-2 PC	12
124/1224	6

MODEMS

HABA 1200	120
ATARI 300	40
ATARI 1200	80
AVATEX 1200	80
AVATEX 1200HC	120
HAYES 1200	370
HAYES 2400	550
CAPETR 1200 HC	115
COURIER 2400	379
DISK/MONITOR CLEANING KIT	14

PRINTERS

EPSON

LX86	$220
TRACTOR	30
FX86E	350
FX286E	490
LQ800	490
TRACTOR	45
CS FEEDER	95
LQ1000	690
TRACTOR	55
EX800	425
COLOR OPT	70
EX1000	560
COLOR OPT	70

PANASONIC

1080I	210
1091I	250
1092I	350

STAR MICRONICS

NP10	180
NX10	220
NX15	340

SEIKOSHA

SP1000A	180

OKIDATA

182P	195
192+P	350
193+P	530

PROWRITER

C310XP	490
C315XP	590

MONITORS

12" GREEN	80
13" COM COLOR	170
1702	190
1802C	210

ST SOFTWARE

DAC PAYROLL	$35
DAC ACCOUNTING	49
10TH FRAME	28
AEGIS ANIMATOR	56
ALT REAL: CITY	28
ARTIC FOX	32
ASSEM PRO	42
BAL OF POWER	35
BALLY HOO	28
ML BASEBALL	39
BRATACCAS	32
BRIDGE 4.0	21
CARDS	28
CHAMP WRESTLING	28
CHESS MASTER 2000	32
COMPUBRIDGE	21
COPY II ST	28
CORNERMAN	35
CRYSTAL CAVES	19
DATA MGR ST	55
DATA TRIEVE	35
DB MAN	105
DOS SHELL	28
EASY DRAW	55
EIGHT BALL	28
ENCHANTER	28
FLIGHT SIM II	35
FOREM BS	45
GATO	35
GFA BASIC	55
GFA COMPILER	55
GFL FOOTBALL	28
GRAPHIC ARTIST	140
HACKER 2	35
HARRIER STRIKE	35
HIGH ROLLER	35
HITCHHIKERS GUIDE	28
JOUST	19
KARATE KID 2	28
KINGSQUEST	35
KINGSQUEST 2	35
KINGSQUEST 3	35
LABELMASTER	28
LB GOLF	28
LEATHER GODDESS	28
MAJOR MOTION	28
MARK WILLIAMS C	126
MATCH POINT	28
MEAN 18 GOLF	32
MEAN 18 COURSE	13
MEGAFONT	28
MEGAMAX C	160
MERCENARY	28
MICRO C-SHELL	35
MICRO C-TOOL	$18
MICRO MAKE	22
MICRO MT C-SHELL	90
MICRO RTX	49
MINDSHADOW	35
MOONMIST	28
MUSIC BOX	35
MUSIC STUDIO	42
N-VISION	28
NEOCHROME	19
NINJA MISSION	14
OGRE	28
PAINT PRO	35
PAINTWORKS	50
PAWN	32
PCB DESIGNER	120
PERSONAL PASCAL	52
PHANTASIE	28
PHANTASIE 2	28
PLANETFALL	28
PRINTMASTER +	28
PUBL PARTNER	99
RAID	24
REGENT BASE	69
REGENT WORD 2	69
RENEGADE	14
ROGUE	28
RUBBER STAMP	28
SDI	35
SHANGHAI	35
SHUTTLE II	28
SILENT SERVICE	28
SKYFOX	32
SPACE QUEST	35
ST POOL	19
ST SHUFFLEBOARD	19
ST TALK	13
STAR FLEET 1	32
STAR RAIDERS	19
STARGLIDER	32
STRIP POKER	28
SUNDOG	28
SUPER CYCLE	28
SUPER HUEY	28
SWIFT CALC	55
TEXT PRO	35
TEMPLE TRILOGY	28
TIME BNDITS	28
TRIM BASE	64
TRINITY	28
TYPESET ELITE	35
ULTIMA 3	48
UNIVERSE 2	49
VIP LITE	75
VIP PRO (GEM)	140
WANDERER	28
WINTER GAMES	28
WORD WRITER ST	55
WORLD GAMES	28
WRITE 90	21
ZORK TRILOGY	45
ZOOMRACKS II	69

SIDE-LINE COMPUTER
86 RIDGEDALE AVENUE
CEDAR KNOLLS, NJ 07927

NO TELEPHONE ORDERS
CALL FOR PRICING OF OTHER ITEMS
(201) 455-7844

WE DO NOT CARRY CLOSEOUT OR DISC ITEMS – APO & FPO CHARGED ACTUAL SHIPPING
NJ RESIDENTS ADD 6% SALES TAX – FREE SHIPPING IN CONTINENTAL UNITED STATES

CIRCLE #112 ON READER SERVICE CARD

48K Disk

GAME

Life in the Fast Lane

The classic gets an update to keep you mesmerized.

by James Hague

Back in 1970, before the home computer industry had taken off, a mathematician by the name of John Horton Conway created a program known simply as **Life**. **Life** mathematically simulated the formation and destruction of colonies of "cells." The almost-limitless possibilities introduced by this rather simple program captured the imagination of practically everyone who "just wanted to try it once." Over the next few years, wherever a computer existed, **Life**—in one form or another—was sure to be found. **ANALOG Computing** published an ST version in issue 49.

Now it's the eighties. **Life** can be found floating around in various public domain libraries, as well as in most college and university computer labs. The concept behind the "game" is just as enticing as it was back in 1970, but incredibly slow BASIC programs with Xs representing cells just don't cut it anymore. Simply determining the outcome of a single colony of cells may take anywhere from several minutes to several hours, depending on the quality of the implementation.

Finally, **Life** has been given a complete overhaul, in an effort to bring it up to par for this decade. Herein, I present **Life in the Fast Lane**.

Typing it in.

Listing 1 is the BASIC data for **Life in the Fast Lane**. *Do not* type this as if it were a BASIC program. Instead, use Clayton Walnum's **M/L Editor** (see page 41) as a means to get all of those numbers safely locked away in a binary file on disk. If you name the resulting file AUTORUN.SYS and place it on a disk containing DOS.SYS (DUP.SYS isn't necessary), **Life in the Fast Lane** will automatically load and run when your computer's turned on. XL and XE owners should hold down the OPTION key during power-up, in order to disable BASIC. If you choose a different filename, use option L from the DOS menu to get the program up and running.

Listing 2 for this offering is included, for those who are interested in assembly language programming. You do not need to type it in.

Creating a colony.

Plug a joystick into port 1, and you're ready to begin.

First of all, note that the screen is divided into three distinct sections. The upper portion contains vital information about the cell colony. The center of the screen is where the actual cells are located. And at the very bottom is a message line which will relay important information to you from the computer.

The joystick controls the flashing square cursor located in the center of the board. Pushing the joystick in any of the eight standard directions will slide the cursor in the corresponding direction.

To place a cell on the board, press the trigger and, with a ding, a cell will appear. Press the trigger again and the cell will instantly vanish. You can hold down the trigger while moving around, in order to "draw" lines of cells. Cells can be placed on any of the 256 board squares, but the real fun is in deciding exactly *which* ones.

What you're actually doing is creating the first "generation" of your cell colony. When you're satisfied with the colony's architecture, a quick press of the START key will crank up the "Life" computer, which will determine the future of the cells.

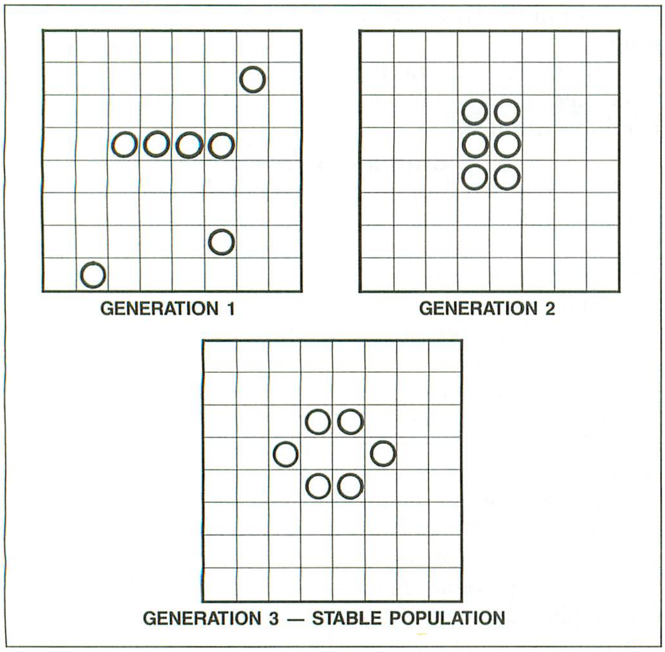
There are two other keys that are active during the editing process. The SELECT key will change the current mode as displayed on the far right of the second line from the top of the screen. The mode determines exactly how the cells will be regenerated, but don't worry about it for now. Pressing ESC (ESCape) instantly erases all cells from the board, providing a clean slate to work with.

Let there be *Life*.

Once START has been pressed, the real fun begins. This program is similar to Joel Gluck's **Bounce** (the original of which appeared in **Our Game**, issue 15), in that you're responsible for setting things up, but are nothing more than an innocent bystander once control is relinquished to the computer. It is this aspect—sitting back and watching your creation—that's responsible for the popularity of both the original **Life** and the more recent **Bounce**. (In its three incarnations, **Bounce** has become one of **ANALOG**'s most popular programs).

The cell colony is regenerated according to a few simple rules. First, a cell that is surrounded by fewer than two or more than three neighboring cells will be killed. Next, if an empty square is bordered by exactly three cells, a new cell will grow in the empty spot. Finally, if a cell is a neighbor of two or three other cells, that cell will survive. Figure 1 shows a small cell colony and how it would regenerate according to these rules. Note that each square on the board is bordered by eight others. These simple rules can transform your colony into a thriving populace—or, just as easily, crumble it into dust.

Figure 1. — Sample cell regeneration.



The cell colony will continue to regenerate until one of four things happen: (1) you press a console key (START, SELECT or OPTION) to turn off the Life computer and return to the editing screen; (2) the cell colony reaches a stable population—meaning that, according to the aforementioned rules, no cells will grow or die—in this case, you'll be returned to the editor; (3) the cell colony repeats—meaning that it continually grows and dies in a short pattern—press a console key to abort; and (4) the cell colony completely dies out.

Pressing the SPACE BAR while the Life computer is running will temporarily pause things until a *different* key is pressed.

Statistics and modes.

The information at the top of the screen supplies you with statistics—some vital, some frivolous—about the current colony. On the very top line is the number of the current generation. While editing the board, this will be *1*. While the Life computer is running, it's important to note that the generation number is incremented *after* changes are made to the current generation.

The second line contains more statistics. The first value is the current cell population. Obviously, when it reads *0*, the colony is no more. The maximum population is limited by the number of squares on the board, and is therefore *256*.

The next value is the number of cells that originally started out in the colony. This is used to determine just how much a certain colony has reproduced. For instance, if the starting population is 40 and the current population is 120, then there's been a threefold increase in the number of living cells. The next-to-last value is for the most cells that have been recorded so far, or, in other words, the highest population.

Finally, we come to a three-letter mnemonic, which represents the current mode the Life computer is in. This is the mode selected while editing the screen. I realized that some people may want to change modes midway through the life of a cell colony, so I added a special feature to allow this capability.

As you already know, pressing a console key will abort the Life computer and allow you to edit the board. The catch is as follows: the cell statistics are only reset if you make a change to the colony. This lets you shut down the Life computer, switch modes and restart, without affecting the colony in any way. The following modes have been provided:

RGN — ReGular Noisy. In this mode, cell growth/death is animated and is accompanied by pleasant-sounding chords. Each generation lasts for about one second.

RGQ — ReGular Quiet. This is the same as RGN, except that the sound is turned off.

SSN — Single Step Noisy. In this mode, after each generation you will be prompted to press the joystick trigger, in order to progress to the next generation. This allows you to watch exactly how the fate of each cell is determined. You can hold down the trigger to speed through successive generations.

SSQ — Single Step Quiet. This is the same as SSN, except that the sound is turned off.

FST — FaST. This mode really shows the speed and power of machine language. Both the animation and sound effects are bypassed, thus simplifying the Life computer's task immensely. This mode can speed through about twenty or so generations per second, so the screen really appears to be alive! There are two major uses for this mode: if you want to quickly find out exactly how many generations a colony will survive, or if you want to get the big picture of the birth and death of a colony.

Exploring *Life*.

It's easy to spend hours upon hours experimenting with

Life in the Fast Lane. There are, for all practical purposes, an infinite number of possible cell colonies. So here are a couple of ideas and observations I've made after a few hours experimentation, to get you started.

Once you get **Life in the Fast Lane** up and running, I know you'll want a quick demonstration of its power, so here's a simple one. Draw a horizontal row of cells across the screen about midway between the top and bottom, and turn on the Life computer. Within seconds, the population will have more than quadrupled, forming one large colony. This colony will spread outward, eventually breaking up into several subcolonies. Finally, around generation 33, six small, infinitely-repeating colonies will be formed. Press a console key to stop. Try this in each of the different modes, to gain a better understanding of how and when to use each one.

It won't take long to be able to identify certain small groups of cells that either repeat or are stable. A block of four cells is a stable configuration. A row or column of three cells will repeat forever and ever. There are many other such patterns, but you should be able to pick up on them after logging in a few hours with your Atari.

The ultimate goal of all **Life** enthusiasts is to develop a colony that survives infinitely, without getting stuck in a pattern. This feat is probably impossible, but that doesn't seem to have stopped people from trying. Colonies have been devised that can last for around 300 generations, but not much progress has been made beyond that point. If you ever get a colony going that surpasses the 300th generation or higher, I want to know about it!

It's amazing how realistically some colonies behave. They start as a small clump and move outward from this position, as if new territory is being explored. Sometimes a small subcolony will be started, in an effort to colonize the area more fully. More than once, I've had several small colonies created, only to be destroyed by a larger colony that "attacks" each one in sequence.

Many people claim that the behavior of certain cell colonies represents real-life beliefs or morals. I'm not going to agree or disagree with this idea, but it is a fact that a cell colony fashioned in the shape of a swastika, if correctly positioned, will die within four generations.

Go ahead and type in **Life in the Fast Lane**. Maybe you'll see pretty little pulsating creatures on a computer screen; maybe you'll see more. . .

James Hague attends North Texas State University and is a member of the Dallas Atari Computer Enthusiasts. He has been programming an Atari 800 in both BASIC and assembly language since 1982.

Listing 1.
***M/L Editor* data.**

```
1000 DATA 255,255,0,32,251,32,216,32,5
4,32,32,106,32,32,4,42,9134
1010 DATA 32,107,33,32,110,36,32,203,3
8,76,13,32,216,165,128,48,3532
1020 DATA 24,169,0,133,77,32,114,35,32
,86,35,165,128,208,10,32,2320
1030 DATA 16,36,230,135,165,135,141,19
9,2,76,98,228,32,101,228,32,6916
1040 DATA 86,32,169,0,133,12,169,32,13
3,13,162,255,134,128,134,133,7714
1050 DATA 232,134,132,160,22,162,32,16
9,7,76,92,228,169,3,141,15,4409
1060 DATA 210,169,0,141,8,210,141,1,21
0,141,3,210,141,5,210,96,6909
1070 DATA 162,0,189,0,224,157,0,8,189,
0,225,157,0,9,189,0,2946
1080 DATA 226,157,0,10,189,0,227,157,0
,11,232,208,229,162,63,189,9822
1090 DATA 152,32,157,16,8,202,16,247,1
69,8,141,244,2,96,0,0,2631
1100 DATA 0,255,0,0,255,255,255,255,0,
0,255,0,0,0,208,208,7493
1110 DATA 208,208,208,208,208,208,11,1
1,11,11,11,11,11,11,0,0,6402
1120 DATA 0,15,8,8,11,11,0,0,0,240,16,
16,208,208,11,11,52
1130 DATA 8,8,15,0,0,0,208,208,16,16,2
40,0,0,0,134,129,1337
1140 DATA 132,130,160,0,177,129,201,25
5,240,6,153,184,13,200,208,244,3222
1150 DATA 169,0,192,40,176,6,153,184,1
3,200,208,246,96,160,14,162,9161
1160 DATA 80,165,252,32,247,33,144,32,
44,33,165,143,76,44,33,160,4067
1170 DATA 61,162,16,165,139,32,44,33,1
65,138,76,44,33,160,66,162,4202
1180 DATA 80,165,139,32,44,33,165,138,
76,44,33,160,71,162,16,165,4290
1190 DATA 141,32,44,33,165,140,72,134,
136,74,74,74,74,5,136,153,4086
1200 DATA 0,12,200,104,41,15,5,136,153
,0,12,200,96,160,76,166,4851
1210 DATA 142,189,92,33,153,0,12,200,1
89,97,33,153,0,12,200,189,5649
1220 DATA 102,33,153,0,12,96,114,114,1
15,115,102,103,103,115,115,115,5250
1230 DATA 110,113,110,113,116,169,0,13
3,142,141,47,2,32,132,33,32,1506
1240 DATA 148,33,32,169,33,32,237,34,1
69,62,141,47,2,96,162,0,2570
1250 DATA 134,138,134,139,134,140,134,
141,134,144,232,134,143,96,162,8,8761
1260 DATA 189,160,33,157,192,2,202,16,
247,96,24,24,24,24,200,86,3769
1270 DATA 0,0,0,162,0,138,157,0,12,157
,240,12,232,224,240,208,1387
1280 DATA 245,162,10,189,86,34,157,2,1
2,202,16,247,162,35,189,97,6635
1290 DATA 34,157,21,12,202,16,247,169,
101,133,129,169,12,133,130,162,8182
1300 DATA 15,169,66,157,82,12,169,67,1
57,166,13,160,0,169,69,145,5537
1310 DATA 129,160,17,169,68,145,129,16
5,129,24,105,20,133,129,165,130,6805
1320 DATA 105,0,248,33,243,34,133,130,
202,16,218,160,70,140,81,12,6264
1330 DATA 200,140,98,12,200,140,165,13
,200,140,182,13,169,54,141,48,6445
1340 DATA 2,169,34,141,49,2,169,133,14
1,0,2,169,34,141,1,2,632
1350 DATA 169,192,141,14,212,32,247,32
,32,5,33,32,19,33,32,33,8421
1360 DATA 33,76,67,33,112,112,112,198,
0,12,130,134,112,6,6,6,362
1370 DATA 6,6,6,6,6,6,6,6,6,6,6,6,6,6,
134,112,5802
1380 DATA 2,65,54,34,39,37,46,37,50,33
,52,41,47,46,26,48,7102
1390 DATA 111,112,117,108,97,116,105,1
11,110,0,51,116,97,114,116,105,4532
1400 DATA 110,103,0,0,0,0,45,111,115,1
16,0,0,0,0,0,45,5834
1410 DATA 111,100,101,72,169,10,141,23
```

```
,208,141,10,212,169,154,141,0,6792
1420 DATA 2,169,34,141,1,2,104,64,72,1
69,138,141,22,208,141,10,4704
1430 DATA 212,169,136,141,23,208,169,1
80,141,0,2,169,34,141,1,2,2720
1440 DATA 104,64,72,169,70,141,22,208,
141,10,212,165,134,141,23,208,8648
1450 DATA 169,206,141,0,2,169,34,141,1
,2,104,64,72,169,10,141,2493
1460 DATA 23,208,141,10,212,169,133,14
1,0,2,169,34,141,1,2,104,2323
1470 DATA 64,162,0,138,157,0,3,232,208
,250,96,32,227,34,162,0,6741
1480 DATA 138,157,244,34,239,35,0,4,15
7,0,5,157,0,6,157,0,28
1490 DATA 7,232,208,241,169,0,141,7,21
2,169,3,141,29,208,142,12,6371
1500 DATA 208,170,157,8,208,202,16,250
,169,20,141,111,2,160,0,162,6377
1510 DATA 15,189,66,35,153,72,4,153,72
,5,153,72,6,153,72,7,1347
1520 DATA 200,202,16,237,192,128,208,2
31,162,3,189,82,35,157,0,208,8684
1530 DATA 202,16,247,96,204,204,204,20
4,204,204,204,204,51,51,51,51,9719
1540 DATA 51,51,51,51,64,96,128,160,23
0,133,208,23,173,10,210,41,7281
1550 DATA 15,10,10,10,10,133,134,173,1
0,210,41,3,24,105,6,5,9454
1560 DATA 134,133,134,96,165,132,10,10
,10,168,162,7,185,144,35,157,5607
1570 DATA 80,8,200,202,16,246,166,132,
232,224,16,144,2,162,0,134,7518
1580 DATA 132,96,24,24,60,118,126,60,2
4,0,8,28,60,118,126,60,364
1590 DATA 24,0,0,28,62,118,126,60,24,0
,0,24,62,119,126,60,9932
1600 DATA 24,0,0,24,60,127,119,60,24,0
,0,24,60,126,119,62,9920
1610 DATA 24,0,0,24,60,126,118,62,28,0
,0,24,60,126,118,60,9922
1620 DATA 28,8,0,24,60,126,110,60,24,2
4,0,24,60,126,110,60,9964
1630 DATA 56,16,0,24,60,126,110,124,56
,0,0,24,60,126,238,124,3522
1640 DATA 24,0,240,35,235,36,0,24,60,2
38,254,60,24,0,0,24,1237
1650 DATA 124,238,126,60,24,0,0,56,124
,110,126,60,24,0,16,56,9206
1660 DATA 60,110,126,60,24,0,165,147,2
4,105,160,141,1,210,165,148,7523
1670 DATA 141,0,210,165,147,240,2,198,
147,96,164,146,162,9,189,64,8804
1680 DATA 36,153,0,3,200,202,16,246,16
2,3,165,145,157,4,208,24,6970
1690 DATA 105,2,202,16,247,96,0,231,12
9,129,0,0,129,129,231,0,5527
1700 DATA 162,0,173,31,208,201,7,240,1
,232,134,137,96,173,31,208,9630
1710 DATA 201,7,208,5,169,0,133,137,96
,72,165,137,240,4,104,169,7924
1720 DATA 0,96,104,96,162,237,160,36,3
2,216,32,169,127,133,146,169,9483
1730 DATA 120,133,145,32,36,36,32,74,3
6,162,255,142,252,2,232,134,9272
1740 DATA 147,134,149,134,152,134,151,
134,128,169,0,133,20,165,152,240,9959
1750 DATA 2,198,152,165,20,201,2,144,6
,32,20,37,76,147,36,173,3128
1760 DATA 252,2,201,28,208,8,32,46,38,
169,255,141,252,2,32,87,6116
1770 DATA 36,240,224,106,176,11,169,12
7,133,128,32,86,32,32,227,34,5201
1780 DATA 96,106,144,6,169,0,133,152,2
40,201,165,152,208,197,169,10,1502
1790 DATA 133,152,166,142,232,224,5,20
8,2,162,0,134,142,32,67,33,4569
1800 DATA 76,157,236,36,231,37,36,0,0,
53,115,101,0,106,111,121,2763
1810 DATA 115,116,105,99,107,0,116,111
,0,112,108,97,99,101,0,99,2860
1820 DATA 101,108,108,115,0,111,110,0,
98,111,97,114,100,255,32,140,6374
1830 DATA 37,173,120,2,166,151,208,12,
201,15,240,48,133,150,169,8,7536
1840 DATA 133,151,208,19,166,150,221,1
25,37,208,12,189,125,37,133,150,8603
1850 DATA 169,8,56,229,151,133,151,166
,150,165,145,24,125,105,37,168,8278
1860 DATA 201,64,144,4,201,185,144,5,1
69,0,133,151,96,165,146,24,6728
1870 DATA 125,115,37,201,71,144,241,20
1,192,176,237,133,146,132,145,198,4434
1880 DATA 151,169,0,133,149,76,36,36,1
,1,1,0,255,255,255,0,5382
1890 DATA 0,0,1,255,0,0,1,255,0,0,1,25
5,10,9,11,0,8452
1900 DATA 6,5,7,0,14,13,174,16,208,240
,4,202,134,149,96,165,8079
1910 DATA 151,208,251,165,149,208,247,
162,1,134,149,134,143,202,134,144,2505
1920 DATA 32,241,37,248,177,129,240,25
,169,0,145,129,165,138,56,233,385
1930 DATA 1,133,138,133,140,165,139,23
3,0,133,139,133,141,162,150,208,1804
1940 DATA 23,169,10,145,129,165,138,24
,105,1,133,138,133,140,165,139,8166
1950 DATA 105,0,133,139,133,141,162,30
,216,169,15,133,147,134,148,32,7809
1960 DATA 247,32,232,37,227,38,32,5,33
,32,19,33,76,33,33,165,549
1970 DATA 146,56,233,71,74,74,74,72,13
3,129,169,0,133,130,162,3,5492
1980 DATA 6,129,38,130,202,16,249,104,
10,10,24,101,129,133,129,165,6339
1990 DATA 130,105,0,133,130,165,129,24
,105,100,133,129,165,130,105,12,6285
2000 DATA 133,130,165,145,56,233,48,74
,74,74,168,96,165,139,208,4,7755
2010 DATA 165,138,240,50,169,102,133,1
29,169,12,133,130,162,15,169,0,6306
2020 DATA 160,15,145,129,136,16,251,16
5,129,24,105,20,133,129,165,130,7900
2030 DATA 105,0,133,130,202,16,231,32,
132,33,32,247,32,32,5,33,2334
2040 DATA 32,33,33,32,19,33,96,173,252
,2,201,33,208,45,165,20,5738
2050 DATA 72,32,86,32,162,39,189,184,1
3,157,224,16,202,16,247,162,9901
2060 DATA 158,160,38,32,216,32,173,252
,2,201,33,240,249,162,39,189,1664
2070 DATA 224,16,157,184,13,202,16,247
,104,133,20,96,0,0,0,0,536
2080 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,48
,97,117,6079
2090 DATA 115,101,100,255,72,169,0,133
,20,104,197,20,240,8,72,32,4616
2100 DATA 87,36,240,245,104,96,169,0,9
6,32,74,36,165,139,208,11,6055
2110 DATA 165,138,208,7,162,79,160,39,
76,70,39,162,255,142,252,2,8791
2120 DATA 232,134,228,38,223,39,157,32
,255,39,165,153,208,7,162,219,1232
2130 DATA 160,39,76,70,39,162,119,160,
39,32,216,32,32,126,40,208,5695
2140 DATA 79,165,139,208,11,165,138,20
8,7,162,149,160,39,76,70,39,5960
2150 DATA 165,142,201,4,208,9,32,105,3
8,32,87,36,240,201,96,201,8017
2160 DATA 2,144,18,173,16,208,240,191,
162,182,160,39,32,216,32,32,7670
2170 DATA 87,36,240,239,96,169,0,133,2
0,165,20,201,30,176,168,32,6911
2180 DATA 105,38,32,87,36,240,242,96,3
2,216,32,169,120,32,182,38,7061
2190 DATA 96,0,46,111,0,99,101,108,108
```

ATARI

B&C ComputerVisions

3283 Kifer Road
Santa Clara CA 95051
(408) 749-1003

ATARI

SUPER SALE

RECONDITIONED ATARI MERCHANDISE

All merchandise has been tested and reconditioned (except where noted). The "A" price indicates like-new condition. The "B" price indicates product may have scratches and other superficial surface mars. 30 day warranty except where noted.

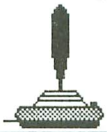
ATARI TRAK BALL

$8.95 B
$10.95 A
FAST ARCADE-LIKE ACTION

ATARI SPACE AGE JOYSTICK

$5.50 B
$6.50 A
LOOKS LIKE A GUN WITH A TRIGGER FOR GREAT SHOOT-EM-UP ARCADE ACTION. COMES IN ORIGINAL BOX.

STANDARD ATARI JOYSTICK



$3.50 B
$4.50 A

2600 REMOTE CONTROL JOYSTICKS (2)

$12.95 B
$15.95 A
REQ. 2600 POWER SUPPLY TO USE ON 800/XL/XE - $5.00

HARD TO FIND TOUCH TABLET

$32.95 B
$39.95 A
IN STYROFOAM BOX

NUMERIC KEY PAD

$6.95 B
$7.95 A
USE WITH VISICALC, BOOKKEEPER & BASIC FOR FAST NUMERIC ENTRY
INCLUDES HANDLER DISK

40 COLUMN 822 PRINTER

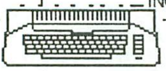
$34.95 B
$39.95 A
INCLUDES ROLL OF THERMAL PAPER

1030 MODEM WITH *EXPRESS*

$29.95 B
$34.95 A
IN STYROFOAM BOX
INCLUDES I/O CABLE & POWER SUP.

16K 400 COMPUTER

$24.95 B
$29.95 A
INCL. POWER SUPPLY AND TV SWITCH BOX
48K UPGRADE $25.00



800 48K COMPUTER

$69.95 B
$79.95 A
INCL. POWER SUPPY, TV SWITCH BOX, BASIC CARTRIDGE & BASIC MANUAL

850 INTERFACE

$79.95 B
$89.95 A
I/O CABLE & POWER SUPPLY
LIMITED SUPPLY

410 PROGRAM RECORDER

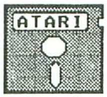
$9.95
IN ORIGINAL BOX - AS IS
NOT TESTED
NO WARRANTY

810 DISK DRIVE

$110.00 B
$120.00 A
COMPLETE WITH CASE, I/O, POWER SUP. & DOS 2

DISKETTES AS LOW AS 20 CENTS

10 FOR $4.00
100 FOR $29.00
1000 FOR $200
UN-NOTCHED WITH DOS 3



THE BOOKKEEPER NEW IN BOX

$24.95
WITHOUT KEYPAD
$29.95
WITH KEYPAD

CARTRIDGES $5.00 EACH

MISSILE COMMAND
STAR RAIDERS
DONKEY KONG
DEFENDER
PAC-MAN
BASIC
SOME IN BOXES

SHIPPING INFORMATION

Prices do not include shipping and handling. Add $5.00 for small items Add $8.00 for disk drive. Calif. residents include 7% sales tax. Mastercard and Visa accepted if your telephone is listed in your local phone directory. Orders may be pre-paid with money order, cashier check, or personal check. Personal checks are held for two weeks before order is processed. C.O.D orders are shipped via UPS and must be paid with cash, cashier check or money order. International and APO orders must be pre-paid with cashier check or money order. $20.00 minimum on all orders. All sales are final - no refunds - prices are subject to change. Phone orders accepted TUESDAY THROUGH FRIDAY from 10:00 am to 6:00 pm PST.

We carry a complete line of ATARI products and have a large public domain library. Write or call for free catalogue. (408) 749-1003 TUE - FRI 10AM - 6 PM

CIRCLE #113 ON READER SERVICE CARD

Life in the Fast Lane *continued*

```
,115,0,104,97,118,101,0,2831
2200 DATA 98,101,101,110,0,112,108,97,
99,101,100,0,111,110,0,116,3287
2210 DATA 104,101,0,98,111,97,114,100,
255,0,0,0,0,0,0,0,7938
2220 DATA 0,0,0,0,44,101,116,0,116,104
,101,114,101,0,98,101,2820
2230 DATA 0,108,105,102,101,1,255,0,0,
0,0,0,0,0,0,52,6297
2240 DATA 104,101,0,99,101,108,108,0,9
9,111,108,111,110,121,0,104,4160
2250 DATA 97,115,0,100,105,101,100,255
,0,0,0,48,114,101,115,115,3885
2260 DATA 0,116,114,105,103,103,101,11
4,0,102,111,114,0,110,101,120,4590
2270 DATA 116,0,103,101,110,101,114,97
,116,105,111,110,255,0,0,0,3779
2280 DATA 0,0,224,39,219,40,35,101,108
,108,0,112,111,112,117,108,5386
2290 DATA 97,116,105,111,110,0,104,97,
115,0,115,116,97,98,105,108,5060
2300 DATA 105,122,101,100,255,169,0,13
3,153,169,81,133,129,169,12,133,8610
2310 DATA 130,169,15,133,155,169,15,13
3,156,169,0,133,154,162,7,188,8386
2320 DATA 118,40,177,129,201,10,208,2,
230,154,202,16,242,166,154,160,2466
2330 DATA 21,177,129,208,6,224,3,208,3
2,240,8,224,2,240,26,224,9807
2340 DATA 3,240,22,230,153,166,153,157
,224,15,165,129,24,105,21,157,8035
2350 DATA 224,13,165,130,105,0,157,224
,14,165,129,24,105,1,133,129,5952
2360 DATA 165,130,105,0,133,130,198,15
6,16,175,165,129,24,105,4,133,6406
2370 DATA 129,165,130,105,0,133,130,19
8,155,16,154,96,0,1,2,20,1696
2380 DATA 22,40,41,42,165,142,201,4,20
8,3,76,39,41,162,7,169,4705
2390 DATA 0,157,88,8,189,144,35,157,96
,8,202,16,242,160,0,166,7710
2400 DATA 153,189,224,13,133,129,189,2
24,14,133,130,189,224,15,208,4,9669
2410 DATA 169,11,208,2,169,12,145,129,
202,208,230,162,7,134,131,166,1157
2420 DATA 131,188,162,41,162,0,185,178
,41,157,88,8,200,232,224,8,9445
2430 DATA 208,244,166,131,188,170,41,1
62,0,185,178,41,157,96,8,200,8696
2440 DATA 232,224,220,40,215,41,8,208,
244,165,142,240,4,201,2,208,1493
2450 DATA 34,166,157,189,242,41,141,0,
210,189,248,41,141,2,210,189,1521
2460 DATA 254,41,141,4,210,165,131,24,
105,161,141,1,210,141,3,210,8611
2470 DATA 141,5,210,169,0,133,20,165,2
0,201,4,176,16,32,105,38,3370
2480 DATA 32,87,36,240,242,32,86,32,16
9,1,72,208,7,198,131,16,5917
2490 DATA 146,169,0,72,160,0,248,166,1
53,189,224,13,133,129,189,224,2967
2500 DATA 14,133,130,189,224,15,208,40
,169,10,145,129,165,138,24,105,7793
2510 DATA 1,133,138,165,139,105,0,133,
139,197,141,240,2,144,34,165,9084
2520 DATA 138,197,140,144,28,165,138,1
33,140,165,139,133,141,76,119,41,8581
2530 DATA 169,0,145,129,165,138,56,233
,1,133,138,165,139,233,0,133,9593
2540 DATA 139,198,153,208,178,32,86,32
,165,143,24,105,1,133,143,165,7405
2550 DATA 144,105,0,133,144,216,166,15
7,232,224,6,208,2,162,0,134,9198
2560 DATA 157,32,247,32,32,5,33,32,33,
33,104,96,0,8,16,24,7986
2570 DATA 32,40,48,56,56,48,40,32,24,1
6,8,0,0,0,56,126,7474
2580 DATA 126,60,24,0,0,0,56,124,124,5
6,24,0,0,0,56,124,9046
2590 DATA 124,24,0,0,0,0,24,60,60,24,0
,0,0,0,0,60,5150
2600 DATA 60,24,216,41,211,42,0,0,0,0,
0,56,56,24,0,0,6563
2610 DATA 0,0,0,56,56,0,0,0,0,0,0,24,2
4,0,0,0,3714
2620 DATA 243,230,217,204,193,182,121,
114,108,102,96,91,60,57,53,50,5919
2630 DATA 47,45,32,74,36,169,0,141,197
,2,141,198,2,141,200,2,6233
2640 DATA 169,110,162,42,141,48,2,142,
49,2,169,60,32,182,38,208,5728
2650 DATA 74,162,9,238,197,2,169,1,32,
182,38,208,62,202,16,243,8999
2660 DATA 169,120,32,182,38,208,52,162
,9,206,197,2,169,1,32,182,6926
2670 DATA 38,208,40,202,16,243,169,60,
32,182,38,208,30,162,3,189,8002
2680 DATA 141,43,157,196,2,202,16,247,
169,170,162,42,141,48,2,142,7786
2690 DATA 49,2,169,127,133,128,32,87,3
6,240,251,96,112,112,112,112,9244
2700 DATA 112,112,112,112,112,112,112,
112,112,112,66,130,42,65,110,42,4924
2710 DATA 0,0,0,0,0,0,0,33,110,97,108,
111,103,0,35,111,1094
2720 DATA 109,112,117,116,105,110,103,
0,112,114,101,115,101,110,116,115,6846
2730 DATA 0,0,0,0,0,0,0,0,112,112,112,
112,112,112,70,197,4660
2740 DATA 42,6,6,6,6,6,112,6,112,112,6
,7,112,112,112,6,812
2750 DATA 65,170,42,0,0,10,0,0,0,0,10,
0,10,10,10,10,4031
2760 DATA 0,10,212,42,143,43,10,10,10,
0,0,0,0,74,0,0,5833
2770 DATA 0,0,74,0,74,0,0,0,0,74,0,0,0
,0,0,0,4102
2780 DATA 0,10,0,0,0,0,10,0,10,10,10,0
,0,10,10,10,3620
2790 DATA 0,0,0,0,0,74,0,0,0,0,74,0,74
,0,0,0,5010
2800 DATA 0,74,0,0,0,0,0,0,0,10,0,0,0,
0,10,0,3198
2810 DATA 10,0,0,0,0,10,0,0,0,0,0,0,0,
74,74,74,6210
2820 DATA 74,0,74,0,74,0,0,0,0,74,74,7
4,74,0,0,128,8938
2830 DATA 128,169,174,128,180,168,165,
128,166,161,179,180,128,172,161,174,49
21
2840 DATA 165,128,128,0,0,0,0,0,35,50,
37,33,52,37,36,0,6997
2850 DATA 34,57,0,0,0,0,0,128,128,128,
128,170,161,173,165,179,9756
2860 DATA 128,128,168,161,167,181,165,
128,128,128,128,64,64,64,64,112,7580
2870 DATA 114,101,115,115,64,64,115,11
6,97,114,116,64,64,64,64,70,4293
2880 DATA 26,150,226,2,227,2,0,32,0,0,
0,0,0,0,0,0,5295
```

●

Listing 2.
Assembly listing.

```
        .OPT OBJ,NO LIST
;
;LIFE IN THE FAST LANE
;Atari 8-bit Ver 1.00 04-20-86
;Copyright 1986 Analog Computing
;Programmed by James Hague
;
;-----------------
; SYSTEM EQUATES
;-----------------
```

```
;
;Page zero
;
DOSINI   =    $0C
RTCLOK   =    $14
ATRACT   =    $4D
;
;Misc.
;
VDSLST   =    $0200
SDLSTL   =    $0230
WSYNC    =    $D40A
SDMCTL   =    $022F
CONSOL   =    $D01F
RANDOM   =    $D20A
STICK0   =    $0278
TRIG0    =    $D010
NMIEN    =    $D40E
CH       =    $02FC
CHBAS    =    $02F4
OLDSET   =    $E000
;
;Audio
;
SKCTL    =    $D20F
AUDCTL   =    $D208
AUDF1    =    $D200
AUDC1    =    $D201
AUDF2    =    $D202
AUDC2    =    $D203
AUDF3    =    $D204
AUDC3    =    $D205
;
;P/M Graphics
;
GRACTL   =    $D01D
GPRIOR   =    $026F
PMBASE   =    $D407
HPOSP0   =    $D000
HPOSP1   =    $D001
HPOSP2   =    $D002
HPOSP3   =    $D003
HPOSM0   =    $D004
HPOSM1   =    $D005
HPOSM2   =    $D006
HPOSM3   =    $D007
SIZEP0   =    $D008
SIZEP1   =    $D009
SIZEP2   =    $D00A
SIZEP3   =    $D00B
SIZEM    =    $D00C
;
;Colors
;
PCOLR0   =    $02C0
PCOLR1   =    $02C1
PCOLR2   =    $02C2
PCOLR3   =    $02C3
COLOR0   =    $02C4
COLOR1   =    $02C5
COLOR2   =    $02C6
COLOR3   =    $02C7
COLBAK   =    $02C8
HCOLR0   =    $D016
HCOLR1   =    $D017
;
;System Routines
;
SIOINV   =    $E465
SETVBV   =    $E45C
XITVBL   =    $E462
;
;-------------------
; PROGRAM VARIABLES
;-------------------
         *=  $80
;
;Global Misc.
;
VBICTL   .DS 1         ;VBI CTrL. flag
PNT1     .DS 2         ;Misc. PoiNTer
TEMP     .DS 1         ;TEMP storage
IMAGE    .DS 1         ;IMAGE pointer
COLTIM   .DS 1         ;COLor TIMer
BORDER   .DS 1         ;BORDER color
CURSOR   .DS 1         ;CURSOR color
COLMSK   .DS 1         ;COLor MaSK
CONKEY   .DS 1         ;CONsole KEY flg
;
;Cell Database
;
POP      .DS 2         ;bcd POPulation
MOST     .DS 2         ;bcd max pop.
MODE     .DS 1         ;growth MODE
GENER    .DS 2         ;bcd GENERation
;
;Placement routines
;
CURX     .DS 1         ;CURsor X pos.
CURY     .DS 1         ;CURsor Y pos.
SOUND    .DS 1         ;SOUND flag
TONE     .DS 1         ;sound TONE
PRESS    .DS 1         ;trig PRESS flag
DIR      .DS 1         ;stick DIRect.
DIST     .DS 1         ;cursor DISTance
SELECT   .DS 1         ;SELECT key time
;
;Regeneration
;
ACTIVE   .DS 1         ;ACTIVE cells
COUNT    .DS 1         ;temp COUNTer
ROW      .DS 1         ;current ROW
COL      .DS 1         ;current COLUMN
CHORD    .DS 1         ;current CHORD #
;
;Color Constants
;
@GENTXT  =    $C8      ;GENer label
@GENNUM  =    $56      ;GENer number
@GR0BAK  =    $00      ;GR.0 BAcKground
```

```
@GR0LUM  =    $0A      ;GR.0 text LUM.
@NUM1    =    $8A      ;NUMber color 1
@NUM2    =    $88      ;NUMber color 2
@BOARD   =    $18      ;BOARD squares
@CELL    =    $46      ;CELLS
;
;Misc. Constants
;
EOL      =    $FF      ;EOL character
;
;-------------
; GLOBAL DATA
;-------------
;
         *=  $00
PM       .DS $0300     ;Player/Missile
MISSL    .DS $0100     ;memory...
PL0      .DS $0100
PL1      .DS $0100
PL2      .DS $0100
PL3      .DS $0100
CHSET    .DS $0400     ;New CHar SET
DISP     .DS 480       ;DISPlay memory
BOARD    =   DISP+102
TEXT     =   DISP+440
CADRL    .DS $0100     ;Regeneration
CADRH    .DS $0100     ;data storage...
CTYPE    .DS $0100
TSAVE    .DS 40        ;Temp buffer
;
;------------------
; PROGRAM MAINLINE
;------------------
;
         *=  $2000
RESET
         CLD           ;Kill decimal
         JSR SETSYS    ;Set-up system
         JSR REDEFINE ;Redef. chset
         JSR TITLE     ;Do title scrn.
         JSR SETUP     ;Set-up program
MAIN     JSR PLACE     ;Place cells
         JSR LIFE      ;Create life!
         JMP MAIN      ;Loop forever
;
;-----------------
; VBLANK MAINLINE
;-----------------
;
VBI
         CLD           ;Kill decimal
         LDA VBICTL    ;Get VBI flag
         BMI NOVBI     ;If <0, exit
         LDA #0        ;else, turn off
         STA ATRACT    ;attract mode
         JSR ANIMATE   ;Animate cells
         JSR BCOLOR    ;Do border color
         LDA VBICTL    ;Get flag again
         BNE NOVBI     ;If not 0, exit
         JSR DING      ;Do ding sound
         INC CURSOR    ;Flash cursor...
         LDA CURSOR
         STA COLOR3
NOVBI    JMP XITVBL    ;Bye-bye VBI!
;
;---------------
; SET-UP SYSTEM
;---------------
;
;Among other things, set up
;the VBI and any variables
;associated with it that
;need be initialized only once.
;
SETSYS
         JSR SIOINV    ;Init sounds...
         JSR QUIET
         LDA # <RESET ;Steal the
         STA DOSINI    ;reset vector...
         LDA # >RESET
         STA DOSINI+1
         LDX #$FF      ;Init VBI
         STX VBICTL    ;variables...
         STX COLTIM
         INX
         STX IMAGE
         LDY # <VBI
         LDX # >VBI
         LDA #7
         JMP SETVBV
;
;----------------
; SHUT OFF SOUND
;----------------
;
;This is a general purpose
;routine that initializes
;sound channels 1,2 & 3.  It is
;not responsible for handling
;routine-specific flags.
;
QUIET
         LDA #3        ;Initialize
         STA SKCTL     ;sound and
         LDA #0        ;turn off
         STA AUDCTL    ;channels 1,2
         STA AUDC1     ;and 3...
         STA AUDC2
         STA AUDC3
         RTS
;
;------------------
; REDEFINE CHARSET
;------------------
;
;Move charset
;
REDEFINE
         LDX #0        ;Move set from
MOVSET   LDA OLDSET,X ;$E000 to
```

```
         STA CHSET,X ;RAM...
         LDA OLDSET+$0100,X
         STA CHSET+$0100,X
         LDA OLDSET+$0200,X
         STA CHSET+$0200,X
         LDA OLDSET+$0300,X
         STA CHSET+$0300,X
         INX
         BNE MOVSET
;
;Redefine a few chars
;
         LDX #63       ;Redefine chars
REDEF    LDA CHRDAT,X ;2 thru 10 as
         STA CHSET+16,X ;the arena's
         DEX           ;borders
         BPL REDEF
;
;Install new set
;
         LDA # >CHSET
         STA CHBAS
         RTS
;
;Character data
;
CHRDAT   .BYTE $00,$00,$00,$FF
         .BYTE $00,$00,$FF,$FF
;
         .BYTE $FF,$FF,$00,$00
         .BYTE $FF,$00,$00,$00
;
         .BYTE $D0,$D0,$D0,$D0
         .BYTE $D0,$D0,$D0,$D0
;
         .BYTE $0B,$0B,$0B,$0B
         .BYTE $0B,$0B,$0B,$0B
;
         .BYTE $00,$00,$00,$0F
         .BYTE $08,$08,$0B,$0B
;
         .BYTE $00,$00,$00,$F0
         .BYTE $10,$10,$D0,$D0
;
         .BYTE $0B,$0B,$08,$08
         .BYTE $0F,$00,$00,$00
;
         .BYTE $D0,$D0,$10,$10
         .BYTE $F0,$00,$00,$00
;
;---------------
; PRINT MESSAGE
;---------------
;
;This routine prints an EOL
;terminated internally-coded
;string on the message line.
;
;Call with:
;  x reg. - lo byte of string
;  y reg. - hi byte of string
;
PRINT
         STX PNT1      ;Save string
         STY PNT1+1    ;address
         LDY #0
DOCHAR   LDA (PNT1),Y ;Get a char
         CMP #EOL      ;EOL?
         BEQ PADIT     ;Yes, pad line.
         STA TEXT,Y    ;No, show char
         INY           ;and do the
         BNE DOCHAR    ;next one
;
;Pad rest of line with blanks
;
PADIT    LDA #0        ;Blank char
DOPAD    CPY #40       ;All done?
         BCS PREXIT    ;Yes, return.
         STA TEXT,Y    ;No, show blank
         INY           ;Loop until
         BNE DOPAD     ;line is done
PREXIT   RTS
;
;--------------------------
; NUMERICAL DISPLAY ROUTINES
;--------------------------
;
;Show generation number
;
SHOGEN
         LDY #14       ;Screen offset
         LDX #80       ;Color mask
         LDA GENER+1 ;Print...
         JSR BCD
         LDA GENER
         JMP BCD
;
;Show population
;
SHOPOP
         LDY #61       ;Screen offset
         LDX #16       ;Color mask
         LDA POP+1     ;Print...
         JSR BCD
         LDA POP
         JMP BCD
;
;Show starting pop.
;
SHOSTRT
         LDY #66       ;Screen offset
         LDX #80       ;Color mask
         LDA POP+1     ;Print...
         JSR BCD
         LDA POP
         JMP BCD
;
;Show largest pop.
;
SHOMOST
         LDY #71       ;Screen offset
         LDX #16     ;Color mask
         LDA MOST+1  ;Print...
         JSR BCD
         LDA MOST    ;Fall into BCD..
;
;Print 2 BCD digits
;
;Call with:
;  a reg. - BCD data
;  x reg. - color mask+16
;  y reg. - offset from DISP
;
BCD
         PHA         ;Save BCD data
         STX COLMSK  ;Save color mask
         LSR A       ;Move upper 4
         LSR A       ;bits into lower
         LSR A       ;4 bits...
         LSR A
         ORA COLMSK  ;make it ATASCII
         STA DISP,Y  ;and show it.
         INY         ;Goto next pos.
         PLA         ;Get original #
         AND #$0F    ;mask upper half
         ORA COLMSK  ;make it ATASCII
         STA DISP,Y  ;and show it.
         INY         ;Goto next pos.
         RTS         ;and leave.
;
;-----------
; SHOW MODE
;-----------
;
SHOMODE
         LDY #76     ;Screen offset
         LDX MODE    ;Get mode #
         LDA LET1,X  ;Print first
         STA DISP,Y  ;letter of mode
         INY
         LDA LET2,X  ;Print second
         STA DISP,Y  ;letter...
         INY
         LDA LET3,X  ;and print
         STA DISP,Y  ;final letter
         RTS
;
;Mode data
;
LET1     .SBYTE +64,"RRSSF"
LET2     .SBYTE +64,"GGSSS"
LET3     .SBYTE +64,"NQNQT"
;
;---------------------
; MAIN SET-UP ROUTINE
;---------------------
;
SETUP
         LDA #0      ;Set mode to
         STA MODE    ;0 (RGN)
         STA SDMCTL  ;Turn DMA off
         JSR SETCELL ;Set cell data
         JSR SETCOLR ;Set colors
         JSR SETDISP ;Set-up display
         JSR SETPM   ;Set PM grphics
         LDA #62     ;Enable DMA for
         STA SDMCTL  ;screen and PM
         RTS         ;and exit!
;
;--------------------
; INIT CELL DATABASE
;--------------------
;
;Reset all cell variables except
;MODE which does not need to be
;changed when the cells are.
;
SETCELL
         LDX #0      ;Clear out the
         STX POP     ;population...
         STX POP+1
         STX MOST    ;largest pop...
         STX MOST+1
         STX GENER+1 ;Set generation
         INX         ;to 1 (BCD)
         STX GENER
         RTS
;
;---------------
; SET UP COLORS
;---------------
;
SETCOLR
         LDX #8      ;Do 9 color regs
COLR     LDA COLORS,X ;Get a color
         STA PCOLR0,X ;& put in reg.
         DEX         ;Repeat til done
         BPL COLR
         RTS
;
;Initial colors
;
COLORS
         .BYTE @BOARD ;PCOLR0
         .BYTE @BOARD ;PCOLR1
         .BYTE @BOARD ;PCOLR2
         .BYTE @BOARD ;PCOLR3
         .BYTE @GENTXT ;COLOR0
         .BYTE @GENNUM ;COLOR1
         .BYTE @GR0BAK ;COLOR2
         .BYTE 0     ;COLOR3
         .BYTE 0     ;COLBAK
;
;--------------------
; SET-UP MAIN DISPLAY
;--------------------
;
;Clear display memory
;
SETDISP
         LDX #0      ;Clear 480 bytes
         TXA         ;in two blocks
CLRD     STA DISP,X  ;of 240 each...
         STA DISP+240,X
         INX
         CPX #240
         BNE CLRD
;
;Print labels
;
         LDX #10     ;Print
PRL1     LDA GLABEL,X ;"GENERATION:"
         STA DISP+2,X ;on line 0...
         DEX
         BPL PRL1
         LDX #35     ;Print cell
PRL2     LDA LABELS,X ;data labels
         STA DISP+21,X ;on line 1...
         DEX
         BPL PRL2
;
; Draw borders
;
         LDA # <[DISP+101] ;Draw the
         STA PNT1    ;long sections
         LDA # >[DISP+101] ;of the
         STA PNT1+1  ;border
         LDX #15     ;(15 chars long)
BDRAW    LDA #66     ;Print a piece
         STA DISP+82,X ;of top line
         LDA #67     ;and the bottom
         STA DISP+422,X ;line plus
         LDY #0      ;part of the
         LDA #69     ;left side
         STA (PNT1),Y
         LDY #17     ;and finally
         LDA #68     ;a piece of the
         STA (PNT1),Y ;right!
         LDA PNT1    ;Add 20 to the
         CLC         ;address of the
         ADC #20     ;current line so
         STA PNT1    ;that we'll know
         LDA PNT1+1  ;where to print
         ADC #0      ;the next side
         STA PNT1+1  ;chars.
         DEX         ;Finish drawing!
         BPL BDRAW
;
         LDY #70     ;Add upper left
         STY DISP+81 ;corner,
         INY         ;upper right,
         STY DISP+98
         INY         ;lower left,
         STY DISP+421
         INY         ;and finally the
         STY DISP+438 ;lower right!
;
;Point to main DL
;
         LDA # <DL
         STA SDLSTL
         LDA # >DL
         STA SDLSTL+1
;
;Start up DLIs
;
         LDA # <DLI1 ;Point to first
         STA VDSLST  ;DLI...
         LDA # >DLI1
         STA VDSLST+1
         LDA #192    ;Enable DLIs and
         STA NMIEN   ;VBIs
;
;Show cell data
;
         JSR SHOGEN  ;Show generation
         JSR SHOPOP  ;current pop.
         JSR SHOSTRT ;starting pop.
         JSR SHOMOST ;Maximum pop.
         JMP SHOMODE ;and mode
;
;Main display list
;
DL       .BYTE 112,112,112
         .BYTE 198, <DISP, >DISP
         .BYTE 130
         .BYTE 134
         .BYTE 112
         .BYTE 6,6,6,6,6,6,6,6,6
         .BYTE 6,6,6,6,6,6,6,6
         .BYTE 134
         .BYTE 112
         .BYTE 2
         .BYTE 65, <DL, >DL
;
;Misc. display data
;
GLABEL   .SBYTE "GENERATION:"
LABELS   .SBYTE "Population "
         .SBYTE "Starting    Most"
         .SBYTE "        Mode"
;
;------------------
; MAIN SCREEN DLIs
;------------------
;
;Set the text luminance for
;the label line (1)
;
DLI1     PHA         ;Save acc.
         LDA #@GR0LUM ;Set text
         STA HCOLR1  ;luminance
         STA WSYNC   ;Wait for sync
         LDA # <DLI2 ;and point to
         STA VDSLST  ;next dli...
         LDA # >DLI2
         STA VDSLST+1
         PLA         ;Restore acc.
         RTI
;
;Set colors for the numbers on
;line 2
;
DLI2     PHA         ;Save acc.
         LDA #@NUM1  ;Set number
         STA HCOLR0  ;color 1,
         STA WSYNC   ;wait for sync,
         LDA #@NUM2  ;and set number
         STA HCOLR1  ;color 2
         LDA # <DLI3 ;Point to next
         STA VDSLST  ;dli...
         LDA # >DLI3
         STA VDSLST+1
         PLA         ;Restore acc.
         RTI
;
;Set colors for the cells and
;border of board
;
DLI3     PHA         ;Save acc.
         LDA #@CELL  ;Set cell color
         STA HCOLR0
         STA WSYNC   ;Wait for sync
         LDA BORDER  ;and set border
         STA HCOLR1  ;color
         LDA # <DLI4 ;Point to final
         STA VDSLST  ;dli...
         LDA # >DLI4
         STA VDSLST+1
         PLA         ;Restore acc.
         RTI
;
;Set text luminance for message
;line
;
DLI4     PHA         ;Save acc
         LDA #@GR0LUM ;Set text
         STA HCOLR1  ;luminance, and
         STA WSYNC   ;wait for sync
         LDA # <DLI1 ;Point back to
         STA VDSLST  ;first dli...
         LDA # >DLI1
         STA VDSLST+1
         PLA         ;Restore acc.
         RTI
;
;----------------------
; CLEAR MISSILE MEMORY
;----------------------
;
CLRMISS
         LDX #0      ;Fill missile
         TXA         ;memory with
CCLR     STA MISSL,X ;zeroes
         INX
         BNE CCLR
         RTS
;
;--------------------
; SET-UP PM GRAPHICS
;--------------------
;
;Clear pm memory
;
SETPM
         JSR CLRMISS ;Clear missiles
         LDX #0      ;Clear players
         TXA         ;0-3...
PMCLR    STA PL0,X
         STA PL1,X
         STA PL2,X
         STA PL3,X
         INX
         BNE PMCLR
;
;Init pm
;
         LDA # >PM   ;Point to pm
         STA PMBASE  ;memory
         LDA #3      ;Turn on pm
         STA GRACTL  ;graphics
         STX SIZEM   ;Set missl. size
         TAX         ;Set all players
SETSIZE  STA SIZEP0,X ;to
         DEX         ;quadruple width
         BPL SETSIZE
         LDA #20     ;Set priority
         STA GPRIOR  ;w/5th player on
;
;Draw board squares
;
         LDY #0      ;Draw squares
SQ1      LDX #15     ;in players 0-3
SQ2      LDA SQRDAT,X ;Get data byte
         STA PL0+72,Y ;and put it in
         STA PL1+72,Y ;each player...
         STA PL2+72,Y
         STA PL3+72,Y
         INY         ;Next pm byte
         DEX         ;Finish up a set
         BPL SQ2     ;of 4 squares
         CPY #128
         BNE SQ1     ;Complete board
;
         LDX #3      ;Set player
MOVSQ    LDA PLHPOS,X ;horizontal
         STA HPOSP0,X ;positions...
         DEX
         BPL MOVSQ
         RTS
;
;Square data
;
SQRDAT   .BYTE $CC,$CC,$CC,$CC
         .BYTE $CC,$CC,$CC,$CC
         .BYTE $33,$33,$33,$33
         .BYTE $33,$33,$33,$33
;
;Player horiz. positions
;
PLHPOS   .BYTE 64,96,128,160
;
```

ATARI®

MICROTYME

ATARI®

VISA®

MasterCard

A DIVISION OF MICRO PERIPHERALS, INC.

P.O. BOX 369 • KETTERING, OHIO 45409

HARDWARE/ACCESSORIES

ATARI

ST's Color or Mono	CALL
ST Monitor Cables	CALL
SF 314 Double Sided Drive	CALL
Supra 20, 30, 60 Meg Hard Disks	CALL
SH 204 20 Megabyte Hard Disk	CALL
SC 1224 RGB Color Monitor	CALL
130 XE	CALL
65 XE	CALL
1050 Disk Drive	CALL
1020 Color Printer/Plotter	27
Astra ''The One''	284
7800 Pro System & Software	CALL
Power Supply 400/800/810 1050/850	15
Power Supply 600/800 XL, 130 XE	26
Power Supply for Indus GT	15

PANASONIC

KX-P1080*i*, 120 cps,	CALL
KX-P1091*i*, 160 cps	CALL
KX-P1092*i*, 240 cps	CALL
KX-P1592 136 col, True 180 cps	549
KX-P3131 L.Q. Daisy, 80 col	279
KX-P3151 L.Q. Daisy, 136 col	429
KX-P110 Ribbon, Blk	9
COLOR RIBBONS	11

CITIZEN

MSP-10	CALL
MSP-15	CALL

STAR MICRONICS

NX-10 (80 col)	CALL
NP-10 .... NEW MODEL	179
NL-10	CALL
NX-15 (135 col)	429
STAR SG-10 Ribbons	4
Ribbons NL or NX	CALL

MODEMS

SX-212	CALL
XM-301 Direct Connect	CALL
HAYES 1200 Smartmodem	399
US ROBOTICS COURIER 2400-100% Hayes!	429
HABA 1200 (Hayes Compatible)	119
VOLKSMODEM VM 520 (Hayes Compat)	139
AVATEX 1200 bps .... **Special**	89
AVATEX 1200 H.C. (Hayes Compatible)	129

INTERFACES/BUFFERS

ATARI 850 .... In Stock!	119
P:R: CONNECTION (100% 850 compatible)	69
CABLES - We've Got 'Em	CALL
XETEC GRAPHIX AT	39
SUPRA/MPP MICROPRINT	39
SPECIAL SUPRA MICROSTUFFER (64K)	59
SUPRA/MPP 1150	CALL

MIDI HEADQUARTERS

CZ's	CALL
Cables	CALL
Hybrid Arts Software	CALL
Other MIDI Software	CALL

MONITORS

TEKNIKA MJ-22 RGB and Composite	289
NAP Green W/Audio	85
NAP Amber W/Audio	90
NAP Composite Color	CALL
NAP, RGB and Composite	CALL
SANYO Composite	189

EPSON

LX-86 (80 col)	CALL
FX-85 (80 col)	CALL
FX-286 200 cps (135 col)	CALL
Tractors	CALL

ACCESSORIES

ST - COVERS, Heavy Grade Vinyl	8
ST - MOUSE MAT	10
ST - 6' Printer Cable	19
ST - Modem Cable (to Hayes, etc.)	17
ST - Monitor Stand, Swivel & Tilt	15
ST Clock Internal or Plug-In	ea 41
Disk File for 3.5'' disks (holds 40)	9
Flip N File DATA CASE (holds 50) 5¼	8
Disk File, with Lock (holds 100!) 5¼	13
Power Strip, 6 outlet, (15 amp Surge)	15
Deluxe Power Strip w/Spike & Surge	24
Printer Stand, Heavy Duty, Sloping	13
ATARI ''Standard'' Joystick	7
Epyx 500XJ Joystick	15
WICO Bat Handle Stick	17
Competition Pro 5000X Stick	18
6' Atari Serial I/O Cable	7
Book of Adventure Games I or II	15
Disk Notcher	5
Compuserve Starter Kit	24
U.S. DOUBLER (Dbl. Density for 1050)	49
U.S. DOUBLER without Sparta DOS	29
''Duplicator'' for 1050 Disk Drive	125

PRINTER SUPPLIES

MAILING LABELS, White, 500 pack	3
per 1000	4
Blu, Pnk, Gn, Yel, 800 pack (200 ea)	9
per 500, any 1 color	5
per 1000, any 1 color	7
Big Labels, 1-7/16x4'', White, per 500	5
PRINTER PAPER, Micro-Fine perfs, 20 lb.	
500 sheets, Pure White Bond	8
1000 sheets, same as above	14
PRINTSHOP ''Rainbow'' Color Paper Packs	
Pastels (5 colors), 50 sheets of ea	12
Matching Envelopes, 20 of each	6
Brights (8 colors), 50 sheets of ea	29
Matching Envelopes, 20 of each	10
ALL 13 colors, 50 sheets of each	39
Matching Envelopes, 20 of each	14

SOFTWARE

We can't list it all. . .call if you don't see it!

8-BIT SOFTWARE

ATARI

Ataпwriter Plus	35
Proofreader	12
Silent Butler	18
Learning Phone	15
Star Raiders II	13
OTHERS	CALL

ACCESS

Triple Pack	14
Leader Board	26
Tournament Disks	14

ACCOLADE

Hardball	20
Fight Night	20

BATTERIES INCLUDED

Paperclip w/Spellpak	39
Homepak	15
B-GRAPH	27

ARTWORKX

Strip Poker	19
S.P. Data Disks each	13

BRODERBUND

Championship Loderunner	18
Karateka	18
Loderunner	23
Print Shop	28
Print Shop Companion	24
Graphics Library 1,2, or 3	ea 16

DATASOFT

Alternate Reality	27
Never Ending Story	21

ELECTRONIC ARTS

Racing Destruction Set	12
Super Boulderdash	12
Touchdown Football	12
Ultima IV	43
Many Other Titles . . . **CALL**	12

EPYX

World Karate Championship	19

INFOCOM

Hitchhiker's Guide	23
Leather Goddesses	22
All Others	CALL

O.S.S.

Action	46
Basic XL	36
MAC/65	46
Tool Kits for above	ea 19
Basic XE	46

SSI

Mech Brigade	38
War Game Const. Set	19
NAM	25
USAAF	36
Phantassie	25
Wizard's Crown	26
OTHERS	CALL

SUBLOGIC

Flight Simulator II	37
Scenery Disks	ea 15
Western Scenery Set	74

SYNAPSE

SYN CALC	31
SYN CALC TEMPLATES	13
SYN FILE +	31

XLENT

Typesetter	22
Rubber Stamp	20
Page Designer	20
Megafont II	16
P.S. Interface	19
Word Processor	20

ST SOFTWARE

ATARI

1st Word Plus . . . . . NEW	32
Microsoft Write	59
Battlezone	19
Crystal Castles	19
Joust	19
Star Raiders	19

ABACUS

Text Pro	33
Datatrieve	33
Power Plan	52
PC Board Designer	CALL

ACCESS

Leader Board	26
Tournament Disks	ea 14

ACCOLADE

Mean 18	33
Hardball	CALL

ACTIVISION

Hacker II	33
Little Computer People	31
Shanghai	29
Tass Times	32

ANTIC

A-CALC PRIME	59
CAD-3D	36
CAD-3D **STEREO**	65
STEREOTEK 3-D Pkg.	125
Flash	29
Expert Opinion	72
Others	CALL

ARTWORX

Strip Poker	26
Bridge 4.0	20
Peggammon	12

BATTERIES INCLUDED

B-Graph Elite	CALL
D.E.G.A.S. Elite	49
Paperclip Elite	CALL
Thunder	26
Time Link	32

ELECTRONIC ARTS

Arctic Fox	27
Auto Duel	32
Marble Madness	CALL
Sky Fox	29
Chessmaster 2000	31
Ultima III	39
Ultima IV	39

EPYX

Summer Games	26
Winter Games	26
Super Cycle	26
Championship Wrestling	26
World Games	26
Rogue	26
Trilogy	26

INFOCOM

All Titles	CALL

MICHTRON

Cornerman	32
DOS Shell	32
M-Disk	26
Mighty Mail	32
Personal Money Manager	32
Pinball Factory	25
Utilities	39
Time Bandits	26
Trim Base	64
Major Motion	26

FIREBIRD

Guild of Thieves	29
Starglider	29
The Pawn	29

REGENT

Regent Base	67
Regent Word II	67

SIERRA-ON-LINE

ST ONE WRITE	39
Donald Duck's Playground	19
Ultima II	39
Kings Quest I, II or III	ea 33
Space Quest	33

TIMEWORKS

Data Manager ST	52
Swift Calc ST	52
Word Writer ST	52

UNICORN

Math Wizard	26
Fraction Action	26
Decimal Dungeon	26
Read & Rhyme	26

UNISON WORLD

Print Master Plus	26
Art Gallery I, II or III	ea 19

ST LANGUAGES

MODULA 2	54
MOD 2 Developer's Kit	99
MARK WILLIAMS C	119
LATTICE C	109
GST-ASM	45
ASSEM PRO	39
GFA Basic	52

DISKETTES

Prices Are Per Box/Pack of 10

5¼"	GENERIC		SONY	MAXELL	BONUS		WABASH
	SS/DD	DS/DD	SS/DD	DS/DD	SS/DD	DS/DD	SS/DD
2-5	6.95	7.95	9.50	11.95	8.95	10.95	8.50
6-10	5.95	6.95	8.50	10.95	7.95	9.95	7.50

Prices Are Per Box/Pack of 10

3½	SONY		MAXELL	VERBATIM
	SS/DD	DS/DD	SS/DD	SS/DD
2-5	15.00	24.00	15.00	15.00
6-10	14.00	22.00	14.00	14.00

Rainbow Colored Centech Disks (2 ea. of 10 colors per pkg)	**17**
''Silver'' Centech Disks (20 Pack)	**17**

TO ORDER, CALL TOLL FREE 1-800-255-5835

M-F 9am - 9pm • EST • SAT 10am - 4pm
Ohio Residents, Order Status or Tech, Info Call (513) 294-6236

TERMS AND CONDITIONS

• NO EXTRA CHARGES FOR CREDIT CARDS! • We do not bill until we ship • Minimum order $20 • C.O.D. to continental U.S. only, add $3 • Ohio residents add 6% sales tax • Please allow 3 weeks for personal or company checks to clear • Shipping/Handling: (Call for Quote). Hardware, minimum $4; Software and most accessories, minimum $3 • Overnight shipment available at extra charge • We ship to Alaska, Hawaii, Puerto Rico (UPS Blue Label Only), APO, and FPO • Canadian orders, actual shipping plus 5%, minimum $5 • All defective products require a return authorization number to be accepted for repair or replacement • No free trials or credit • Returns subject to 15% re-stocking charge • Due to changing market conditions, call toll free for latest price and availability of product. FOR YOUR PROTECTION, WE CHECK ALL CREDIT CARD ORDERS FOR FRAUD.

CIRCLE #114 ON READER SERVICE CARD

```
;-------------------------
; VB BORDER COLOR HANDLER
;-------------------------
;
BCOLOR
          INC COLTIM   ;Add 1 to timer
          BNE OLDCOL   ;If not 0, exit
          LDA RANDOM   ;Get random #
          AND #$0F     ;Make it <16
          ASL A        ;Mult by 16...
          ASL A
          ASL A
          ASL A
          STA BORDER   ;Save the result
          LDA RANDOM   ;Get random #
          AND #3       ;Make it <4,
          CLC          ;add 6, and add
          ADC #6       ;it to previous
          ORA BORDER   ;result for the
          STA BORDER   ;new color!
OLDCOL    RTS
;
;-------------------
; VB CELL ANIMATION
;-------------------
;
ANIMATE
          LDA IMAGE    ;Get image #
          ASL A        ;and multiply
          ASL A        ;it by 8 for an
          ASL A        ;index into the
          TAY          ;cell data
          LDX #7       ;Copy 8 bytes
ANIM      LDA CELIMG,Y ;from the data
          STA CHSET+80,X ;table into
          INY          ;char number 10
          DEX
          BPL ANIM
          LDX IMAGE    ;Point to next
          INX          ;image
          CPX #16      ;Last one done?
          BCC ANXIT    ;No, exit
          LDX #0       ;Yes, reset it
ANXIT     STX IMAGE    ;Save new num
          RTS
;
;Cell image data
;
CELIMG
          .BYTE $18,$18,$3C,$76
          .BYTE $7E,$3C,$18,$00
;
          .BYTE $08,$1C,$3C,$76
          .BYTE $7E,$3C,$18,$00
;
          .BYTE $00,$1C,$3E,$76
          .BYTE $7E,$3C,$18,$00
;
          .BYTE $00,$18,$3E,$77
          .BYTE $7E,$3C,$18,$00
;
          .BYTE $00,$18,$3C,$7F
          .BYTE $77,$3C,$18,$00
;
          .BYTE $00,$18,$3C,$7E
          .BYTE $77,$3E,$18,$00
;
          .BYTE $00,$18,$3C,$7E
          .BYTE $76,$3E,$1C,$00
;
          .BYTE $00,$18,$3C,$7E
          .BYTE $76,$3C,$1C,$08
;
          .BYTE $00,$18,$3C,$7E
          .BYTE $6E,$3C,$18,$18
;
          .BYTE $00,$18,$3C,$7E
          .BYTE $6E,$3C,$38,$10
;
          .BYTE $00,$18,$3C,$7E
          .BYTE $6E,$7C,$38,$00
;
          .BYTE $00,$18,$3C,$7E
          .BYTE $EE,$7C,$18,$00
;
          .BYTE $00,$18,$3C,$EE
          .BYTE $FE,$3C,$18,$00
;
          .BYTE $00,$18,$7C,$EE
          .BYTE $7E,$3C,$18,$00
;
          .BYTE $00,$38,$7C,$6E
          .BYTE $7E,$3C,$18,$00
;
          .BYTE $10,$38,$3C,$6E
          .BYTE $7E,$3C,$18,$00
;
          .INCLUDE #D:FLIFE2.M65
          *=  $02E0
          .WORD RESET
          .END
```

●

```
;LIFE IN THE FAST LANE
;Atari 8-bit Ver 1.00 4-20-86
;Copyright 1986 Analog Computing
;Programmed by James Hague
;
;Part II
;
;-----------------
; VB DING HANDLER
;-----------------
;
DING
          LDA SOUND    ;Get sound vol.
          CLC          ;add distortion
          ADC #$A0     ;value
          STA AUDC1    ;and set it
          LDA TONE     ;Set proper
          STA AUDF1    ;frequency
          LDA SOUND    ;If vol is 0
          BEQ XITDING  ;then exit!
          DEC SOUND    ;Else lower vol
XITDING   RTS
;
;-------------
; DRAW CURSOR
;-------------
;
;The cursor consists of missles
;zero thru three.
;
DRAWCUR
          LDY CURY     ;Move 10 bytes
          LDX #9       ;of cursor shape
CDRAW     LDA CURDAT,X ;data into
          STA MISSL,Y  ;missile memory
          INY
          DEX
          BPL CDRAW
;
          LDX #3       ;Set horizontal
          LDA CURX     ;positions for
SETCHP    STA HPOSM0,X ;the 4
          CLC          ;missiles.  Each
          ADC #2       ;one is two
          DEX          ;units away from
          BPL SETCHP   ;previous
          RTS          ;missile.
;
;Cursor shape data: 8 bytes of
;actual data plus leading and
;trailing zeroes to make motion
;easier.
;
CURDAT    .BYTE 0,231,129,129,0
          .BYTE 0,129,129,231,0
;
;----------------------
; CONSOLE KEY ROUTINES
;----------------------
;
;Init console key flag
;
;Set CONKEY to 0 if no console
;keys are pressed, else set it
;to 1
;
INITCON
          LDX #0
          LDA CONSOL   ;Get keys
          CMP #7       ;If not down
          BEQ NOCKEY   ;set to 0,
          INX          ;else set to 1
NOCKEY    STX CONKEY
          RTS
;
;Check console keys
;
;Return 0 in acc. if no console
;keys are pressed, else return
;the value of CONSOL in acc.
;
CHKCON
          LDA CONSOL   ;Any console
          CMP #7       ;keys pressed?
          BNE CCHK1    ;No, check more!
          LDA #0       ;Yes, zero key
          STA CONKEY   ;flag and return
          RTS          ;a zero!
;
CCHK1     PHA          ;Save CONSOL
          LDA CONKEY   ;Flag clear?
          BEQ CCHK2    ;Yes, go do it!
          PLA          ;No, get CONSOL
          LDA #0       ;back and return
          RTS          ;a zero!
;
CCHK2     PLA          ;A legal press!!
          RTS          ;Return CONSOL!
;
;------------------------
; CELL PLACEMENT HANDLER
;------------------------
;
;Initialize
;
PLACE
          LDX # <MSG0  ;Print instruc-
          LDY # >MSG0  ;tion message
          JSR PRINT
          LDA #127     ;Init cursor
          STA CURY     ;coordinates...
          LDA #120
          STA CURX
          JSR DRAWCUR  ;Draw cursor
          JSR INITCON  ;Set key flag
          LDX #$FF     ;Clear any stray
          STX CH       ;keypresses
          INX          ;Clear important
          STX SOUND    ;variables...
          STX PRESS
          STX SELECT
          STX DIST
          STX VBICTL   ;Set VB mode
;
;Main placement loop
;
CLRCLOK   LDA #0       ;Clear clock
          STA RTCLOK
          LDA SELECT   ;Decrement the
          BEQ HANGON   ;select key
          DEC SELECT   ;timer, if set.
;
HANGON    LDA RTCLOK   ;If 1/30 sec
          CMP #2       ;has elapsed,
          BCC NOTYET   ;handle the
          JSR MOVECUR  ;cursor and
          JMP CLRCLOK  ;reset the clock
;
NOTYET    LDA CH       ;If escape is
          CMP #28      ;pressed, call
          BNE NOESC    ;the cell clear/
          JSR CLEAR    ;reset routine
          LDA #$FF     ;And clear the
          STA CH       ;keyboard!
;
NOESC     JSR CHKCON   ;If no console
          BEQ HANGON   ;keys, go loop!
          ROR A        ;If start is
          BCS NOSTRT   ;pressed
          LDA #127     ;put the VB
          STA VBICTL   ;on hold,
          JSR QUIET    ;kil sound,
          JSR CLRMISS  ;clear cursor,
          RTS          ;and leave!
;
NOSTRT    ROR A        ;If select is
          BCC NEWMODE  ;pressed, handle
          LDA #0       ;it, otherwise
          STA SELECT   ;clear the sel-
          BEQ HANGON   ;ect key timer.
NEWMODE   LDA SELECT   ;If timer is
          BNE HANGON   ;set, then bye!
          LDA #10      ;A legal press!
          STA SELECT   ;Reset key timer
          LDX MODE     ;and add 1 to
          INX          ;the current
          CPX #5       ;mode making
          BNE MODEOK   ;sure it isn't
          LDX #0       ;greater than 4
MODEOK    STX MODE
          JSR SHOMODE  ;Show new mode
          JMP HANGON   ;and loop!
;
;Instruction message
;
MSG0      .SBYTE "  Use joystick to "
          .SBYTE "place cells on "
          .SBYTE "board"
          .BYTE EOL
;
;----------------
; CURSOR HANDLER
;----------------
;
MOVECUR
          JSR CHKTRIG  ;Check trigger
          LDA STICK0   ;If cursor is
          LDX DIST     ;moving, then
          BNE MOVING   ;go move it!
          CMP #15      ;If stick is
          BEQ XITMOVE  ;pushed, then
          STA DIR      ;start up the
          LDA #8       ;cursor and go
          STA DIST     ;slie it!
          BNE SLIDE
;
MOVING    LDX DIR      ;The cursor is
          CMP OPDIR-5,X ;moving!  If
          BNE SLIDE    ;the stick is
          LDA OPDIR-5,X ;being pulled
          STA DIR      ;in the direc-
          LDA #8       ;tion opposite
          SEC          ;of cursor
          SBC DIST     ;motion, reverse
          STA DIST     ;its direction!
;
SLIDE     LDX DIR      ;Calculate new
          LDA CURX     ;cursor x coord
          CLC          ;and make sure
          ADC XOFF-5,X ;it is legal...
          TAY
          CMP #64
          BCC HOLDIT
          CMP #185
          BCC DOY      ;Its legal!
;
HOLDIT    LDA #0       ;Stop all motion
          STA DIST     ;if a bad coord
XITMOVE   RTS          ;is found.
;
DOY       LDA CURY     ;Calculate new
          CLC          ;y coord and
          ADC YOFF-5,X ;make sure it
          CMP #71      ;is legal...
          BCC HOLDIT
          CMP #192
          BCS HOLDIT
          STA CURY     ;Save new y and
          STY CURX     ;new x, then
          DEC DIST     ;record move
          LDA #0       ;and clear the
          STA PRESS    ;trigger flag
          JMP DRAWCUR  ;Re-draw cursor!
;
;Movement data
;
XOFF      .BYTE 1,1,1,0,$FF,$FF,$FF
          .BYTE 0,0,0
YOFF      .BYTE 1,$FF,0,0,1,$FF,0,0
          .BYTE 1,$FF
OPDIR     .BYTE 10,9,11,0,6,5,7,0
          .BYTE 14,13
;
;-----------------
; TRIGGER HANDLER
;-----------------
;
CHKTRIG
          LDX TRIG0    ;If trigger is
          BEQ YESCHK   ;pressed, the
          DEX          ;trigger flag
          STX PRESS    ;is clear, and
NOCHK     RTS          ;the cursor is
YESCHK    LDA DIST     ;aligned on a
          BNE NOCHK    ;square, then
          LDA PRESS    ;process it
          BNE NOCHK
;
          LDX #1       ;Set trigger
          STX PRESS   ;flag
          STX GENER   ;Set generation
          DEX         ;to one
          STX GENER+1 ;Calculate scrn
          JSR CONVERT ;addr of cursor
          SED         ;Decimal mode!
          LDA (PNT1),Y ;If on a space
          BEQ ADDCELL ;add a cell
;
          LDA #0      ;Remove a cell!
          STA (PNT1),Y
          LDA POP     ;Subtract (bcd)
          SEC         ;one from the
          SBC #1      ;colony popula-
          STA POP     ;tion, max. pop.
          STA MOST    ;and starting
          LDA POP+1
          SBC #0
          STA POP+1
          STA MOST+1
          LDX #150    ;Go set proper
          BNE SETSND  ;ding sound
;
ADDCELL   LDA #10     ;Add a cell!
          STA (PNT1),Y
          LDA POP     ;Add 1 to the
          CLC         ;colony popula-
          ADC #1      ;tion, max. pop.
          STA POP     ;and starting
          STA MOST    ;pop...
          LDA POP+1
          ADC #0
          STA POP+1
          STA MOST+1
          LDX #30     ;Go set sound
;
SETSND    CLD         ;Decimal off!
          LDA #15     ;Set correct
          STA SOUND   ;add/remove
          STX TONE    ;sound
          JSR SHOGEN  ;Show updated
          JSR SHOPOP  ;cell data...
          JSR SHOSTRT
          JMP SHOMOST
;
;---------------------
; COORDINATE CONVERTER
;---------------------
;
;Convert cursor pm coordinates
;to a screen memory address
;
;Call with:
;  curx   - cursor pm x coord.
;  cury   - cursor pm y coord.
;
;Return:
;  pnt1   - base line address
;  y reg  - column offset
;
CONVERT
          LDA CURY    ;Convert pm y
          SEC         ;coord to board
          SBC #71     ;row number
          LSR A       ;(0 to 15)...
          LSR A
          LSR A
          PHA         ;Save the result
          STA PNT1    ;and put it into
          LDA #0      ;the work area
          STA PNT1+1
          LDX #3      ;Now multiply
MULT16    ASL PNT1    ;the column num
          ROL PNT1+1  ;by 20 to
          DEX         ;determine the
          BPL MULT16  ;offset from the
          PLA         ;beginning of
          ASL A       ;board memory.
          ASL A       ;This takes 2
          CLC         ;steps: 1st
          ADC PNT1    ;mult the column
          STA PNT1    ;by 16, then by
          LDA PNT1+1  ;4; now add the
          ADC #0      ;two results and
          STA PNT1+1  ;save the mess!
          LDA PNT1    ;Finally, add
          CLC         ;the offset to
          ADC # <[DISP+100] ;the start
          STA PNT1    ;of board memory
          LDA PNT1+1  ;for the base
          ADC # >[DISP+100] ;line
          STA PNT1+1  ;address! WHEW!
;
          LDA CURX    ;Convert the
          SEC         ;cursor pm x
          SBC #48     ;coord to the
          LSR A       ;column offset
          LSR A       ;from the base
          LSR A       ;line addres,
          TAY         ;and put it in
          RTS         ;the y reg!
;
;-------------------
; CLEAR/RESET CELLS
;-------------------
;
CLEAR
          LDA POP+1   ;If cell pop. is
          BNE CLEAROK ;0, then there
          LDA POP     ;is no reason to
          BEQ XITCLR  ;clear them!
;
CLEAROK   LDA # <BOARD ;If some
          STA PNT1    ;cells exist,
          LDA # >BOARD ;fill the
          STA PNT1+1  ;16x16 board
          LDX #15     ;area with
CLRROW    LDA #0      ;0s (spaces)...
          LDY #15
CLRCOL    STA (PNT1),Y
          DEY
          BPL CLRCOL
          LDA PNT1    ;Add 20 to cur-
          CLC         ;rent screen
          ADC #20     ;address to get
          STA PNT1    ;address of
          LDA PNT1+1  ;next line...
          ADC #0
          STA PNT1+1
          DEX
          BPL CLRROW
;
          JSR SETCELL ;Reset cell data
          JSR SHOGEN  ;Display new
          JSR SHOPOP  ;cell data...
          JSR SHOMOST
          JSR SHOSTRT
XITCLR    RTS         ;Return!
;
;-----------------
; CHECK FOR PAUSE
;-----------------
;
;Only the accumulator is used!
;
PAUSE     LDA CH      ;Space bar
          CMP #33     ;pressed?
          BNE NOPAUSE ;Nope, leave!
          LDA RTCLOK  ;Yes! Save
          PHA         ;clock value,
          JSR QUIET   ;shut up, and
          LDX #39     ;save whatever
TCOPY1    LDA TEXT,X  ;text is on
          STA TSAVE,X ;the message
          DEX         ;line...
          BPL TCOPY1
          LDX # <PAUSED ;and print a
          LDY # >PAUSED ;special
          JSR PRINT   ;message
;
PAUZ      LDA CH      ;Wait for
          CMP #33     ;a different
          BEQ PAUZ    ;key.
          LDX #39     ;When one is
TCOPY2    LDA TSAVE,X ;found, res-
          STA TEXT,X  ;tore the old
          DEX         ;message line...
          BPL TCOPY2
          PLA         ;Restore clock
          STA RTCLOK
NOPAUSE   RTS         ;And go back!
;
;Pause message
;
PAUSED    .SBYTE "                 "
          .SBYTE " Paused"
          .BYTE EOL
;
;---------------
; DELAY ROUTINE
;---------------
;
;Call with:
;  acc. - # of jiffies to wait
;
;Return a non-0 number if a
;console key aborts the delay
;else return a 0.
;
;Only the accumulator is used!
;
DELAY
          PHA         ;Clear RTCLOK...
          LDA #0
          STA RTCLOK
DELAY2    PLA         ;Wair until time
          CMP RTCLOK  ;is up, or until
          BEQ TIMESUP ;a console key
          PHA         ;is pressed
          JSR CHKCON
          BEQ DELAY2
          PLA
          RTS
TIMESUP   LDA #0
          RTS
;
;-------------------
; MAIN LIFE ROUTINE
;-------------------
;
LIFE
          JSR INITCON ;Init console
          LDA POP+1   ;If population
          BNE DOLIFE  ;is zero, then
          LDA POP     ;print a message
          BNE DOLIFE  ;saying so and
          LDX # <MSG1 ;exit...
          LDY # >MSG1
```

```
         JMP DOMSG
;
DOLIFE   LDX #$FF     ;Clear keyboard
         STX CH
         INX          ;Use the first
         STX CHORD    ;chord
NEXTGEN  JSR SCAN     ;Scan board
         LDA ACTIVE   ;If no cells are
         BNE UNSTAB   ;active then the
         LDX # <MSG5  ;population has
         LDY # >MSG5  ;stabilized!
         JMP DOMSG    ;(right Zink?)
UNSTAB   LDX # <MSG2  ;Unsatble, so
         LDY # >MSG2  ;print life
         JSR PRINT    ;message
         JSR GROWTH   ;Regenerate!
         BNE XITLIFE  ;If aborted, bye
         LDA POP+1    ;If new popu-
         BNE CONT     ;lation is 0,
         LDA POP      ;then the colony
         BNE CONT     ;is dead, so
         LDX # <MSG3  ;print a message
         LDY # >MSG3  ;and exit...
         JMP DOMSG
;
CONT     LDA MODE     ;If in FaST mode
         CMP #4       ;check for pause
         BNE NOTFST   ;or console key
         JSR PAUSE    ;then jump right
         JSR CHKCON   ;into the next
         BEQ NEXTGEN  ;generation
         RTS
;
NOTFST   CMP #2       ;If in Single
         BCC REGULAR  ;Step mode,
SSTEP    LDA TRIG0    ;check for a
         BEQ NEXTGEN  ;trigger press,
         LDX # <MSG4  ;print a message
         LDY # >MSG4  ;to the user,
         JSR PRINT    ;check console
         JSR CHKCON   ;keys, and
         BEQ SSTEP    ;repeat!
         RTS
;
REGULAR  LDA #0       ;If in ReGular
         STA RTCLOK   ;mode, wait 1/2
REGWAIT  LDA RTCLOK   ;a sec while
         CMP #30      ;checking for a
         BCS NEXTGEN  ;pause or a
         JSR PAUSE    ;console key...
         JSR CHKCON
         BEQ REGWAIT
         RTS
;
DOMSG    JSR PRINT    ;Print message,
         LDA #120     ;wait a while,
         JSR DELAY    ;and then
XITLIFE  RTS          ;exit life!
;
;Life related messages
;
MSG1     .SBYTE " No cells have bee"
         .SBYTE "n placed on the boa"
         .SBYTE "rd"
         .BYTE EOL
MSG2     .SBYTE "             Let the"
         .SBYTE "re be life!"
         .BYTE EOL
MSG3     .SBYTE "          The cell c"
         .SBYTE "olony has died"
         .BYTE EOL
MSG4     .SBYTE "   Press trigger f"
         .SBYTE "or next generation"
         .BYTE EOL
MSG5     .SBYTE "     Cell populati"
         .SBYTE "on has stabilized"
         .BYTE EOL
;
;-------------------
; PLAYFIELD SCANNER
;-------------------
;
;Return # of active cells in
;ACTIVE and information about
;these cells in CADRL, CADRH,
;and CTYPE
;
SCAN
         LDA #0       ;Clear cell
         STA ACTIVE   ;counter, and
         LDA # <[BOARD-21] ;move
         STA PNT1     ;board address
         LDA # >[BOARD-21] ;into
         STA PNT1+1   ;PNT1...
;
         LDA #15      ;Init row index
         STA ROW
DOROW    LDA #15      ;Init column
         STA COL      ;index
DOCOL    LDA #0       ;Clear counter
         STA COUNT
         LDX #7       ;Scan 8 times
SEARCH   LDY SCANADD,X ;Get offset
         LDA (PNT1),Y ;Is there a
         CMP #10      ;cell here?
         BNE NOCELL   ;No, ignore it!
         INC COUNT    ;Yes, add 1 to
NOCELL   DEX          ;the counter and
         BPL SEARCH   ;keep searching!
;
         LDX COUNT    ;If the current
         LDY #21      ;char is a space
         LDA (PNT1),Y ;with 3 friends
         BNE ACELL    ;then grow a
         CPX #3       ;cell here
         BNE NOACT
         BEQ RECORD
ACELL    CPX #2       ;If it is a cell
         BEQ NOACT    ;with 2 or 3
         CPX #3       ;friends, keep
         BEQ NOACT    ;it alive, else
```

```
RECORD   INC ACTIVE   ;kill it!
         LDX ACTIVE   ;Record any
         STA CTYPE,X  ;changes that
         LDA PNT1     ;are made to the
         CLC          ;cell colony...
         ADC #21      ;(the cell is 21
         STA CADRL,X  ;bytes ahead of
         LDA PNT1+1   ;the current
         ADC #0       ;address in
         STA CADRH,X  ;PNT1)
;
NOACT    LDA PNT1     ;Add 1 to PNT1
         CLC          ;so it points
         ADC #1       ;to the next
         STA PNT1     ;board square...
         LDA PNT1+1
         ADC #0
         STA PNT1+1
         DEC COL      ;Loop until 16
         BPL DOCOL    ;cols are done!
         LDA PNT1     ;Add 4 to PNT1
         CLC          ;so it points
         ADC #4       ;to the first
         STA PNT1     ;square in the
         LDA PNT1+1   ;next row...
         ADC #0
         STA PNT1+1
         DEC ROW      ;Loop until 16
         BPL DOROW    ;rows are done!
         RTS
;
;Scan data
;
SCANADD  .BYTE 0,1,2,20
         .BYTE 22,40,41,42
;
;----------------------
; GROWTH/DEATH HANDLER
;----------------------
;
;Put growth/death characters
;in proper places on board
;
GROWTH
         LDA MODE     ;If in FaST mode
         CMP #4       ;then skip the
         BNE DOFX     ;fancy animation
         JMP FSTMODE  ;and just do it!
DOFX     LDX #7       ;Init the
SETCHAR  LDA #0       ;growth/death
         STA CHSET+88,X ;chars...
         LDA CELIMG,X
         STA CHSET+96,X
         DEX
         BPL SETCHAR
;
         LDY #0       ;Find all of the
         LDX ACTIVE   ;board locations
PLOT1    LDA CADRL,X  ;where active
         STA PNT1     ;cells or spaces
         LDA CADRH,X  ;are, and place
         STA PNT1+1   ;growth or death
         LDA CTYPE,X  ;characters
         BNE DYING    ;there so we can
         LDA #11      ;animate them...
         BNE PLOTIT
DYING    LDA #12
PLOTIT   STA (PNT1),Y
         DEX
         BNE PLOT1
;
;Now animate the growing and
;dying cells along with sound
;effects if the current mode
;permits.
;
         LDX #7       ;Handle 8 frames
         STX TEMP
GROW     LDX TEMP     ;Get growth
         LDY GROIMG,X ;image # and
         LDX #0       ;copy the image
GROANIM  LDA IMAGES,Y ;into
         STA CHSET+88,X char 11...
         INY
         INX
         CPX #8
         BNE GROANIM
         LDX TEMP     ;Get death
         LDY DIEIMG,X ;image # and
         LDX #0       ;copy the image
DIEANIM  LDA IMAGES,Y ;into
         STA CHSET+96,X ;char 12...
         INY
         INX
         CPX #8
         BNE DIEANIM
;
         LDA MODE     ;If mode does
         BEQ DOCHORD  ;not allow sound
         CMP #2       ;then skip the
         BNE TIMEOUT  ;sound routine
DOCHORD  LDX CHORD    ;otherwise
         LDA NOTE1,X  ;generate the
         STA AUDF1    ;appropriate
         LDA NOTE2,X  ;3 note chord...
         STA AUDF2
         LDA NOTE3,X
         STA AUDF3
         LDA TEMP     ;Set volume and
         CLC          ;distortion
         ADC #$A1     ;(same for all 3
         STA AUDC1    ;channels)...
         STA AUDC2
         STA AUDC3
;
TIMEOUT  LDA #0       ;Now wait a
         STA RTCLOK   ;short while
SLEEP    LDA RTCLOK   ;Get clock
         CMP #4       ;Time up?
         BCS GROMORE  ;Yes!
         JSR PAUSE    ;No, check pause
```

```
         JSR CHKCON   ;and console key
         BEQ SLEEP    ;and loop!
         JSR QUIET    ;A console key
         LDA #1       ;is pressed so
         PHA          ;set return
         BNE REPLOT   ;value to 1
;
GROMORE  DEC TEMP     ;Do all frames
         BPL GROW     ;and when done
FSTMODE  LDA #0       ;set return
         PHA          ;value to 0
;
;Update board and cell data
;
REPLOT   LDY #0
         SED          ;Decimal mode!
PLOT2    LDX ACTIVE   ;Get the
         LDA CADRL,X  ;address of each
         STA PNT1     ;of the active
         LDA CADRH,X  ;cells and put
         STA PNT1+1   ;it into PNT1
         LDA CTYPE,X  ;Get cell type
         BNE DODIE    ;Cell is dying!
         LDA #10      ;Its growing so
         STA (PNT1),Y ;show a fully
         LDA POP      ;grown cell and
         CLC          ;add one to the
         ADC #1       ;cell population
         STA POP      ;(in decimal
         LDA POP+1    ;mode of course)
         ADC #0
         STA POP+1
         CMP MOST+1   ;If the cell pop
         BEQ CKMOST   ;is greater than
         BCC PLOTNXT  ;(or=to) the
CKMOST   LDA POP      ;highest pop so
         CMP MOST     ;far, then copy
         BCC PLOTNXT  ;POP into MOST..
NEWMOST  LDA POP
         STA MOST
         LDA POP+1
         STA MOST+1
         JMP PLOTNXT
DODIE    LDA #0       ;Its dying so
         STA (PNT1),Y ;plot a blank
         LDA POP      ;space and
         SEC          ;subtract 1 from
         SBC #1       ;the population
         STA POP
         LDA POP+1
         SBC #0
         STA POP+1
PLOTNXT  DEC ACTIVE   ;Loop til
         BNE PLOT2    ;ACTIVE is 0!
;
XITGRO   JSR QUIET    ;No sound!
         LDA GENER    ;Add 1 to
         CLC          ;generation (we
         ADC #1       ;are still in
         STA GENER    ;decimal mode)
         LDA GENER+1
         ADC #0
         STA GENER+1
         CLD          ;No decimal!
         LDX CHORD    ;Add 1 to chord
         INX          ;number making
         CPX #6       ;sure it is not
         BNE CHORDOK  ;greater than
         LDX #0       ;5...
CHORDOK  STX CHORD
         JSR SHOGEN   ;Show new cell
         JSR SHOPOP   ;data...
         JSR SHOMOST
         PLA          ;and return the
         RTS          ;correct value!
;
;Regeneration image data
;
GROIMG   .BYTE 0,8,16,24
         .BYTE 32,40,48,56
DIEIMG   .BYTE 56,48,40,32
         .BYTE 24,16,8,0
;
IMAGES   .BYTE 0,0,56,126
         .BYTE 126,60,24,0
;
         .BYTE 0,0,56,124
         .BYTE 124,56,24,0
;
         .BYTE 0,0,56,124
         .BYTE 124,24,0,0
;
         .BYTE 0,0,24,60
         .BYTE 60,24,0,0
;
         .BYTE 0,0,0,60
         .BYTE 60,24,0,0
;
         .BYTE 0,0,0,56
         .BYTE 56,24,0,0
;
         .BYTE 0,0,0,56
         .BYTE 56,0,0,0
;
         .BYTE 0,0,0,24
         .BYTE 24,0,0,0
;
;Chord data
;
NOTE1    .BYTE 243,230,217
         .BYTE 204,193,182
NOTE2    .BYTE 121,114,108
         .BYTE 102,96,91
NOTE3    .BYTE 60,57,53
         .BYTE 50,47,45
;
;---------------------
; INTRO & TITLE STUFF
;---------------------
;
;Do quickie intro
```

```
;
TITLE
         JSR INITCON ;Init console
         LDA #0      ;Clear colors...
         STA COLOR1
         STA COLOR2
         STA COLBAK
         LDA # <IDL  ;Point to intro
         LDX # >IDL  ;display list...
         STA SDLSTL
         STX SDLSTL+1
         LDA #60     ;Wait a sec...
         JSR DELAY
         BNE GOTOIT
         LDX #9      ;"Fade on" the
FADEON   INC COLOR1  ;magazine
         LDA #1      ;credits...
         JSR DELAY
         BNE GOTOIT
         DEX
         BPL FADEON
         LDA #120    ;Let it hang
         JSR DELAY   ;around for a
         BNE GOTOIT  ;sec, then
         LDX #9      ;fade to black..
FADEOFF  DEC COLOR1
         LDA #1
         JSR DELAY
         BNE GOTOIT
         DEX
         BPL FADEOFF
         LDA #60     ;Pause a little
         JSR DELAY   ;before showing
         BNE GOTOIT  ;the title
;
;Display title screen
;
         LDX #3      ;Set-up the
CLOAD    LDA TCOLORS,X ;proper
         STA COLOR0,X ;title screen
         DEX         ;colors...
         BPL CLOAD
         LDA # <TDL  ;Install the new
         LDX # >TDL  ;display list...
         STA SDLSTL
         STX SDLSTL+1
         LDA #127    ;Turn on cell
         STA VBICTL  ;animation
TWAIT    JSR CHKCON  ;and wait for
         BEQ TWAIT   ;a console key!
GOTOIT   RTS         ;AHHhhhhh.
;
;Intro display list
;
IDL      .BYTE 112,112,112
         .BYTE 112,112,112,112,112
         .BYTE 112,112,112,112,112
         .BYTE 112
         .BYTE 66
         .WORD ISCRN
         .BYTE 65
         .WORD IDL
;
;Intro screen data
;
ISCRN    .SBYTE "         Analog Com"
         .SBYTE "puting presents     "
         .SBYTE "    "
;
;Title screen display list
;
TDL      .BYTE 112,112,112
         .BYTE 112,112,112
         .BYTE 70
         .WORD TSCRN
         .BYTE 6,6,6,6,6
         .BYTE 112
         .BYTE 6
         .BYTE 112,112
         .BYTE 6,7
         .BYTE 112,112,112
         .BYTE 6
         .BYTE 65
         .WORD TDL
;
;Title screen data
;
TSCRN    .BYTE 0,0,10,0,0,0,0,10,0
         .BYTE 10,10,10,10,0,10,10,10
         .BYTE 10,0,0
;
         .BYTE 0,0,74,0,0,0,0,74,0,74
         .BYTE 0,0,0,0,74,0,0,0,0,0
;
         .BYTE 0,0,10,0,0,0,0,10,0,10
         .BYTE 10,10,0,0,10,10,10,0,0
         .BYTE 0
;
         .BYTE 0,0,74,0,0,0,0,74,0,74
         .BYTE 0,0,0,0,74,0,0,0,0,0
;
         .BYTE 0,0,10,0,0,0,0,10,0,10
         .BYTE 0,0,0,0,10,0,0,0,0,0
;
         .BYTE 0,0,74,74,74,74,0,74,0
         .BYTE 74,0,0,0,0,74,74,74,74
         .BYTE 0,0
;
         .SBYTE +128,"  IN THE FAST "
         .SBYTE +128,"LANE  "
         .SBYTE "     CREATED BY    "
         .SBYTE " "
         .SBYTE +128,"    JAMES  HAG"
         .SBYTE +128,"UE     "
         .SBYTE +64,"    PRESS  STAR"
         .SBYTE +64,"T     "
;
;Title screen colors
;
TCOLORS  .BYTE $46,$1A,$96
```

●

SALE • Famous National Brand • SAVE

NLQ 180

Hi-Speed Printer Sale

•160 - 180 CPS • Near Letter Quality • Lifetime Warranty*

NEW Price Break Thru

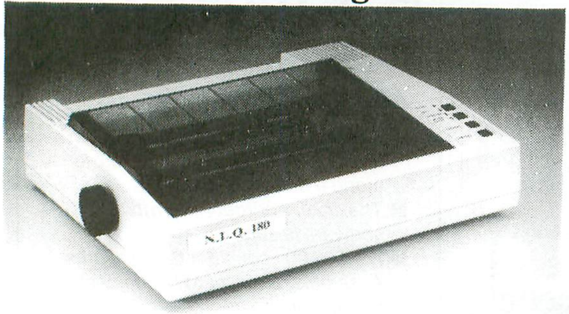
Sale $199.00

Below Wholesale Cost Prices!

List $499.95

60% OFF LIST PRICE

10'' Carriage



NLQ-180 Premium Quality Printer

Near Letter Quality Selectable From Front Panel Controls • High Speed Dot Matrix• Letter Quality Modes • 8K Buffer frees up computer 4-times faster • Pica, Elite, Italics, Condensed • Super Graphics • Business or Personal • Tractor/Friction • 15 Day Free Trial • Lifetime Warranty on Print Head* • 6 Month Immediate Replacement Policy •

NEW

All New up Front Panel Controls

Fantastic Graphics

Easy to Use

Fantastic Price

Sale

NLQ-180 Print Samples

```
This is an example of ITALICS
Enhanced        Boldface
 Condensed Text    Double-strike
example of Near Letter Quality
```

——— APPLE — ATARI — EPSON —— NLQ180 SPECIFICATIONS —— IBM — COMMODORE — ETC. ———

Print Buffer
8K bytes utility buffer

Printing Direction
Text Mode — Bi-directional
Graphic Mode — Uni-directional

Interface
Centronics Parallel Port

Paper
Plain paper, Roll paper, Single sheet
Fanfold, Multipart paper: max. 3 sheets (original plus 2 copies)

Character Fonts
Pica, Elite, Italics, Condensed

Printing Method
Impact dot matrix

Printing Speed
160-180 CPS at standard character printing

Printing Characters
Standard 9 x 9 dot matrix
NLQ 12 x 18 dot matrix (33cps)
Character size: 2.12 x 2.8 mm (standard)
Character sets: Full ASCII character set (96)
32 International characters

Ink Ribbon Cartridge
Ribbon Life: 3 million characters/cartridge

Physical Dimensions
Size: 15'' x 12'' x 5''
Weight: 12.7 lbs.

Maximum Number of Characters

Standard:	10 cpi	80 cpl
Standard enlarged:	5 cpi	40 cpl
Elite:	12 cpi	96 cpl
Elite enlarged:	6 cpi	48 cpl
Condensed:	17 cpi	132 cpl
Condensed enlarged:	8.5 cpi	66 cpl
Condensed elite:	20 cpi	160 cpl

INTERFACES

Atari $39.95 Apple II $44.95 Commodore $29.95 IBM $24.95 Laser 128 $19.95 Macintosh $49.95

Shipping, Handling & Insurance Charges

Add $10.00 for shipping, handling, and insurance. Illinois residents please add 6½% sales tax. Add $20.00 for ALASKA, CANADA, HAWAII, PUERTO RICO & APO-FPO orders. All orders must be in U.S. Dollars. WE DO NOT EXPORT TO OTHER COUNTRIES EXCEPT CANADA & PUERTO RICO. Enclose cashier check, money order or personal check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail. Prices & Availability subject to change without notice.

VISA — MASTER CARD — C.O.D. Call For C.O.D. Charges.

COMPUTER DIRECT

22292 N. Pepper Rd., Barrington, Illinois 60010

Call (312) 382-5050 or 382-5244
To Order

We Love Our Customers

CIRCLE #121 ON READER SERVICE CARD

10 MHz IBM® XT Compatible Computer System Sale

Complete System for Only

$^{\$}599^{00}$

COMPUTER DIRECT

22292 N. Pepper Road, Barrington, Ill. 60010
" The Computer Experts "

Call (312) 382-5050 or 382-5244 for
Free Catalogs of Over 1000 Programs & Accesories

Best Service • Best Price • 1000 Programs • 500 Accessories • 15 Day Free Trial

BEST GUARANTEE

FREE CATALOGS

10 MHz Super Turbo IBM® XT Compatible Computer System

Twice the speed* at just a fraction of the cost!

* Computes over two times faster than the IBM® XT

Look at all you get for only $^{\$}599^{00}$

This computer system is sold as a complete unit, NO SUBSTITUTIONS are allowed.

Run 1000's of IBM® software programs available.

15 Day Free Trial

90 Day Replacement Policy

Double 90 Day Policy
On Computer

The complete system	List Price	Sale Price
10 MHz Super Turbo XT Computer	$^{\$}1295^{00}$	$^{\$}499^{00}$
* 512K Memory	$^{\$}99^{95}$	No extra cost
* Single floppy disk drive	$^{\$}129^{95}$	No extra cost
* Parallel printer port	$^{\$}59^{95}$	No extra cost
* Serial printer port	$^{\$}59^{95}$	No extra cost
* Mouse/joystick port	$^{\$}59^{95}$	No extra cost
* RGB color graphics port	$^{\$}99^{95}$	No extra cost
* Hercules compatible monochrome port	$^{\$}79^{95}$	No extra cost
MS DOS 3.2 & GW Basic	$^{\$}199^{00}$	$^{\$}99^{00}$
12" Hi-Res 35 MHz Green Screen Monitor (TTL & EGA compatible)	$^{\$}249^{00}$	$^{\$}99^{00}$
Monitor interface cable	$^{\$}24^{95}$	$^{\$}19^{95}$
Big Blue Printer	$^{\$}199^{00}$	$^{\$}39^{95}$
RS 232 IBM to Big Blue cable	$^{\$}19^{95}$	$^{\$}9^{95}$
2 rolls of paper	$^{\$}19^{95}$	$^{\$}5^{95}$
Word First • Word Processor	$^{\$}99^{00}$	$^{\$}39^{95}$
Data First • Data Base	$^{\$}99^{00}$	$^{\$}39^{95}$
Calc First • Spreadsheet	$^{\$}99^{00}$	$^{\$}39^{95}$
Total price when bought separately	$^{\$}2893^{85}$	$^{\$}892^{65}$

* Built-in the Super Turbo XT

Home & Business

This IBM® XT compatible is perfect for your home and/or business uses. It makes life easier in more ways than you can imagine. Use the system for personal letters, form letters, address storage, listing valuables, figuring finances, school reports, business reports, calculations, business projections...the list can go on and on. With the addition of some of the thousands of software programs available for IBM® you can increase the capabilities of your system even further. A terrific home improvement, business enhancer, entertainment center & educational aid!

Save over $^{\$}275^{00}$ off sale prices!
Complete System only $^{\$}599$

IBM® is the trademark of International Business Machines Inc.

VISA - MASTERCARD - C.O.D. **Shipping, Handling & Insurance Charges and Information** VISA - MASTERCARD - C.O.D.

Add $35.00 for shipping, handling and insurance. Illinois residents please add 6½ % sales tax. Add $70.00 for CANADA, PUERTO RICO, HAWAII, ALASKA and APO-FPO. All orders must be in U.S. dollars. WE DO NOT EXPORT TO OTHER COUNTRIES EXCEPT CANADA & PUERTO RICO. Enclose Cashier Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail. Prices & Availability subject to change without notice. Please call for C.O.D. charges.

CALL COMPUTER DIRECT AT
(312) 382-5050 or 382-5244
8am - 8pm CST Weekdays / 9am - 12 noon Saturdays

MasterCard VISA

CIRCLE #121 ON READER SERVICE CARD

PROTECTO
ENTERPRIZES
"The Computer Experts"

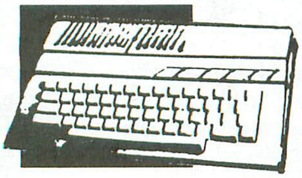
22292 N. Pepper Rd., Barrington IL 60010

• OUR WARRANTY •

Most of our products carry a minimum 90 day warranty from the date of purchase. If problems arise, simply send your product to us via U.P.S. prepaid. We will IMMEDIATELY send you a replacement at no charge via U.P.S. prepaid. This proves once again that...

We Love Our Customers!

ATARI 130 XE

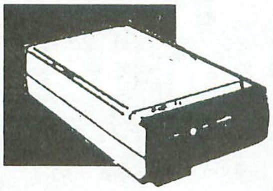


Includes 128K Ram & 24 K Rom

SALE $ 134.95

(Add $10.00 shipping) List $249

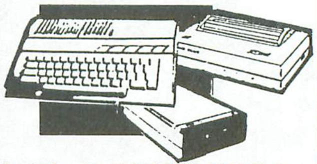
1050 127K DISK DRIVE



SALE $159.95

(Add $10.00 shipping) List $299

ATARI SYSTEM



Includes Atari 130 XE, 1050 Drive, Big Blue Printer, Interface, 2 Rolls of Paperand 2 Free Games.

SALE $ 299.95

(Add $25.00 shipping) List $857

APPLE IIc/IIe COMPATIBLE SYSTEM

Marvelous! Save over $700! Includes Laser 128, Big Blue Printer, Printer Interface, 12'' Monochrome Monitor, Magic Window IIe Word Processor, Magic Calc Spreadsheet, Magic Memory Database and 2 Rolls of Paper.

SALE $ 499.00

(Add $35.00 shipping) List $1228

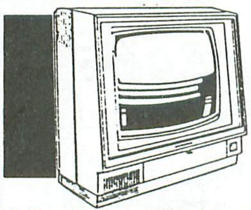
TV TUNER

Now switch your computer monitor into a television set. This tuner has a UHF/VHF/Computer selctor switch with easy front panel programmable selection buttons. A rabbit ear antenna for VHF viewing and adapters for cable or outdoor antenna make this fantastic. 75 ohm terminal with 300-75 ohm adapter.

SALE $ 79.95

(Add $3.00 shipping) List $130

13'' COLOR MONITOR

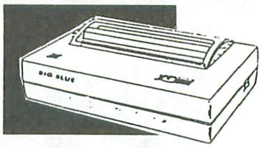


High Resolution, 1000 character display, with audio speaker. Perfect for your Atari Computer.

SALE $ 179.95

(Add $14.50 shipping) List $329

BIG BLUE PRINTER



This is the affordable printer you've waited for! 8½'' letter size, 80 column dot matrix, heat transfer printer features upper and lower case, underline, word processing, and much more.

SALE $ 39.95

(Add $7.50 shipping) List $199

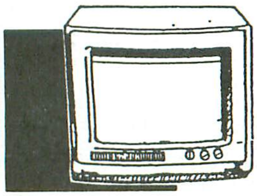
87 EP PRINTER & TYPEWRITER COMBINATION

Superb Silver Reed letter quality daisy wheel printer/typewriter, just a flick of the switch to interchange. Extra large carriage, typewriter keyboard, automatic margin control, compact, lightweight, drop in cassette ribbon! Centronics Parallel Interface port is built-in.

SALE $159.95

(Add $12.00 shipping) List $299

12'' 80 COLUMN MONITOR

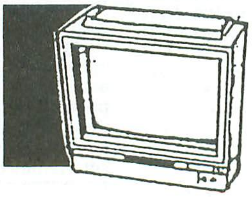


High Resolution green screen monitor. 80 col. x 1000 lines at center. Non-glare screen. Works great with your Atari, Apple, IBM or Laser 128 computer.

SALE $ 99.00

(Add $10.00 shipping) List $249

14'' RGB & COMPOSITE COLOR MONITOR

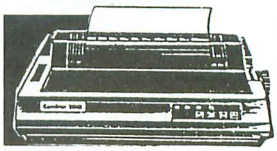


High Resolution, 80 column Monitor. Switch from RGB to Composite. (Atari - C128 - IBM Apple) RGB cable $19.95.

SALE $ 237.00

(Add $14.50 shipping)List $399

COMSTAR 1000 PRINTER



Print letters, documents, etc., at 100 cps. Works in Near Letter Quality mode. Features are dot addressable graphics, underline, tractor and friction feed, margin settings, pica, elite, condensed, italics, super/subscript, & more. Centronics Parallel Port

SALE $189.95

(Add $10.00 shipping) List $349

160-180 CPS N.L.Q. 180 PRINTER

This printer has a Near Letter Quality button on the front panel. No more turning the printer on and off. The 8K buffer will free up your computer four times faster than conventional printers and the high speed will keep you computing more than printing. Super graphics along with Pica, Elite, Italics, and Condensed print. **Lifetime Warranty** on Print Head plus 6 month immediate replacement policy. Centronics Parallel Interface port.

SALE $ 199.00

(Add $10.00 shipping) List $499

MasterCard

TO ORDER CALL (312) 382-5244

8 am - 8 pm CST Weekdays / 9 am - 12 noon CST Saturdays

VISA

CIRCLE #121 ON READER SERVICE CARD

BEST SERVICE IN THE USA • 1 DAY EXPRESS MAIL • OVER 500 PROGRAMS • VOLUME DISCOUNTS • 15 DAY FREE TRIAL • CUSTOMER LIST OF OVER 3,000,000 - LARGEST IN THE USA

FOR FREE CATALOG CALL

(312) 382-5244

PLEASE CALL TO ORDER: PRICES MAY BE LOWER & WE OFFER SPECIAL SYSTEM DEALS

DISK FILER

Dust and dirt can hurt your disks. Proper filing and protection of your disk collection will reduce unneccessary wear and tear. Over 50 disks fit in the filer.

SALE $12.95

List $24.95

JOYSTICK SALE

This Pro 5000 with microswitches has been rated No. 1 in joysticks. Arcade Quality!

SALE $16.95

List $24.95

COMPUTER CLEANERS

TV/Monitor Screen Restorer / Cleaner Kit • Disk Drive Cleaner • Anti-Static Keyboard Cleaner

*Choose any of these three computer cleaners for only $9.95 each!

SALE $9.95*

List $19.95

ATARI WRITER WORD PROCESSOR

Write papers, schoolwork, etc. quickly and efficiently with No. 1 Atari word processor w/spelling checker. Make changes with preview modes. Includes: full screen editing, double column printing, condensed, elongated, page no.'s headers-footers, margin set and more.

SALE $49.95

List $99.00

DOUBLE SIDED DOUBLE DENSITY DISK

100% Certified 5¼'' floppy disks. Lifetime Warranty. 1 Box of 100 -$29.00. List $1.99 each. **.29¢ ea.**

ATARI SOFTWARE

AMER. EDUC. COMPUTER

BIOLOGY FACTS (D)	$9.95
ELEM. SCIENCE FACTS (D)	9.95
U.S. GEOGRAPHY FACTS (D)	9.95
U.S. GOVERNMENT (D)	9.95
U.S. HISTORY FACTS (D)	9.95
WORLD GEOGRAPHY FACTS (D)	9.95
WORLD HISTORY FACTS (D)	9.95
FRENCH VOCAB. SKILLS (D)	9.95
SPANISH VOCAB. SKILLS (D)	9.95
AEC SPELLING (D)	19.95
LEARN TO READ (D)	19.95
READING COMPREHENSION (D)	19.95

BLUE CHIP

TYCOON (D)	$14.95
MILLIONAIRE (D)	14.95

DESIGNWARE

TRAP A ZOID (D)	$9.95
SPELLAGRAPH (D)	9.95
MATH MAZE (D)	14.95
CRYPTO CUBE (D)	12.95
SPELLICOPTER (D)	14.95
MISSION ALGEBRA (D)	24.95
SAT WORD ATTACK SKILLS (D)	22.95
PSAT WORD ATTACK SKILLS (D)	22.95
INTRO TO COUNTING (D)	22.95

FISHER PRICE

LINKING LOGIC (C)	$9.95
DANCE FANTASY (C)	9.95
MEMORY MANOR (C)	9.95
LOGIC LEVELS (C)	9.95

SPINNAKER

DELTA DRAWING (C)	$9.95
FACEMAKER (C)	9.95
AEROBICS (D)	6.95

SYNAPSE

SYNFILE (D)	$32.95
SYNCALC (D)	29.95
SYNCALC TEMPLATES (D)	14.95

WEEKLYREADER

PICTURE BUILDER (D)	$16.95
STICKYBEAR BOP (D)	16.95
STICKYBEAR NUMBERS (D)	16.95
STICKYBEAR BOUNCE (D)	16.95
STICKYBEAR OPPOSITES (D)	16.95
STICKYBEAR ABC (D)	16.95
STICKYBEAR SHAPES (D)	16.95

ACCESS

TRIPLE PACK (D)	$14.95
LEADER BOARD (D)	22.95

ACCOLADE

FIGHT NIGHT (D)	$18.95
HARDBALL (D)	18.95

ACTIVISION

PITFALL/DEMON ATTACK (D)	$12.95
HACKER (D)	12.95
GREAT AMER. ROAD RACE (D)	12.95
GHOSTBUSTERS (D)	12.95

BATTERIES INCLUDED

HOME PACK (D)	$19.95
PAPER CLIP (D)	29.95

BRODERBUND

PRINT SHOP (D)	$25.95
GRAPHICS LIB. 1,2 or 3 (D)	15.95
COMPANION (D)	22.95
KARATEKA (D)	18.95

COSMI

CAVERNS OF KHAFKA (D)	$10.95
TALLADEGA (D)	10.95
FORBIDDEN FOREST (D)	10.95
MASTERS OF TIME (D)	10.95
SUPER HUEY (D)	12.95
TEXT PRO/DATA PRO (D)	16.95

DATASOFT

PAC MAN (D)	$9.95
CONAN (D)	9.95
POLE POSITION (D)	9.95
DIG DUG (D)	9.95
MR. DO (D)	9.95
BRUCE LEE (D)	12.95
ZORRO (D)	9.95
GOONIES (D)	12.95
221 B BAKER STREET (D)	18.95

ELECTRONIC ARTS

AGE OF ADVENTURE (D)	$9.95
SUPER BOULDERDASH (D)	9.95
SEVEN CITIES OF GOLD (D)	9.95
ONE ON ONE (D)	9.95
PINBALL CONSTRUCTION (D)	9.95
MUSIC CONSTRUCTION (D)	9.95
RACING DESTRUCTION (D)	9.95
GOLDEN OLDIES (D)	19.95
MOVIE MAKER (D)	12.95
ROMMEL BATTLES TOBRUK (D)	26.95
STARFLEET 1 (D)	32.95
BATTLE OF CHICKAMAUGA (D)	23.95

EPYX

500XJ JOYSTICK (D)	$14.95
BALL BLAZER (D)	4.95
SUMMER GAMES (D)	14.95
RESCUE ON FRACTULUS (D)	4.95
WORLD KARATE (D)	18.95

HAYDEN

SARGON II (D)	$9.95
SAT VERBAL (D)	14.95
SAT MATH (D)	14.95
SAT PRACTICE TEST (D)	14.95

HI-TECH EXPRESSIONS

CARD WARE (D)	$7.95
HEART WARE (D)	7.95
PARTY WARE (D)	9.95
WARE WITH ALL KIT (D)	9.95
HOLIDAY PRINT PAPER (D)	7.95

INFOCOM

MOONMIST (D)	$22.95
BALLYHOO (D)	22.95
HITHHIKER'S GUIDE (D)	22.95
WISHBRINGER (D)	22.95

PEACH TREE

GENERAL LEDGER (D)	$37.95
ACCOUNTS RECEIVABLE (D)	37.95
ACCOUNTS PAYABLE (D)	37.95

S.S.I.

BATTLE OF ANTIETAM (D)	31.95
GETTYSBURG (D)	32.95
MECH BRIGADE (D)	34.95
NAM (D)	23.95
U.S.S.A.F. (D)	34.95
KAMPFGRUPPE (D)	32.95
FIELD OF FIRE (D)	23.95
CARRIER FORCE (D)	32.95
WAR SHIP (D)	3495
BROADSIDES (D)	23.95
COMPUTER AMBUSH (D)	32.95
BATTLE CRUISER (D)	34.95
COMPUTER QUARTERBACK (D)	22.95
COLONIAL CONQUEST (D)	23.95

XLENT

TYPESETTER (D)	$21.95
RUBBER STAMP (D)	18.95
PAGE DESIGNER (D)	18.95
P.S. INTERFACE (D)	18.95
1ST XLENT WORD PROCESSOR (D)	18.95
MEGA FONT (D)	16.95

Shipping, Handling & Insurance Information and Charges

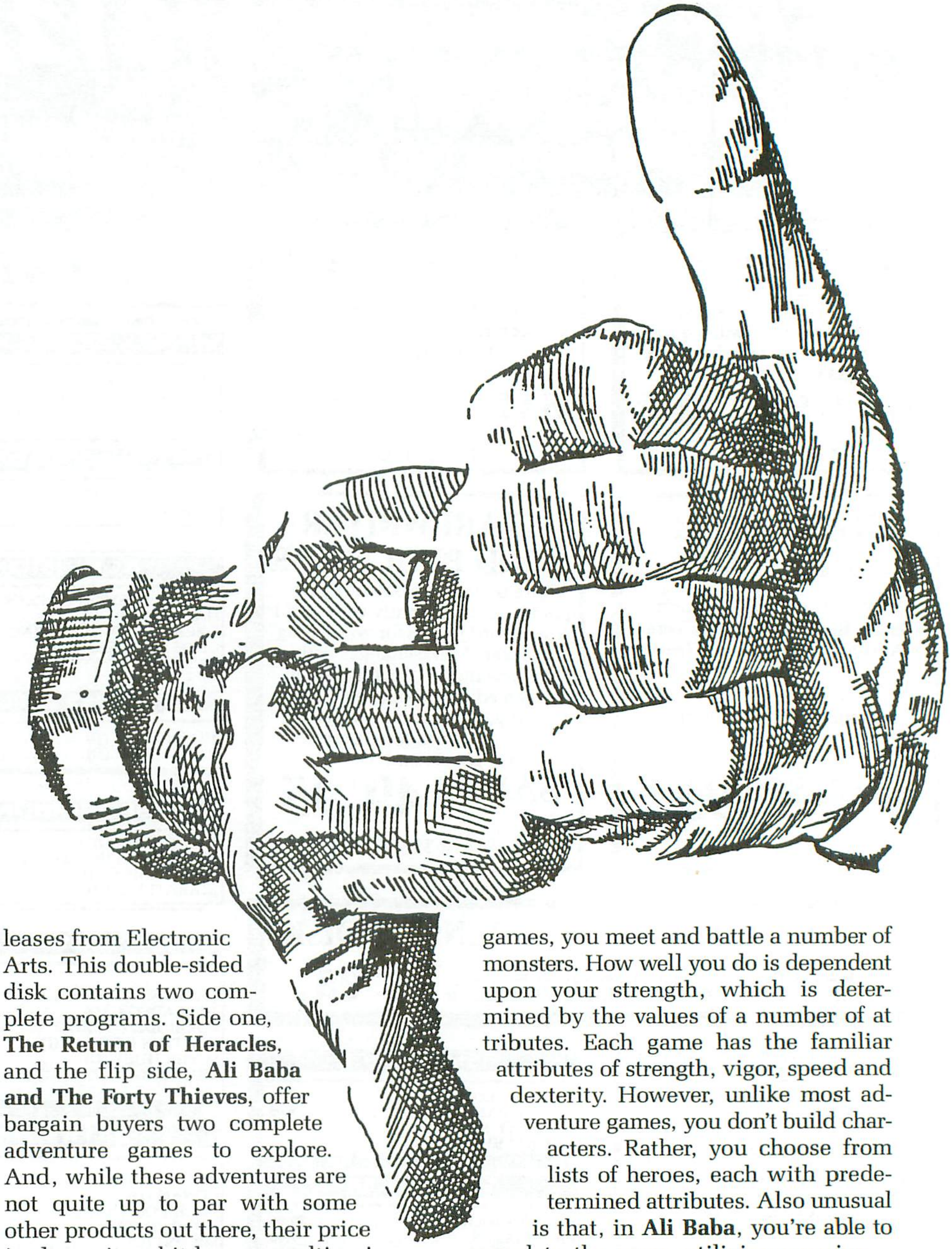
Add $3.00 (unless noted) for shipping, handling and insurance. Illinois residents please add 6½ % sales tax. Add $6.00 (double the amount) for CANADA, PUERTO RICO, HAWAII, ALASKA, APO-FPO orders. All orders must be in U.S. Dollars. WE DO NOT EXPORT TO OTHER COUNTRIES EXCEPT CANADA. Enclose Cashier Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail. Prices & Availability subject to change without notice. Hardware shipping prices vary according to weight. Please call for amount.

VISA - MASTERCARD - C.O.D. No APO-FPO for Monitors. Monitors can only be shipped to 48 contiguous United States. ICall for C.O.D. Charges.

CIRCLE #121 ON READER SERVICE CARD

Panak strikes!

Reviews of the latest software



by Steve Panak

It seems that the summer drought is upon us again. The few months before the summer Consumer Electronics Show are the most barren of the year. I would give anything to be back in December, with all its holiday releases. Then I look out the window and wonder. Perhaps there *is* something better to do than play these games day in and day out. Maybe there's something to be gained by running on the freshly cut grass, breathing the clean, spring air. Maybe there *are* worlds outside of my throbbing CPU.

Then I come to my senses and release the PAUSE key.

Age of Adventure

by Stuart Smith
ELECTRONIC ARTS
1820 Gateway Drive
San Mateo, CA 94404
48K Disk $14.95

Into a world where the software prices of major manufacturers are typically *out* of this world, Electronic Arts has decided to unleash a package which not only has a below-the-average suggested retail price, but also contains *two* complete adventure games. Even if these games are simply fair, this marks the beginning of a new trend—a trend I, and probably the rest of you, can live with.

Age of Adventure is one of the latest releases from Electronic Arts. This double-sided disk contains two complete programs. Side one, **The Return of Heracles**, and the flip side, **Ali Baba and The Forty Thieves**, offer bargain buyers two complete adventure games to explore. And, while these adventures are not quite up to par with some other products out there, their price is also quite a bit lower, resulting in much more gaming for the dollar.

The plots and goals of both these games are simple. In **Ali Baba**, it is your task to rescue the princess Buddir al-Buddoor, who has been kidnapped (or, rather, princess-napped) and secreted in a mountain stronghold. You must brave the confusing caverns to rescue her. In **Heracles**, you repeatedly visit the oracle, who gives you tasks. Typically, you're required to kill—or otherwise injure—some other being. Not very original, but rather entertaining. The oracle also provides clues (unreferenced to any specific task), which get better as you increase your contributions to the priests.

As occurs in most fantasy-adventure games, you meet and battle a number of monsters. How well you do is dependent upon your strength, which is determined by the values of a number of at tributes. Each game has the familiar attributes of strength, vigor, speed and dexterity. However, unlike most adventure games, you don't build characters. Rather, you choose from lists of heroes, each with predetermined attributes. Also unusual is that, in **Ali Baba**, you're able to complete the game utilizing cunning—with no fighting or violence. It may be a little (or maybe a lot) harder, but it is possible for pacifists to prosper in Ali Baba.

Reincarnation is also a possibility in **Ali Baba**, while you get only one life in **Heracles**. The latter has the additional benefit of teaching players a bit about ancient Greek mythology. Both games allow you to control more than one adventurer at a time, each becoming active in rotation. You may assemble your group of adventurers into a party—although, in **Heracles**, you must first unite your party, as the players start in different rooms. **Ali Baba** automatically places the heroes together. In **Heracles**, you can permit one of your members to desert. He will then

Software Discounters of America

S.D. of A.

For Orders Only—1-800-225-7638
PA Orders—1-800-223-7784
Customer Service 412-361-5291

- Free shipping on orders over $100 in continental USA
- No surcharge for VISA/MasterCard
- Your card is not charged until we ship

ABACUS BOOKS	
ST Gem Prog. Ref.	
ST Graphics & Sound	CALL
ST Internals	FOR
ST Machine Language	LOW PRICES
ST Midi Programming	
ST Peeks & Pokes	
ST 3-D Graphics	
ST Tricks & Tips	
ABACUS SOFTWARE	
Assem/Pro ST	$39
Data Trieve ST	$33
Forth/MT ST	$33
Paint Pro ST	$33
Text Pro ST	$33
ACADEMY	
Typing Tutor ST	$23
ACCESS	
Leader Board Golf (D)	$25
Leader Board Golf ST	$25
L.B. Tourn. Disk #1 ST	$14
L.B. Tourn. Disk #1 (D)	$14
Tenth Frame ST	$25
Triple Pack: BH1, BH2, Raid Over Moscow (D)	$14
ACCOLADE	
Fight Night (D)	$19
Hardball (D)	$19
Hardball ST	$29
Mean 18 Golf ST	$25
Mean 18 Famous Course Disk ST	$14
ACTIVISION	
Great American Cross Country Road Race (D)	$16
Hacker (D)	$16
Hacker 2 ST	$33
Music Studio ST	$39
Music Studio (D)	$23
Paintworks ST	$44
Portal ST	$33
Shanghai ST	$29
Tass Times in Tonetown ST	$33
AMERICAN EDUCATIONAL COMPUTER	
Biology (D)	$14
French (D)	$14
Grammar (D)	$14
Science: Grades 3/4 (D)	$14
Science: Grades 5/6 (D)	$14
Science: Grades 7/8 (D)	$14
Spanish (D)	$14
ARTWORX	
Bridge 4.0 (D)	$16
Bridge 4.0 ST	$19
Bridge 5.0 ST	$23
Hole in One Golf ST	$14
Mail List ST	$14
Strip Poker (D)	$21
Strip Poker ST	$25
Female Data Disk 1	$14
Male Data Disk 2	$14
Female Data Disk 3	$14
AVANTAGE	
Spy vs. Spy 1 & 2 (D)	$9.88
BATTERIES INCLUDED	
Degas ST	$23
Degas Elite ST	$49
Isgur Portfolio System ST	$129
Paperclip (D)	$36
Paperclip w/Spell 130XE	$36
Thunder ST	$25
BRODERBUND	
Champ. Loderunner (D)	$19
Karateka (D)	$19

Loderunner (D)	$23
Print Shop (D)	$26
Print Shop Graphics Library #1, #2, #3 (D)	$16 Ea.
P.S. Companion 64K (D)	$23
CENTRAL POINT	
Copy 2 ST	$25
CDA	
America Cooks Series:	
American ST	$9.88
Chinese ST	$9.88
French ST	$9.88
Italian ST	$9.88
Mexican ST	$9.88
DATASOFT	
Alternate Reality: The City ST	$23
Mercenary ST	$23
DAVIDSON	
Math Blaster (D)	$33
Spell It (D)	$33
Word Attack (D)	$33

Skyfox ST	$29
Star Fleet I (D)	$33
Star Fleet 1 ST	$33
EPYX	
Ballblazer (D)	$9.88
Champ. Wrestling ST	$25
Dragonriders Pern (D)	$9.88
Koronis Rift (D)	$9.88
Rescue on Fractalus (D)	$9.88
Rogue ST	$23
Sub Battle Simulator ST	$25
Super Cycle ST	$23
Temple Apshai Trilogy (D)	$14
Temple Apshai Trilogy ST	$14
The Eidolon (D)	$9.88
Winter Games ST	$23
World Champ. Karate ST	$25
World Games ST	$25
FIREBIRD	
Golden Path ST	$29
Guild of Thieves ST	$29
The Pawn (D)	$25
The Pawn ST	$29

Leather Goddesses (D)	$23
Trinity ST	$25
Wishbringer (D)	$23
Zork 1 (D)	$23
Zork Trilogy (D)	$39
** All titles in stock for 520 ST—Call for prices*	
MASTERTRONIC	
Action Biker (D)	$6.88
Elektraglide (D	$6.88
Magnum Joystick	$12
Ninja (D)	$6.88
Ninja ST	$14
Renegade ST	$14
Speed King (D)	$6.88
The Last V8 (D)	$6.88
Vegas Poker & Jackpot (D)	$6.88
MICHTRON	
BBS 2.0 ST	$49
Calendar ST	$19
Corner Man ST	$33
GFA Basic Int. ST	$49

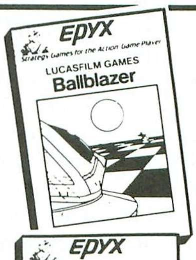
MINDSCAPE	
Balance of Power ST	$33
Bop & Wrestle (D)	$19
Brattacus ST	$29
Crossword Magic (D)	$33
High Roller ST	$33
Infiltrator (D)	$19
Trail Blazer (D)	$19
MINDSCAPE CINEMAWARE	
Defender of the Crown ST	Call
S.D.I. ST	$33
Sinbad: Throne of the Falcon ST	Call
The King of Chicago ST	Call
ORIGIN	
Autoduel (D)	$33
Autoduel ST	$33
Ogre (D)	$26
Ogre ST	$26
Ultima 3 ST	$39
Ultima 3 (D)	$32
Ultima 4 (D)	$39

Great Fantasy Role-Playing—You're a Techno-Scavenger Salvaging Hi-Tech Weapons from the Ancients!

MONTHLY MADNESS SALE

Your Choice

9.88 Each

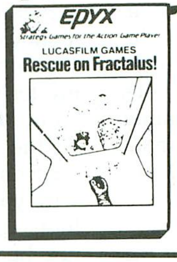


Realistic, Three-Dimensional Depth Perception

Great Fantasy Adventure—The Eidolon™ Transports You Into a Magical Realm

Special Bonus: Buy any 3, get the 4th *FREE*

Available for: Atari XL/XE (disk)



Realistic, Three-Dimensional Landscape

ELECTRONIC ARTS	
Software Classic Series:	
Age of Adventure (D)	$9.88
Archon (D)	$9.88
Archon 2: Adept (D)	$9.88
Financial Cookbook (D)	$9.88
Mail Order Monsters (D)	$9.88
Movie Maker (D)	$9.88
M.U.L.E. (D)	$9.88
Music Const. Set (D)	$9.88
One-on-One (D)	$9.88
Pinball Const. Set (D)	$9.88
Racing Dest. Set (D)	$9.88
Seven Cities of Gold (D)	$9.88
Super Boulder Dash (D)	$9.88
Touchdown Football (D)	$9.88
ELECTRONIC ARTS	
Arcticfox ST	$29
Chessmaster 2000 (D)	$26
Chessmaster 2000 ST	$29
Financial Cookbook ST	$33
Lords of Conquest (D)	$23
New Tech Coloring Book ST	$14
Rommel Battles for Tobruk (D)	$26

Star Glider ST	$29
FIRST BYTE	
First Shapes ST	$33
Kid Talk ST	$33
Math Talk ST	$33
Speller Bee ST	$33
**These programs talk!*	
FTL	
Dungeonmaster ST	$25
R.P.V. ST	$25
Sundog ST	$25
GAMESTAR	
Champ. Baseball ST	$29
Champ. Basketball ST	$29
Champ. Football ST	$29
HI-TECH EXPRESSIONS	
Award Ware (D)	$9.88
Card Ware (D)	$6.88
Heart Ware (D)	$6.88
Party Ware (D)	$9.88
INFOCOM	
Ballyhoo (D)	$23
Bureaucracy (D)	$23
Hitchhiker's Guide (D)	$23
Hollywood Hijinx (D)	$23

GFA Basic Compiler ST	$49
Karate Kid II ST	$25
Major Motion ST	$25
Match Point ST	$25
M-Disk Plus ST	$25
Mighty Mail ST	$33
Personal Money Mgr. ST	$33
Pinball Factory ST	$25
Space Shuttle II ST	$25
Time Bandit ST	$25
Trim Base ST	$59
MICROLEAGUE	
Baseball (D)	$25
Box Score Stats (D)	$16
General Manager (D)	$25
1985 Team Data Disk (D)	$14
1986 Team Data Disk	$14
WWF Wrestling ST	Call
MICROPROSE	
F15 Strike Eagle (D)	$23
Kennedy Approach (D)	$16
Silent Service (D)	$23
Silent Service ST	$25
Top Gunner (D)	$16
MILES COMPUTING	
Harrier Strike Mission ST	$33

OSS	
Action (R)	$47
Action Tool Kit (D)	$19
Basic XE (R)	$47
Basic XL (R)	$37
Basic XL Tool Kit (D)	$19
Personal Pascal ST	$49
PCA	
Graphic Artist ST	$129
Font Editor ST	$49
PROGRESSIVE	
Logistix Jr. ST	$59
Logistix 1040 ST	$95
Superbase ST	$95
SHELBOURNE	
Pool ST	$19
Shuffleboard ST	$19
SIERRA	
Black Cauldron ST	$25
Donald Duck's Playground ST	$16
Kings Quest 1, 2, or 3 ST	$33 Ea.
Space Quest ST	$33
Winnie the Pooh ST	$16

SOFTLOGIC	
Publishing Partner ST	$95
SOLID GOLD	
Pitfall/Demon Attack (D)	$9.88
SPECTRUM HOLOBYTE	
Gato ST	$23
SPRINGBOARD	
Certificate Maker ST	$33
SSI	
Gettysburg (D)	$37
Phantasie (D)	$25
Phantasie ST	$25
Phantasie 2 ST	$25
Phantasie 3 ST	Call
Rings of Zilfin ST	Call
Roadwar 2000 ST	Call
War Game Const. Set (D)	$19
Wizard's Crown (D)	$25
Wizard's Crown ST	Call
SUBLOGIC	
Flight Simulator 2 (D)	$32
Flight Simulator 2 ST	$33
F.S. Scenery Disks	Call
Jet ST	Call
SYNAPSE	
Syn-Calc (D)	$33
Syn-File (D)	$33
TDI	
Modula 2 ST	$49
Modula 2 (Dev.) ST	$95
TIMEWORKS	
Data Manager ST	$49
Swiftcalc ST	$49
Word Writer ST	$49
TRONIX	
S.A.M. (D)	$19
UNICORN	
Animal Kingdom ST	$25
Decimal Dungeon ST	$25
Fraction Action ST	$25
Kinderama ST	$25
Math Wizard ST	$25
Read & Rhyme ST	$25
UNISON WORLD	
Art Gallery 1 ST	$19
Art Gallery 2 ST	$19
Print Master ST	$25
VALUE TIME	
Calendars & Stationery ST	$9.88
Greeting Cards ST	$9.88
Signs & Banners ST	$9.88
Art Library 1 or 2 ST	$9.88 Ea.
VIP TECHNOLOGIES	
VIP Professional ST	$149
XLENT	
First Xlent W.P. (D)	$19
Page Designer (D)	$19
P.S. Interface (D)	$19
Rubber Stamp (D)	$19
Rubber Stamp ST	$25
Typesetter (D)	$23
Typesetter Elite ST	$33
Write 90° ST	$19
ACCESSORIES	
Anchor VM 520 300/1200 Baud Modem ST	$129
Bonus 5¼ SS, DD	$5.99 Bx
Bonus 5¼ DS, DD	$6.99 Bx
Compuserve Starter Kit	$23
Disk Case (40-3½)	$9
Disk Case (75-5¼)	$9
Disk Drive Cleaner-3½	$9
Epyx 500XJ Joystick	$14
Wico Boss	$12
Wico Bat Handle	$17

P.O. BOX 111327—DEPT. AN—BLAWNOX, PA 15238

*Please Read The Following Ordering Terms & Conditions Carefully Before Placing Your Order: Orders with cashiers check or money order shipped immediately on in stock items. Personal & Company checks, allow 3 weeks clearance. No C.O.D.'s! *Shipping: Continental U.S.A.*—Orders under $100 add $3; *free shipping on orders over $100. AK, HI, FPO, APO—add $5 on all orders. Canada & Puerto Rico—add $10 on all orders.* Sorry, no other International orders accepted! *PA residents add 6% sales tax on the total amount of order including shipping charges! REASONS FOR CALLING CUSTOMER SERVICE—412-361-5291 (1)* Status of order or back order *(2)* If any merchandise purchased within 60 days from S.D. of A. is defective, please call for a return authorization number. Defective merchandise will be replaced with the same merchandise only. Other returns subject to a 15% restocking charge. After 60 days please refer to the manufacturers warranty included with the merchandise & return directly to the manufacturer. *Customer service will not accept collect calls or calls on S.D. OF A.'s 800# order lines!* Have you seen our on line catalog of 1000 software titles for Commodore, Atari, Apple, IBM and Amiga? It's on Compuserve's Electronic Mall—just type GO SDA and shopping for software will never be the same again! *HOURS: Mon.-Fri. 9AM-5:30 PM, Sat. 10AM-4PM Eastern Time. Because this ad had to be written 2-3 mos. before it was published, prices & availability are subject to change!*

CIRCLE #122 ON READER SERVICE CARD

venture out on his own, controlled by the computer.

As each side of the floppy contains one of the programs, you'll need a blank disk if you want to save any games. Both offerings, while different in content, have similar control interfaces—so you won't be burdened by learning two completely different systems. You choose to use either the joystick (preferred) or the keyboard. Movement is controlled by the compass direction on the stick, or by the cursor control keys. Other options (such as attacking, defending and resting) are accessed through a series of menus reached by means of the joystick button. In **Ali Baba**, another option menu controls various program attributes—sound on or off, difficulty and save game. In **Heracles**, these options are accessed from a menu tied to the OPTION key. For such bargain games, the number of options available is quite unexpected. Finally, although only one person can play, two players could agree to each control a given character, then take turns with the stick.

The graphics in each game are acceptable, although not breathtaking; they're merely competent images which adequately represent the games' content. In contrast, the documentation is extensive, even for programs at more than twice the cost. A 24-page manual fully describes game content and program control. It's broken into two sections, each dealing with one of the games. An additional reference card gives machine-specific information. It's all in the familiar Electronic Arts record-album-like cover.

So, while **Age of Adventure** is not the greatest game I've ever seen, not the most detailed game I've ever played, and hasn't got the best graphics ever to burn my retinas, it *is* an exceptional value. Its two complete games are engaging, and its price is certainly right. **Age of Adventure** is an adventure bargain.

Hollywood Hijinx

by "Hollywood" Dave Anderson
INFOCOM
125 CambridgePark Drive
Cambridge, MA 02140
48K Disk $34.95

It was only a matter of time until the lunatics at Infocom went Hollywood. Every one of their many works of interactive fiction has made you the main character in the story, a story whose author is also you, the player. It seems natural that they now place you in a movie.

Hollywood Hijinx is the latest release from Infocom. It is also the first work from author Dave Anderson, and its glitzy and subtle sense of humor makes me hope it won't be his last. The premise of this game is as hokey as the plots of the movies it parodies. Your rich uncle Buddy, a Hollywood mogul, has been dead for years now. But his wife, Aunt Hildegarde, just died—and guess what? That's right, you've been named in the will as the sole beneficiary of their massive estate. But, like all respectable Hollywood contracts, there's a catch.

This catch forms the basis of the game —a scavenger hunt. To claim the estate, you must find ten treasures from some of Uncle Buddy's greatest movies. These memorabilia have been cached throughout the mansion and its spacious grounds. To make matters worse, you have but one night to locate it all. As the story begins, you're dumped outside the sprawling estate, aptly named Hildebud. After walking past the weapon-laden statue of Buck Palace, fighting mailman (one of the many corny characters you'll encounter from your uncle's chain of B movies), you must try to get into the house. This is the first in a chain of riddles and puzzles which will keep you entertained for weeks.

Assuming you get into the house (a tenuous assumption, at best), you'll have a great time exploring the enormous and mysterious mansion. Memorabilia from many of your uncle's worst movies clutter the many rooms, and your task is to determine which ten pieces of junk will win you the estate. Unfortunately, the footsteps you hear indicate that you may not be the only one interested in the booty.

So you'll hurry to explore the house's various nooks and crannies. You'll move from the private theater where Uncle Buddy screened his latest films to the study in which he softened up potential investors with a little liquid confidence, to the guest room with the brass handles on the floor—so those same investors would have something to hold onto when the room began to spin. And I won't even *mention* the massive hedge maze you'll have to chart out.

While this is a standard-level game, I found the puzzles in it difficult. This is due to the way the thing lulled me into false assumptions and fed me red herrings. Clues abound, but the puzzles are a step above the norm: most challenging, requiring careful attention to the prose, the settings and the documentation. Also, due to the content of some clues, I found the printer more of an asset than in any other Infocom game I've played.

Speaking of documentation, I've come to expect the best in packaging and documentation from Infocom, and I wasn't disappointed. Inside the booklike box is a copy of the gossip rag "Tinselworld," followed by a clear, well written manual. It would be hard not to learn to play this game after simply scanning its illustrative instructions. "Tinselworld" was very entertaining to read, a humorous parody of grocery-store tabloids, featuring articles that fill you in on your relatives' rather rousing past. Also included is a copy of Aunt Hildegarde's will, an autographed publicity photo of Uncle Buddy and a lucky palm tree swizzle stick.

As far as program performance and design go, **Hijinx** has utilized the basic Infocom program—which means flawless execution and a vast vocabulary. I was disappointed that I couldn't use X as an abbreviation for "Examine," as I could in **Moonmist**. I hope that shortcut will become a standard Infocom ingredient.

Despite this slight flaw, **Hollywood Hijinx** is a fine addition to the growing list of Infocom products. Like the city and movies it parodies, beneath its thin veneer lies a complete absence of rational thought —a space filled with only mindless entertainment. And you can't help but be entertained.

The author wishes to thank the Magic One Computer Shop of Barberton, Ohio for their valuable assistance in the creation of this chronicle.

Database Delphi

News and updates from the *ANALOG Computing* Atari Users' Group on Delphi

by Matthew J.W. Ratcliff

From: CANTTHINK (Jim Ellwanger)
To: MATRAT (Matthew J.W. Ratcliff)

I downloaded your program **Tablet Typist** from the Atari SIG, and followed the directions in **ANALOG Computing** 51. I ran Listing 2, and it worked okay. However, when I ran Listing 1, it told me: *NO TYPE.SYS FILE.* Do you have a correction for this? I am using SpartaDOS 3.2d. Is the program only for DOS 2?

From: MATRAT
To: CANTTHINK

Unfortunately, the program was written long before I was using the new SpartaDOS.

If programs are written carefully, they can be compatible with DOS 2.0s and SpartaDOS at the same time. The real glitch that caught me was the status you receive after reading a disk file. Atari and OSS DOS XL both report a status of *1* after a successful disk read, and a value of *3* if the completed disk read was also the *last* byte in the file you're reading. I used this often to make sure that some files are *exactly* the size expected by the program. It's an extra level of error checking.

However, SpartaDOS returns a value of *1* in *either case*. To make **Tablet Typist** Sparta compatible, you need only change several error-trapping lines in Listing 1:

```
940 GOSUB 130:IF ERR<=C3 T
HEN 960
1340 GOSUB 130:IF ERR>C3 T
HEN CST$="":? "* Bad Font
File *":GOTO 1300
1740 IO=C2:AD=C256*C5N:NM=
1024:CMND=CGET:GOSUB 130:I
F ERR>C3 THEN 1760
1830 IO=C2:AD=ADR(MOV$):NM
=76:CMND=CGET:GOSUB 130:IF
 ERR>C3 THEN 1800
```

CANTTHINK's TYPE.SYS file was indeed on the disk, but, because SpartaDOS didn't return the completion status of *3* expected, **Tablet Typist** assumed an error condition. The above modifications will make the **Tablet Typist** SpartaDOS compatible, and still provide sufficient error trapping in the program.

New in the databases.

In the Sound and Graphics database, you'll find a digitized graphics and sound demo called PEEWEE DEMO. The 32K file was sent up by ATARICOP (John Mackintosh). When you get bored with PeeWee's multiple digitized faces (a good ComputerEyes picture), you can press START to hear him sing "connect the dots," digitally. This is a brief, but great, sound and graphics demo (especially if you love PeeWee Herman as I do).

ATARICOP has also sent up a graphics demo which uses multiple DLIs. He promises you won't be sorry for downloading OPART.BAS.

User 25BNH (Brian Hershey) has sent up the TURBO FRACTAL GENERATOR. It's a full-featured fractal program in compiled Turbo BASIC. It's touted as being fifteen times faster than Atari BASIC, and can perform 62-sector screen dumps to disk or to Epson/Gemini printers, plus it has magnification and color cycling.

In the Utilities database, you'll find ARC—8-bit. This package lets you compress and decompress files in the popular ARC format. We're now using it for ST and 8-bit programs, to save time downloading. These files, sent up by AJQ (Aaron Quantz), include ARC.COM (a machine language COMPRESSOR utility), ARCX.COM (extractor, decompressor) and ARC.DOC (documentation).

You will definitely need to download these files, and learn how to use them, as more and more of our files are compressed on the system. With ARC, not only is a file squeezed down to a smaller size, but multiple files can be libraried together in a single file. This minimizes the number of XM commands (and related time to set up for it) you must issue when acquiring a package of related files.

Delphi tips.

I've been asked about the REQUEST feature on Delphi for free *upload* time. When you enter the REQUEST command at the *ANALOG>* prompt, you're given a *REQUEST>Which Area?* You would *think* that you should enter a database topic name here. You don't. If you enter anything but the word *REQUEST* again, you won't get far. After the second *REQUEST* prompt comes up, you can READ information to prompt you through your free-time query.

In the March (issue 52) **Database Delphi**, I presented a brief program listing to print ASCII documentation files for you. Well, brief as it was, it had a minor glitch.

Database Delphi *continued*

Most of you probably caught it. Simply add this line: *65 GOTO 30.*

In that same issue, I also made a typographical error with the SEND command. The SEND should *not* be preceded with a slash (/) at the *FORUM* or *ANALOG* prompts. It is only needed while in the conference mode.

New faces and conferences.

I mentioned previously that ICDINC is now on Delphi. Generally, Tom Harker, President of ICD, uses the account. Keith Ledbetter, author of the phenomenal 8-bit **Express!** programs, now works for ICD. He will be on-line occasionally.

By the time you read this, we will have had a conference with Infocom's Brian Moriarity, author of **Wishbringer** and **Trinity**. There will be a transcript in Reviews and News.

SCHEDULED ATARI FAIRS

JUNE 19 & 20, 1987
SAN JOSE, CALIFORNIA
Santa Clara Convention Center
Great America Parkway
For info, call Atari at (408) 745-2000

JULY 25 & 26, 1987
CHICAGO, ILLINOIS
Ramada Inn Hotel, O'Hare
At Manheim and Higgins Roads

UPCOMING SHOWS

Additional information will be published in later issues.

AUGUST 1987
DETROIT, MICHIGAN

SEPTEMBER 1987
GLENDALE, CALIFORNIA

OCTOBER 1987
WORCESTER, MA

NOVEMBER 1987
PALM BEACH, FLORIDA

NOT YET SCHEDULED:

HONOLULU, HAWAII
NEW YORK CITY, NEW YORK
ST. LOUIS, MISSOURI

Make the connection!

ANALOG Computing on Delphi puts you on-line with the world.

Delphi, an on-line, full-service information network, offers news and sports from the Associated Press, weather reports, movie reviews, shopping services, travel information, and more.

ANALOG Computing, the #1 magazine for Atari owners, brings you the Atari Users' Group on Delphi. We offer a message forum and an extensive database for up- or downloading—all from as little as 10 cents per minute from most U.S. cities, with no additional telephone charges and no extra charge for 1200 or 2400 bps. We'll use the group's conference feature for electronic meetings with well known Atarians and, of course, **ANALOG** staff. Bring on your toughest questions!

Special rates

Subscribers to **ANALOG Computing** or **ST-Log** may join without charge, and will receive a free lifetime Delphi membership, a Delphi Command Card and $5.00 of line-time credit applicable to their account. If you purchase the *Delphi Handbook*—the highly detailed manual on using the whole Delphi system—for $29.95, you will get an additional $20.00 of line-time credit. And you can subscribe to either **ANALOG Computing** or **ST-Log** directly, while on-line, to be eligible for these bonuses.

How to connect

First, select a data communications network: Telenet or Tymnet (in the U.S.), or DataPac (in Canada). In the Boston area, dial Delphi direct (617-576-0862). To determine your local Telenet number, dial 800-TELENET or 703-689-5700 (in Alaska, 907-264-7391). To obtain a Tymnet number, call 800-336-0149. If you have difficulty, call Delphi at 800-544-4005 (in Massachusetts, 617-491-3393). Current subscribers to **ANALOG Computing** or **ST-Log** should type *JOINATARI* when asked for user name. When asked for a password, type *ANALOG*. Those who wish to subscribe to either magazine on-line should, instead, type *SUBSCRIBE* at the password prompt. Once on Delphi, you'll find our group on the "Groups & Clubs" menu. To get there, just type *GR ATARI* from the main menu prompt.

THE #1 MAGAZINE FOR ATARI® COMPUTER OWNERS

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

P.O. BOX 23, WORCESTER, MA 01603

Dragonlord Dungeon

And sundry enhancements.

by Clayton Walnum

Over the years, I've written many games for Atari computers, but, if someone were to break into my house late one night, hold a gun to my head and tell me they were going to destroy all the programs I had written save one (my choice), I would beg for the salvation of **Dragonlord** (issue 29). I can't tell you why. All I can say is that, even though I spent weeks writing it and many more weeks testing it, I never became jaded with it. It always managed—for me, anyway—to retain its fun.

Just the other day, I got a letter from a couple of faithful **ANALOG Computing** readers, Fern and Michael Funk, and they seemed to agree with what I thought was an extremely prejudiced viewpoint. The letter read:

> Ever since my wife and I got our Atari 800XL, we've typed in over 300 games, programs and utilities, primarily from **ANALOG Computing**. Most games lose their enjoyability after a few months, but we feel **Dragonlord** is the exception. We find it intriguing and exciting each time we play.

Now let me tell you a little secret. Move in a little closer 'cause I've got to whisper. The **Dragonlord** in issue 29 is very different from the **Dragonlord** *I* now play.

Why? Well, it's like this. Some of us just don't know when to quit. Unless we have someone who's standing over us screaming, "All right, already! That's enough! Wrap this durned project up!" we just keep on going and going and . . .

One would think that the day I received the contract for **Dragonlord**, I would have considered the job complete. Nope. You could at least expect that, when **ANALOG** 29 hit the stands, I would have finally filed away my work disk. Nope. How about two years after its publication?

Nope.

The fact is: I kept getting more ideas. And, because I liked the game so much, I just *had* to make it better. Why waste a good idea?

When I received the Funks' letter and gave the matter a little thought, I suddenly realized that **Dragonlord** had changed *a lot*, and I just as suddenly had an irresistible urge to share those changes. Of course, I had to be prodded, had to *believe* that people would be interested.

The second paragraph of the letter not only got this article started, but added a new idea to the game, an idea I'm surprised didn't occur to me in the first place:

> After two years of enjoyment from **Dragonlord**, we have only one request. After fighting orcs, winning gold, shopping at the store and almost clearing the entire board—in the instances when we're not able to capture the dragon and the game ends—it would be great if a routine could be added to show where the dragon was hiding. It would cap off the perfect ending to a perfect game.

To Fern and Michael: your wish is my command. And, while I'm showing you where that dragon was holed up, I might as well pass along all the other changes, as well. Wow! What a bargain.

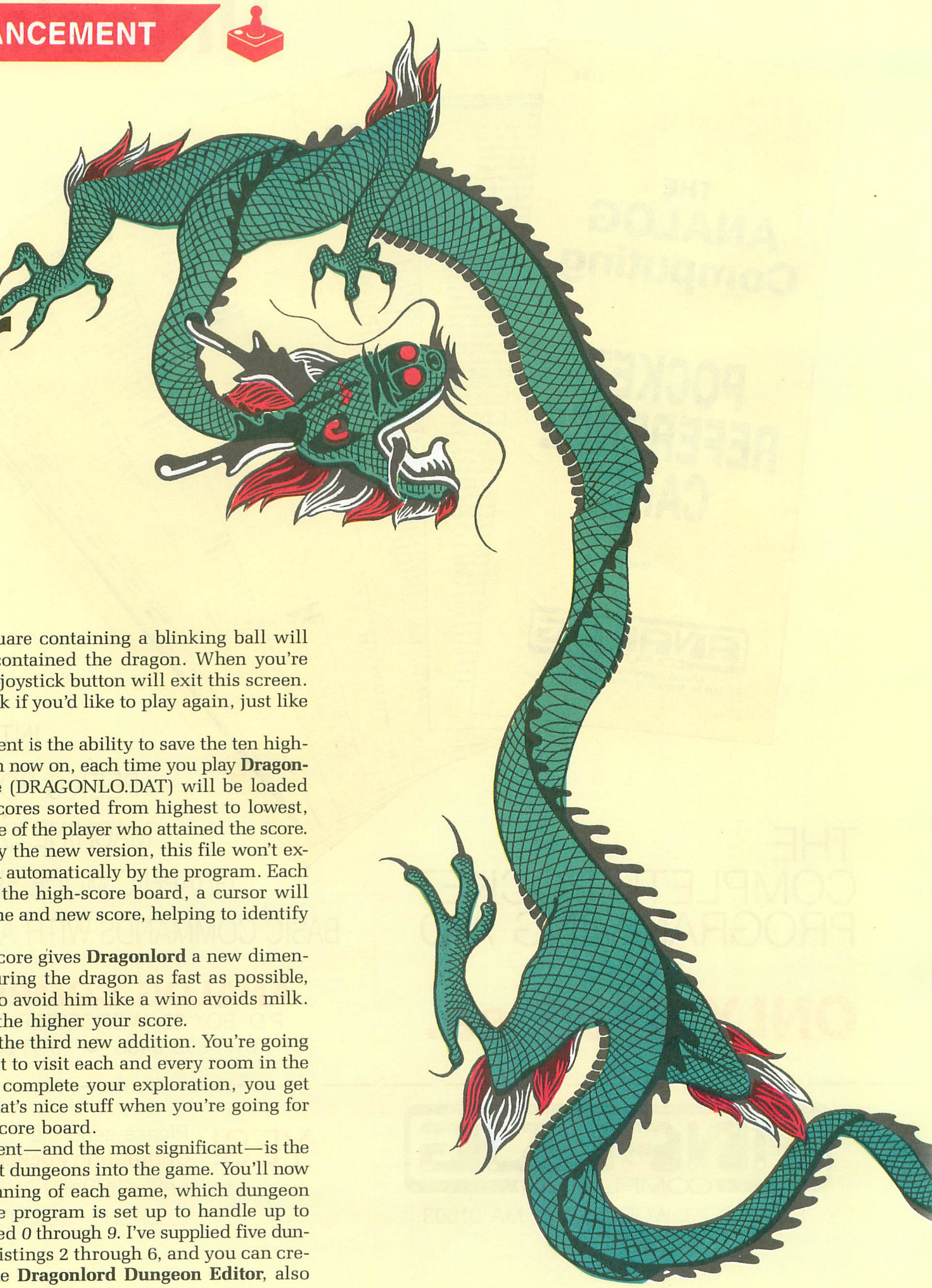
The enhancements.

The enhancements to **Dragonlord** can be found in Listing 1. Type this listing using the **BASIC Editor II** (issue 47) to check your work, then save the code to disk with the command *LIST "D:NEWDRAGN.LST"*. Load the original version of **Dragonlord** and merge the new lines with the command *ENTER "D:NEWDRAGN.LST"*. Save the new version of **Dragonlord**, making sure you retain a copy of the original—just in case something went wrong.

Just what *are* these enhancements?

The first on the list is the Funks' suggestion. Now, whenever you "die" before finding the dragon, the map screen

Editor



will appear, and a square containing a blinking ball will mark the room that contained the dragon. When you're through, pressing the joystick button will exit this screen. The game will then ask if you'd like to play again, just like the original version.

The next enhancement is the ability to save the ten highest scores to disk. From now on, each time you play **Dragonlord**, a high-score file (DRAGONLO.DAT) will be loaded from disk, with the scores sorted from highest to lowest, and including the name of the player who attained the score. The first time you play the new version, this file won't exist, but will be created automatically by the program. Each time a player gets on the high-score board, a cursor will appear next to his name and new score, helping to identify the addition.

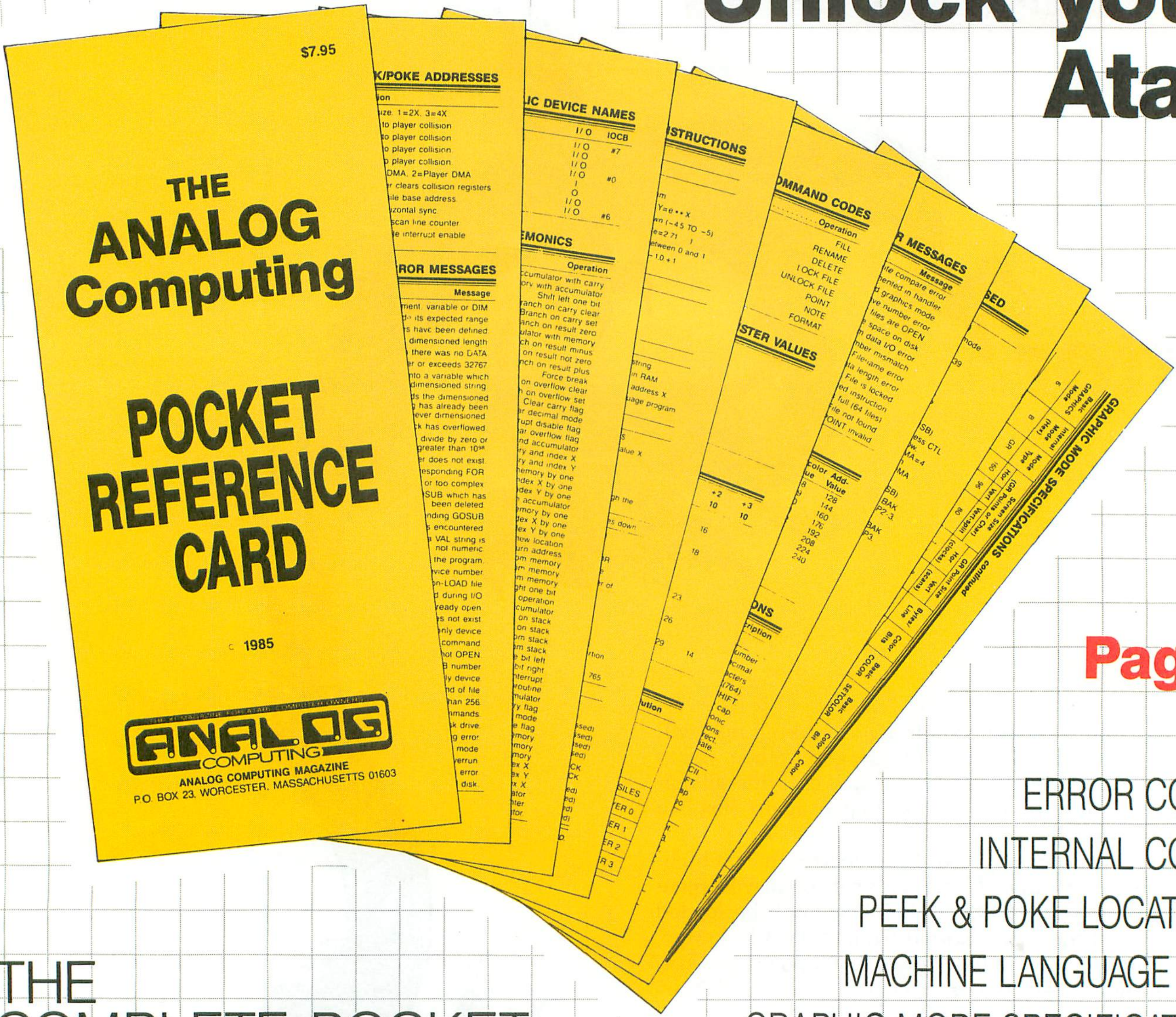
Playing for a high score gives **Dragonlord** a new dimension: instead of capturing the dragon as fast as possible, you're going to want to avoid him like a wino avoids milk. The longer you last, the higher your score.

Which brings us to the third new addition. You're going to want to do your best to visit each and every room in the maze, because if you complete your exploration, you get a 150-point bonus. That's nice stuff when you're going for the top of that high-score board.

The final enhancement—and the most significant—is the ability to load different dungeons into the game. You'll now be asked, at the beginning of each game, which dungeon you'd like to play. The program is set up to handle up to ten dungeons, numbered 0 through 9. I've supplied five dungeons in the form of Listings 2 through 6, and you can create your own with the **Dragonlord Dungeon Editor**, also included here.

To use the dungeons shown in Listings 2 through 6, type each one in separately, using the **BASIC Editor II** to check your work, then LIST them to your **Dragonlord** game disk disk, under the names DUNGEON1.DAT, DUNGEON2.DAT,

Unlock your Atari



16 Pages

ERROR CODES
INTERNAL CODES
PEEK & POKE LOCATIONS
MACHINE LANGUAGE AIDS
GRAPHIC MODE SPECIFICATIONS
BASIC COMMANDS WITH ABBREVIATIONS

THE COMPLETE POCKET PROGRAMMING AID

ONLY $7.95 ea.

ANALOG COMPUTING
P.O. BOX 23, WORCESTER, MA 01603
(617) 892-3488 • (617) 892-9230

THE #1 MAGAZINE FOR ATARI COMPUTER OWNERS™
ANALOG COMPUTING
P.O. BOX 23, WORCESTER, MA 01603

YES! Please send me ________ **ANALOG Computing Pocket Reference Cards.**
I am enclosing $7.95 per copy.
□ CASH □ CHECK □ CHARGE

Name ______________________________

Address ______________________________

City ______________ State ____ Zip ________

Card # ______________________________

Exp. date ______________________________

Signature ______________________________

DUNGEON3.DAT, DUNGEON4.DAT and DUNGEON5.DAT, respectively. DUNGEON1.DAT is the dungeon that was included in the original game. The others present completely new mazes.

When playing the game, if you try to load a dungeon that's not available on the disk, you'll be prompted for another dungeon number. Any dungeon you want to load must be on the same disk as the game program.

The *Dungeon Editor*.

In order to help you create your own dungeons, I spent months (okay, okay: four hours) laboring over the program shown in Listing 7. Type the listing, checking your work with **BASIC Editor II**, then save it to disk.

When you run the program, a familiar dungeon map will be drawn on the screen. You'll notice, though, that the rooms comprising this dungeon have no exits. I'll give you a guess as to who's going to add those exits. Go ahead, take a shot. (Hint: look in a mirror.)

Each of the seventy-seven rooms in the dungeon must have at least one exit. The round cursor shows which room you're currently working on. To add an exit to a room, press the *N, S, E* or *W* keys (a shiny, new nickel to the first person who can tell me what those letters stand for). If you make a mistake, a quick tap on the *C* key will clear the exits from the room, giving you a second chance.

To move between the rooms, use the arrow keys (without pressing CONTROL). If you should try to move the cursor beyond the boundaries of the dungeon, it'll wrap around in a logical manner, determined by the direction of your movement. For instance, trying to move the cursor up from the top row will place you on the same column, but in the bottom row. Moving right from the last room (77), will put you in the first room.

When you've finished your dungeon, press *Q* (another of those tough-to-figure-out initials, eh?), and you'll be asked what number you'd like the dungeon saved under. Enter a number from *0* to *9*, and press RETURN. If a dungeon of that number already exists on the disk, you'll be asked if you want to overwrite it. Remember, you can have only ten dungeons per game disk. Once you get a filename entered, the dungeon will be saved to disk in the proper format for the new version of **Dragonlord**.

The editor is a stripped-down model and doesn't include such options as saving (except for use with the game) or loading dungeons. So before you press Q, make sure your dungeon is just the way you want it. Otherwise, you'll have to start over.

In the course of creating new dungeons, I stumbled upon a truly sadistic design idea: it's possible to have rooms with one-way doors. For example, you can make things so that going north into a room doesn't necessarily mean you can go south to get out. Using this technique, you can design dungeons with areas that, once stumbled upon, cannot be exited without finding a teleport room or using a spell (chuckle, chuckle).

Back issues for sale.

Those of you who are new to **ANALOG Computing** may have missed the original version of **Dragonlord**. If all this talk of dungeons and dragons makes you want to know what the fuss is about, I urge to to order a copy of issue 29. Once you've typed **Dragonlord** (or downloaded a copy from the **ANALOG Computing** Atari Users' Group on Delphi), you can add the enhancements presented here and start your career as dragon hunter *extraordinaire*.

All you experienced Dragonlords out there will find that the enhancements lend the game a new feel—and maybe even a complete, 180-degree turn in strategy. When you're trying for the high score and approaching the end of the game, each turn will be "living on the edge." A single gold piece can make the difference between an entry on the high-score board and an anonymous death somewhere within the dragon's labyrinth.

Is that a challenge? You bet.

The two-letter checksum code preceding the line numbers here is *not* a part of the BASIC program. For further information, see the *BASIC Editor II*, in issue 47.

Listing 1.
BASIC listing.

```
TY 0 REM -----------------
KE 1 REM    DRAGONLORD     
RV 3 REM NEW VERSION 5/87
UC 4 REM -----------------
TH 80 FOR X=N1 TO N77:I(X)=N0:ER(X)=N0:NE
   XT X:I(39)=N7:ER(39)=1:RESTORE 3010
NO 955 IF  NOT ER(ROOM) THEN ER(ROOM)=1:R
   C=RC+1
HP 2162 IF ROOM=DR THEN 2175
GF 2163 GRAPHICS N0:POKE 756,CHSET:POKE 7
   52,N1:POKE 710,N0:POKE 711,54:POKE 712
   ,112
FO 2164 DL=PEEK(560)+256*PEEK(561)+N4:POK
   E DL+22,N6:POKE DL+23,N6:POKE DL+24,N6
   :POKE DL,N0:POKE DL+N1,MAP
QG 2165 POKE 88,N0:POKE 89,MAP:POKE 559,3
   4:POSITION 23,N21:? "     den of      ":P
   OSITION N4,22:? " the dragon "
KT 2166 R=DR:GOSUB ROOMPOS:POSITION COL,R
   OW:? "┌─┐↓←←←|  |↓←←←└─┘"
KP 2167 POSITION COL+N1,ROW+N1:? "●":FOR
   X=N1 TO N10:NEXT X
ZN 2168 POSITION COL+N1,ROW+N1:? " ":FOR
   X=N1 TO N10:NEXT X
AV 2169 IF STRIG(N0)=N0 THEN GOSUB SND1:G
   OTO 2175
VD 2170 GOTO 2167
RM 2175 GRAPHICS N18:POKE 756,CHSET:POSIT
   ION N2,N1:? #N6;"YOU ARE DEAD!"
DK 2195 IF RC=77 THEN SC=SC+150
QC 2210 OPEN #N1,N4,N0,"K:":GET #N1,A:CLO
   SE #N1:IF A<>ASC("N") AND A<>ASC("Y")
   THEN 2210
QU 2211 CN=N0:FOR X=N1 TO N10:PR(X)=N0:NE
   XT X
VJ 2212 IF SC<=RS(N10) THEN 2220
DX 2213 CN=CN+N1:IF SC>RS(CN) THEN 2215
QY 2214 GOTO 2213
UX 2215 FOR X=N9 TO CN STEP -N1:RS(X+N1)=
   RS(X):PR(X+N1)=PR(X):RS$((X+N1)*N9-N8,
   (X+N1)*N9)=RS$(X*N9-N8,X*N9):NEXT X
WP 2216 B=N0:FOR X=N1 TO N9:IF N$(X,X)<>"
    " THEN NEXT X:GOTO 2219
LW 2217 B=B+N1:IF B=N2 THEN FOR Y=X-N1 TO
    N9:N$(Y,Y)="█":NEXT Y:GOTO 2219
MM 2218 NEXT X
LY 2219 RS(CN)=SC:PR(CN)=N1:RS$(CN*N9-N8,
   CN*N9)=N$:F=N1
```

Dragonlord Dungeon Editor *continued*

```
BL 2220 CN=N0:IF  NOT F THEN 2223
XU 2221 CLOSE #N1:OPEN #N1,N8,N0,"D:DRAGO
   NLO.DAT":FOR X=N1 TO N10:? #N1;RS(X):?
    #N1;RS$(X*N9-N8,X*N9):NEXT X
ZM 2222 CLOSE #N1:F=N0
SH 2223 GRAPHICS N17:POKE 756,CHSET:POSIT
   ION N3,N0:? #N6;"record scores":POSITI
   ON N3,N1:? #N6;"----------------"
DX 2224 FOR X=N1 TO N10:POSITION N3,X+N4:
   ? #N6;RS$(X*N9-N8,X*N9);"■";RS(X):NEXT
    X
QJ 2225 FOR Z=N1 TO N10:IF PR(Z)=N0 THEN
   NEXT Z:GOTO 2229
QB 2226 FOR X=N1 TO N20:NEXT X:POSITION N
   2,Z+N4:? #N6;"▶":FOR X=N8 TO N0 STEP -
   0.4:SOUND N0,N10,N10,X
ZR 2227 SOUND N1,N15,N10,X:FOR Y=N1 TO N4
   :NEXT Y:SOUND N0,N0,N0,N0:SOUND N1,N0,
   N0,N0:FOR Y=N1 TO N3:NEXT Y:NEXT X
NL 2228 NEXT Z
HM 2229 POKE 711,54:POSITION N2,22:? #N6;
   "push fire button"
WK 2230 FOR X=N1 TO 30:NEXT X:POKE 711,N0
   :IF STRIG(N0)=N0 THEN 2232
IJ 2231 FOR X=N1 TO N10:NEXT X:POKE 711,5
   4:GOTO 2230
NT 2232 IF A=ASC("N") THEN POKE 82,N2:END
WO 2233 POSITION N2,22:? #N6;"   ONE MOME
   NT   "
PG 2240 RESTORE 2940:HP=50:STR=100:SPL=N0
   :PIE=N1:CNT=N0:B=N0:WN=N1:G=60:ROOM=39
   :SC=N0:SM=N0:TURN=N0:RC=1:GOTO 80
AU 2300 DIM ER(77),RM(N77),I(N77),RM$(255
   ),MV$(N20),N$(N9),W$(28),DIR$(N20),S$(
   120),B$(40),CL$(40)
IK 2310 DIM R$(N1),R(N5),C$(N1),E$(N4),D$
   (81),F$(N14),RS(N10),RS$(90),PR(N10)
ZU 2340 WN=N1:PIE=N1:CNT=N0:G=60:SPL=N0:H
   P=50:STR=100:B=N0:ROOM=39:SM=N0:SC=N0:
   TURN=N0:RC=1
QI 2350 ROOMPOS=180:CHOOSE=290:DIR=370:CL
   UE=210:DEAD=2162:DELAY2=360:SND1=160:S
   ND2=170:TEL=130
EV 3030 POKE 559,0:POKE 87,N0:POKE 88,N0:
   POKE 89,MAP:A=USR(ADR(CL$),N15)
VR 3040 POSITION N18,N9:? RM$(RM(ROOM)*N1
   7-N16,RM(ROOM)*N17)
SB 3301 TRAP 3351:OPEN #N1,N4,N0,"D:DRAGO
   NLO.DAT"
TC 3302 FOR X=N1 TO N10:INPUT #N1;A:RS(X)
   =A:INPUT #N1;N$:RS$(X*N9-N8,X*N9)=N$:N
   EXT X:TRAP 40000:CLOSE #N1
IY 3340 FOR X=N1 TO N9:IF N$(X,X)<>"-" TH
   EN NEXT X
KE 3342 F$="D:DUNGEON .DAT"
KG 3343 POSITION N2,N15:? "WHAT DUNGEON <
   0-9>   ←←←";:TRAP 3343:INPUT D:CLOSE #
   1:F$(N10,N10)=STR$(D)
LG 3344 OPEN #1,4,0,F$:CLOSE #1
CA 3345 GRAPHICS N0:POKE 559,N0:? :? :? "
   ENTER";CHR$(34);F$:? :? :? "CONT"
CQ 3346 POSITION N0,N0:POKE 842,N13:STOP
SW 3347 POKE 842,N12:RESTORE 2940:FOR Y=N
   1 TO 77:READ A:RM(Y)=A:NEXT Y
IR 3350 TRAP 3030:N$(X)="         ":POP :
   TRAP 40000:POKE 559,34:GOTO 550.
CF 3351 RS$(N1)=" ":RS$(90)=" ":RS$(N2)=R
   S$:CLOSE #N1:OPEN #N1,N8,N0,"D:DRAGONL
   O.DAT"
VM 3352 FOR X=N1 TO N10:RS(X)=N0:? #N1;RS
   (X):? #N1;RS$(X*N9-N8,X*N9):NEXT X:CLO
   SE #N1:GOTO 3310
```

●

Listing 2.
BASIC listing.

```
YC 2940 DATA 4,6,6,9,6,9,6,9,6,6,5
UM 2950 DATA 1,9,9,8,5,2,4,8,9,9,3
BU 2960 DATA 12,11,2,4,3,2,1,5,2,11,12
RU 2970 DATA 7,6,10,10,6,10,6,10,10,6,15
CG 2980 DATA 11,12,2,1,5,2,4,3,2,12,11
SW 2990 DATA 4,8,8,9,3,2,1,9,8,8,5
QX 3000 DATA 1,6,6,8,6,8,6,8,6,6,3
```

●

Listing 3.
BASIC listing.

```
MQ 2940 DATA 4,9,9,9,9,9,9,9,9,9,5
PF 2950 DATA 7,8,8,8,8,8,8,8,8,8,15
ZI 2960 DATA 4,5,4,9,9,10,9,9,5,4,5
SP 2970 DATA 2,7,15,2,7,10,15,2,7,15,2
KE 2980 DATA 1,3,1,8,8,10,8,8,3,1,3
WY 2990 DATA 7,9,9,9,9,9,9,9,9,9,15
AP 3000 DATA 1,8,8,8,8,8,8,8,8,8,3
```

●

Listing 4.
BASIC listing.

```
VR 2940 DATA 4,6,6,6,9,6,9,6,6,6,5
AJ 2950 DATA 1,6,5,9,10,9,10,9,4,6,3
IF 2960 DATA 4,6,3,7,3,2,1,15,1,6,5
XX 2970 DATA 1,6,5,7,6,10,6,15,4,6,3
MO 2980 DATA 4,6,3,7,5,2,4,15,1,6,5
PP 2990 DATA 7,6,5,8,10,8,10,8,4,6,15
SN 3000 DATA 1,6,8,6,8,6,8,6,8,6,3
```

●

Listing 5.
BASIC listing.

```
VP 2940 DATA 4,6,9,6,5,9,4,6,9,6,5
XJ 2950 DATA 7,6,10,6,15,2,7,6,10,6,15
GU 2960 DATA 1,6,8,6,3,2,1,6,8,6,3
GE 2970 DATA 7,6,6,6,6,10,6,6,6,6,15
QK 2980 DATA 4,6,9,6,5,2,4,6,9,6,5
HT 2990 DATA 2,13,10,14,2,2,2,13,10,14,2
KH 3000 DATA 1,6,8,6,3,8,1,6,8,6,3
```

●

Listing 6.
BASIC listing.

```
MQ 2940 DATA 4,9,9,9,9,9,9,9,9,9,5
DQ 2950 DATA 7,10,10,10,10,10,10,10,10,10
   ,15
DT 2960 DATA 7,10,10,10,10,10,10,10,10,10
   ,15
DW 2970 DATA 7,10,10,10,10,10,10,10,10,10
   ,15
DZ 2980 DATA 7,10,10,10,10,10,10,10,10,10
   ,15
EC 2990 DATA 7,10,10,10,10,10,10,10,10,10
   ,15
AP 3000 DATA 1,8,8,8,8,8,8,8,8,8,3
```

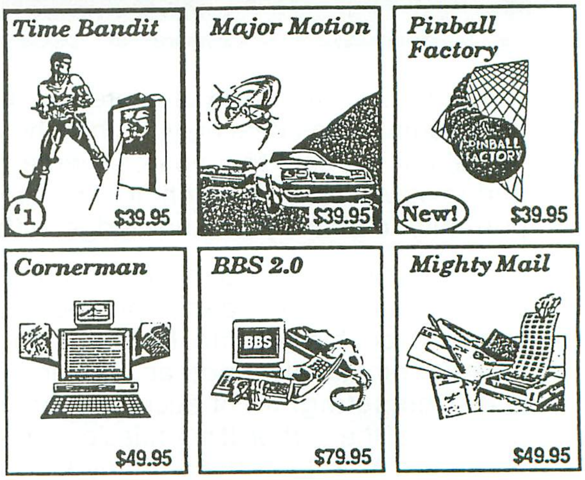
●

Listing 7.
BASIC listing.

```
YG 10 GRAPHICS 1:POKE 710,0:POKE 752,1
WF 20 POSITION 5,6:? #6;"dragonlord":POSI
   TION 3,8:? #6;"DUNGEON EDITOR":? "
```

```
          By Clayton Walnum"
RS 30 ? :? " Copyright 1987 by ANALOG Com
   puting"
KW 40 N=1:E=2:S=3:W=4:REPEAT=0
MP 50 DIM RM$(17),F$(15),A$(1),L$(75),B$(
   37)
YD 60 DIM RM(77,4),CMP(15,4)
VG 70 FOR X=1 TO 77:FOR Y=1 TO 4:RM(X,Y)=
   0:NEXT Y:NEXT X
XQ 80 FOR X=1 TO 15:FOR Y=1 TO 4:READ A:C
   MP(X,Y)=A:NEXT Y:NEXT X
MK 90 DATA 1,1,0,0,1,0,1,0,1,0,0,1,0,1,1,
   0,0,0,1,1,0,1,0,1,1,1,1,0,1,1,0,1,0,1,
   1,1,1,1,1,1,1,0,0,0,0,0,1,0
WT 100 DATA 0,1,0,0,0,0,0,1,1,0,1,1
AD 110 B$(1)=" ":B$(37)=" ":B$(2)=B$
YU 120 RM$="┌─┐↓←←←│ │↓←←←└─┘"
BX 130 GRAPHICS 0:POKE 710,0:POKE 752,1
ET 140 FOR Y=0 TO 18 STEP 3:FOR X=3 TO 33
    STEP 3
BQ 150 POSITION X,Y:? RM$
KM 160 NEXT X:NEXT Y:POKE 766,1
UK 170 ? "↑ = UP  ↓ = DOWN  → = RIGHT  ←
   = LEFT":POKE 766,0
QR 180 ? "N,S,E,W = EXITS  C = CLEAR  Q =
    QUIT"
MT 190 C=4:R=1:OC=C:OR=R:RM=1:POSITION C,
   R:? "●"
IW 200 OPEN #1,4,0,"K:":GET #1,A:CLOSE #1
MT 210 IF A=ASC("Q") OR A=ASC("q") THEN 3
   20
AZ 220 IF (A=ASC("N") OR A=ASC("n")) AND
   RM>11 THEN POSITION C,R-1:? "┴":RM(RM,
   N)=1
GE 230 IF (A=ASC("S") OR A=ASC("s")) AND
   RM<67 THEN POSITION C,R+1:? "┬":RM(RM,
   S)=1
HM 240 IF (A=ASC("E") OR A=ASC("e")) AND
   RM/11<>INT(RM/11) THEN POSITION C+1,R:
   ? "├":RM(RM,E)=1
KX 250 IF (A=ASC("W") OR A=ASC("w")) AND
   (RM-1)/11<>INT((RM-1)/11) THEN POSITIO
   N C-1,R:? "┤":RM(RM,W)=1
TZ 260 IF A=ASC("C") OR A=ASC("c") THEN P
   OSITION C-1,R-1:? RM$:FOR X=1 TO 4:RM(
   RM,X)=0:NEXT X
ER 270 IF A=ASC("*") THEN RM=RM+1:IF RM>7
   7 THEN RM=1
DS 280 IF A=ASC("+") THEN RM=RM-1:IF RM<1
    THEN RM=77
PC 290 IF A=ASC("-") THEN RM=RM-11:IF RM<
   1 THEN RM=RM+77
AO 300 IF A=ASC("=") THEN RM=RM+11:IF RM>
   77 THEN RM=RM-77
AX 310 GOSUB 520:GOTO 200
CI 320 GOSUB 510:IF REPEAT THEN 380
SI 330 F$="D:DUNGEON0.DAT"
IJ 340 TRAP 340:POSITION 2,21:? "DUNGEON
   NUMBER";:INPUT DN
FG 350 IF DN<1 OR DN>9 THEN 340
RK 360 TRAP 380:F$(10,10)=STR$(DN):CLOSE
   #1:OPEN #1,4,0,F$
DO 370 POSITION 2,22:? "FILE ALREADY EXIS
   TS! OVERWRITE IT";:INPUT A$:IF A$<>"Y"
    THEN 320
NZ 380 CLOSE #1:OPEN #1,8,0,F$:GOSUB 510:
   POSITION 5,21:? "WORKING..."
UZ 390 LINE=2940
MD 400 FOR RC=1 TO 67 STEP 11
PB 410 L$=STR$(LINE):L$(5)=" DATA ":TRAP
   40000
MR 420 FOR RM=RC TO RC+10:T=1:GOSUB 520
LN 430 FOR D=1 TO 4
CY 440 IF RM(RM,D)=CMP(T,D) THEN NEXT D:G
   OTO 470
ZM 450 POP :T=T+1:IF T>15 THEN 500
OG 460 GOTO 430
DF 470 L$(LEN(L$)+1)=STR$(T):L$(LEN(L$)+1
   )=",":NEXT RM
PQ 480 ? #1;L$(1,LEN(L$)-1):LINE=LINE+10:
   NEXT RC
LD 490 GRAPHICS 0:? "ALL DONE!":END
SQ 500 POSITION 2,22:? "ROOM #";RM;" DOES
   N'T HAVE AN EXIT!":CLOSE #1:REPEAT=1:G
   OTO 310
QU 510 POSITION 2,21:? B$:POSITION 2,22:?
    B$
EM 520 R=INT((RM-1)/11)*3+1:C=(RM-INT((RM
   -1)/11)*11+1)*3-2:POSITION OC,OR:? " "
   :POSITION C,R:? "●":OC=C:OR=R:RETURN
```

●



// Review

continued from page 39

Once the components and connections are defined and associated with a particular library, you can begin positioning the components. You continue by test routing, to minimize incompleted connections during autoroutes. The PC board can be resized, to minimize wasted board space. Blockades can be set aside, for drilling mounting holes on the final product, for example. You will generally complete the routing in the manual mode, then save and print your final work.

The manual is quite well done (after a major revision). It begins with several sample designs to get you accustomed to the process of going from schematic to final print. From there, the text coaches you through the more advanced features of **PC Board Designer**, followed by a section on practical design techniques. A detailed explanation of all the program's menus is presented, once you've gotten your feet wet. Another chapter is dedicated to the component definition program, which can be used to create your own libraries, up to sixty per file. Among the appendices is a complete list of all the component definitions provided with the **PC Board Designer** package.

There's room for a lot of improvement in **PC Board Designer**. The editor for component and connection lists is awkward, and should be a separate GEM window. This would allow a slightly larger PC board, and make the generation of a complete net list much less tedious. It would be nice if pins on circuits could be defined as input or output, analog or digital. Then, during routing, the program could warn you if you're making an improper connection. Since the price has been lowered drastically and the manual improved significantly, I feel that **PC Board Designer** provides a good value. It isn't nearly as powerful as some of the design software out for the IBM machines, but it is much less expensive. The program's biggest flaw is the size limitation of the PC boards. A

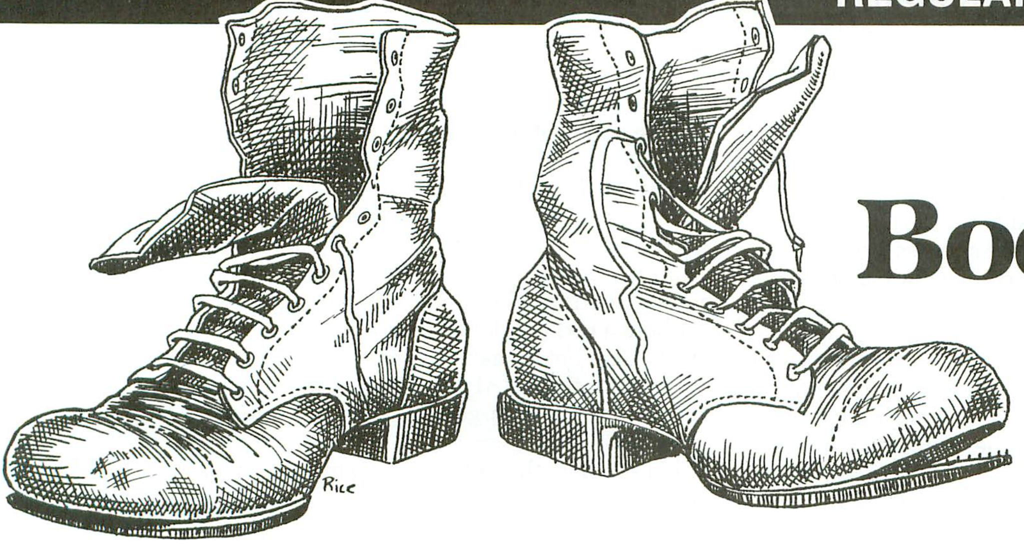
Attention Programmers!

ANALOG Computing is interested in programs, articles, tutorials and hardware/software review submissions dealing with the whole line of Atari personal computers, including the new ST models. If you feel that you can write as well as you can program, then submit those articles and reviews that have been floating around in your head, awaiting publication. This is your opportunity to share your knowledge with the growing family of Atari computer owners.

All submissions for publication, both program listings and text, should be provided in printed and magnetic form. Typed or printed copy of text is mandatory and should be in upper and lower case with double spacing. By submitting articles to **ANALOG Computing**, authors acknowledge that such materials, upon acceptance for publication, become the exclusive property of **ANALOG Computing**. If not accepted for publication, the articles and/or programs will remain the property of the author. If submissions are to be returned, please supply a self-addressed, stamped envelope. All submissions of any kind must be accompanied by the author's full address and telephone number.

Send programs to:
Editor, **ANALOG Computing**
P.O. Box 23, Worcester, MA 01603.

REGULAR FEATURE


Rice

Boot Camp

Son of Macromania.

by Karl E. Wiegers

Let's begin today's discussion with a bit of time travel. Way back in issues 41 and 42 of **ANALOG Computing**, I showed you how to simulate a number of Atari BASIC graphics commands in assembly language. The culmination of that discussion was a program that drew a yellow five-pointed star on a purple background in graphics 5. Please go dig up issue 42 if you have it; I'll wait here until you get back. (Don't panic if you don't have issue 42 handy—you'll live.)

Got it? Now, please turn to page 101 of that issue. There you see a BASIC program that did exactly what our final assembly program did. I'll reproduce it here, in case your back issues are lost somewhere:

```
10 GRAPHICS 5+16
20 SETCOLOR 0,1,12
30 SETCOLOR 4,6,8
40 COLOR 1
50 PLOT 20,15
60 DRAWTO 60,15
70 DRAWTO 28,35
80 DRAWTO 40,5
90 DRAWTO 52,35
100 DRAWTO 20,15
110 GOTO 110
```

What would you say if I told you that, after studying today's **Boot Camp**, you'll be able to write a program in assembly language that looks almost identical to the above BASIC program—and does exactly the same thing? I detect some murmurings of skepticism in the crowd. But it's all true, through the Miracle of Macros!

Today, I present macros to simulate ten common BASIC statements: SETCOLOR, POKE, POSITION, OPEN, CLOSE, GRAPHICS, PRINT, COLOR, PLOT and DRAWTO. Once you have this set in hand, writing assembly programs to perform any of these functions is scarcely more difficult than in BASIC. We'll begin to build a macro library that covers most of the common BASIC commands, as well as some commands BASIC doesn't have but should. Future issues will add to this library from time to time.

The (possible) bad news is that you must have a macro assembler to use these programs. My examples are specifically written for MAC/65. With a little effort, you can modify most of them to work with the Atari Macro Assembler (AMAC), although I'll point out some important differences between the assemblers. If you have a different macro assembler, try to adapt the ones I have here to your own environment. The manual for your assembler should help with specific syntax questions.

Even if you don't have a macro assembler, I think you'll get some useful ideas about assembly programming in the words that follow, so plunge onward.

Macro refresher.

Last month, I introduced the idea of macros and looked at two examples, POKE and SETCOLOR, in detail. A macro is just a named block of assembly language statements. When we invoke (call, refer to, use—all are synonyms) a macro, the instructions in the macro are placed right into the source (and hence object) code upon assembly. This is called "expansion" of the macro. Two very powerful features of macros are: the ability to pass variable parameters each time the macro is invoked, and the use of conditional (.IF, .ELSE, .ENDIF) assembly statements within the macro. These methods let the macro generate different object code each time it is expanded.

Today, we'll look at some other ways to use macros. We'll see macros that invoke other macros, macros that call subroutines, macros that use string (as opposed to numeric) parameters, macros containing labels, and more. I'll also demonstrate a convenient way to keep track of your macro (and subroutine) libraries as you build them. All these methods will help you write assembly language programs much faster and cleaner than you may be doing now. So let's get started!

A macro library.

Listing 1 contains the assembly code definitions for the ten macros I mentioned earlier, plus the equates needed by the macros. You could use these macros in a couple of ways as you write programs. In the simplest case, you could just type the code for the macros you need into each program you write. A less time-consuming method is to put each macro into a separate file and use the editor portion of your assembler to pull in the ones you need and piece together a program. But the best way is to use the .INCLUDE directive of your assembler.

The .INCLUDE directive has the form: *.INCLUDE #D: MACRO.LIB*. Whenever this directive is encountered during assembly of a program, the entire contents of the file named are inserted into the source code being assembled, at the location of the .INCLUDE. This way, you can manage a large program—or programs containing common segments of code—by linking together several smaller files.

I'd like you to create a file called MACRO.LIB that contains everything in Listing 1. Now, anytime you want to use a macro from this library, just put the .INCLUDE #D:MACRO.LIB statement at the beginning of your program. Please note that, with MAC/65, .INCLUDEd files must be saved in the standard MAC/65 tokenized form. LISTed files cannot be .INCLUDEd this way.

There *is* a bad side to the use of .INCLUDE files. One of MAC/65's strong points is that it assembles very quickly from the source code in RAM. If we .INCLUDE some files, the assembly process will slow down, as those files must be read from disk. This isn't such a big deal with AMAC, since AMAC must read source code files from disk to assemble them, anyway. If you have a 130XE with a RAMdisk, you're in luck. I always begin my MAC/65 sessions by booting DOS, setting up the RAMdisk, then copying all my .INCLUDE files onto the RAMdisk. (If the DOS you use permits batch files, you can copy these files automatically on booting.) All my .INCLUDE statements then refer to files on drive D8:, and assembly is still very fast.

Listing 2 is another file that will be .INCLUDEd in our programs. Please call this one *SUBS.LIB*. It contains a couple of subroutines that are called by some of the macros. You might consider adding to this file any other subroutines you use often in your own programs. We have encountered a few in previous columns, such as timing delay routines. Now, let's look at a couple of sample programs and study the macros in more detail.

Example 1. — Text.

Listing 3 is a short assembly program that looks a lot like BASIC. It sets up a standard graphics 2 screen, changes a couple of color registers, and prints one line of text on the screen and one in the text window. Sounds simple enough.

Note that the MACRO.LIB file is .INCLUDEd at the beginning of the source code, and the subroutines are at the end. Macro definitions can go just about anywhere except in the middle of the program. I like to group them at the top. Conceptually, I like the subroutines at the end of the program, so the assembler doesn't get confused by hitting source code before it gets to the program origin address in Line 150.

I won't dissect the SETCOLOR and POKE macros, because we covered them in gory detail last month. The GRAPHICS macro does essentially the same thing that the Atari BASIC GRAPHICS statement does. It opens the graphics screen device S: in a particular graphics mode, with or without a text window. By now, you're all old hands at opening IOCBs, but to make it even easier, I wrote an OPEN macro. In fact, the GRAPHICS macro actually calls the OPEN macro. . . but we're getting ahead of ourselves.

OPEN.

Let's look at the OPEN macro in Lines 1290-1550 of Listing 1. For each macro, I've shown a usage example and described the parameters it uses. Notice that no comments are included in the body of the macro (that is, between the .MACRO and .ENDM directives). Since the whole macro is plugged right into the source code during the assembly process, you waste RAM if you have comments within the macro definition.

The OPEN macro, just like the OPEN command in BASIC, requires four parameters: the IOCB number to open; two auxiliary values; and the name of the device to use. As with all these macros, the first function is to test and make sure the correct number of parameters is supplied (Lines 1300-1310). If not, an error message is printed on the assembly listing.

Incidentally, you may have seen macros similar to mine in other publications. There are different ways to write these macros, although they all do essentially the same thing if they're trying to simulate their BASIC counterparts. You can make the macros quite a bit more elaborate than I have in many cases, but I'll let you do some of that on your own.

The first strange thing about the OPEN macro is that device name jazz. Aren't those usually some kind of character string, like S:, or D1:DATA.FIL? They sure are. Fortunately, MAC/65 permits string parameters to be used in macros, something AMAC can't do. String parameters are numbered as usual, but a dollar sign ($) after the percent sign (%) identifies the parameter as a string: %$4, for example. Line 1340 reserves a block of data bytes the length of whatever is in %$4 as the device name, followed by an end-of-line (EOL) character. We don't want to try to "execute" those data bytes as instructions, so Line 1330 jumps over the line labeled *@DEVICE*.

Since AMAC can't handle string variables, you have to get more inventive. You could write a version of the OPEN macro that uses an address for the fourth (device) parameter. This address could be a byte string set to something like S: or D1:DATA.FIL. Lines 580-630 of Listing 2 show some common device names you might want to use. If you take this approach, an AMAC OPEN macro call might look like: *OPEN 6,12,2,S*.

And now a word about labels. You can use labels in macros just like you do elsewhere in your program. However, it's actually possible to refer to those intra-macro labels from outside the macro, which can cause some strange things to happen. It's a good practice to use a different convention for naming labels inside and outside macros. I use label names beginning with an at sign (@) within a macro. Fortunately, the label definitions are changed automatically each time the macro is invoked, so @DEVICE will have a unique address each time you call OPEN. You don't have to worry about this, since the assembler will.

Line 1360 uses the first parameter to determine which IOCB we want to use. This parameter must be an actual IOCB number, not an address containing the desired IOCB number. Lines 1370-1400 set up the IOCB with the pointers to the device name passed in parameter %$4.

I elected to provide the flexibility to handle the auxiliary values as being either actual numbers or addresses containing the numbers to be used. Lines 1410-1460 illustrate the logic used for auxiliary value *1*. If parameter %2 is greater than *255*, I assume it's referring to an address, so I snatch the contents of that address in Line 1420. Otherwise, I assume the actual auxiliary number has been supplied, so I use it directly in Line 1440. Whatever value I end up with is stored in ICAX1 in Line 1460. The same method is used for the second auxiliary byte in Lines 1470-1520. Of course, this method will fail if you want to point to an address in page 0, in which case %1 will be *255* or lower (think about it!) You could use this same address/value approach to make the IOCB selection from parameter 1 more intelligent, if you like.

At this point, all necessary registers are set to perform the OPEN, except the command value and the jump to CIO. Since these steps are the same for all OPEN operations, I decided to put them into a subroutine called *OPENIOCB* (Lines 390-430 in Listing 2). By doing this, we save a few bytes of object code each time the OPEN macro is expanded. Including a subroutine call within a macro definition makes your object code more efficient. In this case, we're only talking about a few bytes, so if you feel more comfortable including the entire OPEN procedure within the macro, go right ahead. I often prefer that sort of comprehensibility to a small efficiency improvement.

This is all fine, but what does it have to do with the GRAPHICS command? Well, since the heart of the BASIC GRAPHICS command is opening IOCB 6 for the S: device, I thought we'd better cover the OPEN macro first. Now, how do we take the single parameter from the GRAPHICS macro and turn it into the four parameters needed by OPEN?

GRAPHICS.

The GRAPHICS macro lives in Lines 1870-2070. It requires one parameter, a value (not address) that is any valid BASIC graphics mode for your computer. (Remember that the XL/XE machines have access to some ANTIC modes from BASIC that the 400/800 do not.) As with BASIC, you can add *16* to this mode number to suppress the text window.

Line 1910 jumps beyond a couple of bytes reserved for the two auxiliary parameters needed by OPEN. The BASIC graphics mode number must be placed into @AUX2. Line 1980 keeps only the 4 least significant bits of parameter %1, thereby making sure that @AUX2 has a valid value from *0* through *15*.

Some gyrations are required to set the right value in @AUX1 to control the text window. Back in issue 41, I told you that @AUX1 (ICAX1 in IOCB terminology) should be set to *28* to have a text window present, and to *12* to get rid of the text window. However, since we told the GRAPHICS macro to get rid of the text window by adding *16* to the graphics number (equivalent to setting bit 5), some bit manipulations are needed. Lines 2000-2040 load @AUX2 with a decimal *28* if the graphics mode is *less than 16* (window) and a decimal *12* if the mode is *greater than 16* (no window). You might find it illuminating to work through the AND, EOR, and ORA for the two cases.

Line 2050 finally does the dirty work. Notice that we are invoking the OPEN macro from within the GRAPHICS macro. This nested macro situation is perfectly legal. (Oh right, when I talked about macro ecology last issue, I forgot to tell you where they nest!) The four parameters passed to OPEN are a *6* (IOCB 6 for the screen, always), the values we synthesized for @AUX1 and @AUX2, and the screen device S:. Note the use of the literal string in parameter 4.

This isn't so bad, is it? The nicest part is: now that you have these macros, you don't have to worry about such details anymore.

Macros calling macros.

What do you suppose would happen if you had a macro that called *itself*? This is called "recursion." You might be able to get away with this, provided you have conditional code to permit an exit under specific circumstances. If a recursive macro with no way out tries to call itself indefinitely, your computer will explode after 5 seconds (just kidding).

Getting back to Listing 3, once the GRAPHICS macro has been called, you can set the color registers to whatever values you like. Line 170 shows how to use the SETCOLOR macro for this, and Line 180 illustrates the use of the POKE macro to set a color register. Now let's print something on the screen.

POSITION.

POSITION is one of the obvious candidates for a simple macro. Lines 1040-1150 of Listing 1 show that POSITION requires two parameters, the X- and Y-coordinates where the cursor should go. Both parameters are assumed to be values, not addresses. I think you can follow the code in this macro without too much difficulty. The first parameter (X-coordinate) can only be greater than *255* in graphics mode 8, where it can range from *0* to *319*.

PRINT.

Here's another one where you can imagine a pretty sophisticated macro to fully emulate its BASIC counterpart. In BASIC, a PRINT command can be followed by nothing (print a blank line), a literal (print what's between the quotes), or a variable name (print the current value of the variable). Also, the IOCB number to use may or may not be specified; the absence of an IOCB number means use IOCB 0 for the screen editor, device E:.

I haven't written a full implementation of the PRINT macro; the MAC/65 manual contains a very detailed version you might try to comprehend. The one I give here doesn't handle the printing of literals, just strings at addresses. Each such string must terminate with an EOL character. Lines 250-260 of Listing 3 are some text strings to print.

The PRINT macro is in Lines 2260-2450 of Listing 1. The IOCB number is optional; IOCB 0 is assumed if you supply only one parameter (the string address). This macro calls subroutine PRINTLINE (Lines 230-310 of Listing 2), which issues the PUTREC command to CIO. In Listing 3, text is printed on both the graphics 2 screen and in the graphics 0 text window. You shouldn't have much trouble adapting this macro to AMAC. How's about whipping up a method to handle string literals, thereby completing the emulation of the BASIC PRINT command (except for trailing commas and semicolons)?

CLOSE.

It's a good practice to close an IOCB when you're all done with it. Hence, the CLOSE macro (Lines 1650-1740 of Listing 1). Very simply, it takes the parameter you pass as the IOCB number to be closed, and does the dirty deed forthwith. What could be simpler?

This wraps up our discussion of macros in the first sample program. Play with the graphics mode, colors, position and text strings until you're convinced this really *is* almost as easy as BASIC. The big difference becomes apparent when you have to go through the assembly process each time you make a change in Listing 3. The value of an interpreted language like BASIC becomes clearer. However, things execute an awful lot faster in machine language, so you just have to grit your teeth through the assembly step.

Example 2. — Graphics.

Now turn to Listing 4. Do you notice a strong similarity to the short BASIC program you encountered at the beginning of this article? Essentially, we've just added some of the assembly directives, but the resemblance is uncanny. This program uses just three new macros, COLOR, PLOT and DRAWTO.

COLOR.

Lines 2550-2620 of Listing 1 are devoted to the COLOR macro. This works exactly like the BASIC COLOR statement. However, we do need to stash the color value you pass until we need it in the PLOT and DRAWTO macros. Address $C8, labeled *COLOR*, is the temporary holding tank for the color value. Note that there's no conflict between this label and the identical macro name.

PLOT.

Lines 2730-2870 of Listing 1 define the PLOT macro. As you expect, it takes two parameters, X- and Y-coordinates, to plot a point. It simply passes these along to the POSITION macro in Line 2770, which actually positions the cursor in the right place. Again, we have a case of a macro calling a macro. We've now seen examples in which a macro passes parameters to a second macro in the form of numeric values, labels and string values (all from GRAPHICS to OPEN), and as raw parameters (%1 and %2 from PLOT to POSITION).

PLOT assumes you are using IOCB 6 (Line 2780). It uses an unusual form of the CIO PUTREC operation. Normally, PUTREC requires that you point to a buffer address and set a buffer length for the record to be output. An exception to this allows you to output a 1-byte "record." First, set the buffer length to *0* (Lines 2810-2830). Then load the accumulator with the character to be output. Line 2840 takes the number stored in address COLOR from the COLOR statement. It turns out that "printing" a *1* in this way (COLOR 1) selects color register *0*, and so on for the other COLOR values. A simple JSR to CIOV in Line 2850 plots a single point in our graphics 5+16 screen.

DRAWTO.

We wrap up today's macros with DRAWTO, in Lines 2980-3130 of Listing 1. Again, two parameters are expected; a line is to be drawn from the current cursor position to these coordinates. The POSITION macro is again called to place the cursor as desired. The command for drawing a line is (guess what) DRAW. ICAX1 is set to *12*, and ICAX2 to *0*. To select the color register for the line drawn, we again fish out the result from the most recent COLOR macro call and stuff it into a location called ATACHR ($02FB). Jump to CIO, and your line magically appears.

You should now have a pretty good understanding of how we can get away with the pseudo-BASIC program in Listing 4. I think you'll agree that having these macros around makes it easier and faster to write assembly language programs. When you combine these macro and subroutine library files with the use of the RAMdisk on a 130XE and the great speed of the MAC/65 assembler, you have a powerful assembly development environment.

Tarot cards.

I see a **Boot Camp** column with macros for still more BASIC commands. I see macros for BASIC commands you never expected from Atari (maybe from OSS). I see a discussion of how to do all sorts of disk things from assembly language. I see a return to graphics programming and scrolling. I see you—next month.

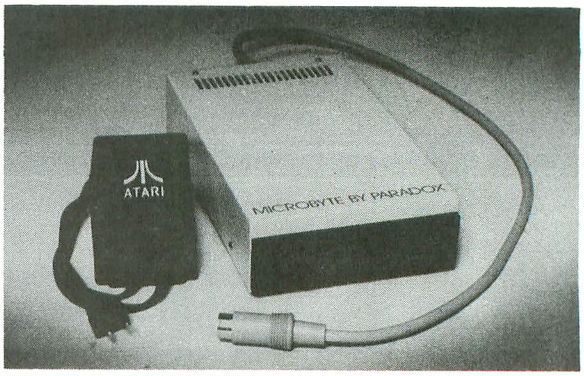
Listing 1.
Assembly listing.

```
0100 ;pseudo-BASIC macros for MAC/65
0110 ;put in file called MACRO.LIB
0120 ;
0130 ;********************************
0140 ;
0150 ;equates needed by macros
0160 ;
0170 EOL =    $9B
0180 OPEN =   $03
0190 GETREC = $05
0200 PUTREC = $09
0210 CLOSE = $0C
0220 DRAW =   $11
0230 ROWCRS = $54
0240 COLCRS = $55
0250 COLOR = $C8
0260 COLOR0 = $02C4
0270 ATACHR = $02FB
0280 ICCOM = $0342
0290 ICBAL = $0344
0300 ICBAH = $0345
0310 ICBLL = $0348
0320 ICBLH = $0349
0330 ICAX1 = $034A
0340 ICAX2 = $034B
0350 CIOV =   $E456
0360 ;
0370 ;********************************
0380 ;
0390 ;SETCOLOR macro
0400 ;
```

```
0410 ;Usage:  SETCOLOR X,Y,Z
0420 ;
0430 ;X, Y, and Z can be values or
0440 ;memory addresses
0450 ;
0460     .MACRO SETCOLOR
0470       .IF %0<>3
0480       .ERROR "Error in SETCOLOR"
0490       .ELSE
0500         .IF %1>4
0510         LDX %1
0520         .ELSE
0530         LDX #%1
0540         .ENDIF
0550         .IF %2>15
0560         LDA %2
0570         ASL A
0580         ASL A
0590         ASL A
0600         ASL A
0610         .ELSE
0620         LDA #%2*16
0630         .ENDIF
0640         .IF %3>15
0650         LDY %3
0660         .ELSE
0670         LDY #%3
0680         .ENDIF
0690       STA COLOR0,X
0700       TYA
0710       AND #15
0720       CLC
0730       ADC COLOR0,X
0740       STA COLOR0,X
0750       .ENDIF
0760     .ENDM
0770 ;
0780 ;******************************
0790 ;
0800 ;POKE macro
0810 ;
0820 ;usage:  POKE X,Y
0830 ;
0840 ;X is an address, Y is a value
0850 ;
0860     .MACRO POKE
0870       .IF %0<>2
0880       .ERROR "Error in POKE"
0890       .ELSE
0900       LDA # <%2
0910       STA %1
0920       .ENDIF
0930     .ENDM
0940 ;
0950 ;******************************
0960 ;
0970 ;POSITION macro
0980 ;
0990 ;usage:  POSITION X,Y
1000 ;
1010 ;X and Y are both values; X
1020 ;can go from 0-319, Y from 0-191
1030 ;
1040     .MACRO POSITION
1050       .IF %0<>2
1060       .ERROR "Error in POSITION"
1070       .ELSE
1080       LDA # <%1
1090       STA COLCRS
1100       LDA # >%1
1110       STA COLCRS+1
1120       LDA # <%2
1130       STA ROWCRS
1140       .ENDIF
1150     .ENDM
1160 ;
1170 ;******************************
1180 ;
```

```
1190 ;OPEN macro
1200 ;
1210 ;Usage:  OPEN chan,aux1,aux2,dev
1220 ;
1230 ;'chan' is an IOCB number
1240 ;'aux1' is a task number
1250 ;'aux2' is the 2nd auxiliary byte
1260 ;'dev' is the name of the device
1270 ; to open, as a literal
1280 ;
1290     .MACRO OPEN
1300       .IF %0<>4
1310       .ERROR "Error in OPEN"
1320       .ELSE
1330       JMP @SKIPOPEN
1340 @DEVICE .BYTE %$4,EOL
1350 @SKIPOPEN
1360       LDX #%1*16
1370       LDA # <@DEVICE
1380       STA ICBAL,X
1390       LDA # >@DEVICE
1400       STA ICBAH,X
1410         .IF %2>255
1420         LDA %2
1430         .ELSE
1440         LDA #%2
1450         .ENDIF
1460       STA ICAX1,X
1470         .IF %3>255
1480         LDA %3
1490         .ELSE
1500         LDA #%3
1510         .ENDIF
1520       STA ICAX2,X
1530       JSR OPENIOCB
1540       .ENDIF
1550     .ENDM
1560 ;
1570 ;******************************
1580 ;
1590 ;CLOSE macro
1600 ;
1610 ;Usage:  CLOSE chan
1620 ;
1630 ;'chan' is an IOCB number
1640 ;
1650     .MACRO CLOSE
1660       .IF %0<>1
1670       .ERROR "Error in CLOSE"
1680       .ELSE
1690       LDX #%1*16
1700       LDA #CLOSE
1710       STA ICCOM,X
1720       JSR CIOV
1730       .ENDIF
1740     .ENDM
1750 ;
1760 ;******************************
1770 ;
1780 ;GRAPHICS macro
1790 ;
1800 ;Usage:  GRAPHICS X
1810 ;
1820 ;X is the number of the graphics
1830 ;mode desired;  add 16 to this
1840 ;number to eliminate the text
1850 ;window
1860 ;
1870     .MACRO GRAPHICS
1880       .IF %0<>1
1890       .ERROR "Error in GRAPHICS"
1900       .ELSE
1910       JMP @SKIPGR
1920 @AUX1
1930      .BYTE 0
1940 @AUX2
1950      .BYTE 0
1960 @SKIPGR
```

```
1970      LDA #%1
1980      AND #$0F
1990      STA @AUX2
2000      LDA #%1
2010      AND #$F0
2020      EOR #$10
2030      ORA #$0C
2040      STA @AUX1
2050       OPEN  6,@AUX1,@AUX2,"S:"
2060      .ENDIF
2070    .ENDM
2080 ;
2090 ;*******************************
2100 ;
2110 ;PRINT macro
2120 ;
2130 ;usage:  PRINT IOCB,address
2140 ;
2150 ;IOCB is channel number to use;
2160 ;'address' is the label of the
2170 ;text string to be printed; the
2180 ;text string must have an EOL
2190 ;character ($9B) at the end
2200 ;
2210 ;if only one parameter, then
2220 ;IOCB is assumed to be 0 (E:)
2230 ;
2240 ;calls subroutine PRINTLINE
2250 ;
2260    .MACRO PRINT
2270      .IF %0<1 .OR %0>2
2280      .ERROR "Error in PRINT"
2290      .ELSE
2300        .IF %0=1
2310        LDX #0
2320        LDA # <%1
2330        STA ICBAL,X
2340        LDA # >%1
2350        STA ICBAH,X
2360        .ELSE
2370        LDX #%1*16
2380        LDA # <%2
2390        STA ICBAL,X
2400        LDA # >%2
2410        STA ICBAH,X
2420        .ENDIF
2430      JSR PRINTLINE
2440      .ENDIF
2450    .ENDM
2460 ;
2470 ;*******************************
2480 ;
2490 ;COLOR macro
2500 ;
2510 ;Usage:  COLOR X
2520 ;
2530 ;X must be a value
2540 ;
2550    .MACRO COLOR
2560      .IF %0<>1
2570      .ERROR "Error in COLOR"
2580      .ELSE
```



```
2590          LDA #%1
2600          STA COLOR
2610          .ENDIF
2620        .ENDM
2630 ;
2640 ;*********************************
2650 ;
2660 ;PLOT macro
2670 ;
2680 ;Usage:   PLOT X,Y
2690 ;
2700 ;X is the x-coordinate
2710 ;Y is the Y-coordinate
2720 ;
2730        .MACRO PLOT
2740          .IF %0<>2
2750          .ERROR "Error in PLOT"
2760          .ELSE
2770           POSITION   %1,%2
2780          LDX #$60
2790          LDA #PUTREC
2800          STA ICCOM,X
2810          LDA #0
2820          STA ICBLL,X
2830          STA ICBLH,X
2840          LDA COLOR
2850          JSR CIOV
2860          .ENDIF
2870        .ENDM
2880 ;
2890 ;*********************************
2900 ;
2910 ;DRAWTO macro
2920 ;
2930 ;Usage:   DRAWTO X,Y
2940 ;
2950 ;X and Y are the endpoints of
2960 ;the line to draw; must be values
2970 ;
2980        .MACRO DRAWTO
2990          .IF %0<>2
3000          .ERROR "Error in DRAWTO"
3010          .ELSE
3020           POSITION   %1,%2
3030          LDX #$60
3040          LDA #DRAW
3050          STA ICCOM,X
3060          LDA #12
3070          STA ICAX1,X
3080          LDA #0
3090          STA ICAX2,X
3100          LDA COLOR
3110          STA ATACHR
3120          JSR CIOV
3130          .ENDM
```

●

Listing 2.
Assembly listing.

```
0100 ;SUBS.LIB file to go with macro
0110 ;library file; required equates
```

```
0120 ;are in MACRO.LIB file
0130 ;
0140 ;******************************
0150 ;
0160 ;subroutine PRINTLINE
0170 ;called by PRINT macro
0180 ;
0190 ;prints up to 160 characters on
0200 ;IOCB number that is already
0210 ;in the X-register
0220 ;
0230 PRINTLINE
0240      LDA #160
0250      STA ICBLL,X
0260      LDA #0
0270      STA ICBLH,X
0280      LDA #PUTREC
0290      STA ICCOM,X
0300      JSR CIOV
0310      RTS
0320 ;
0330 ;******************************
0340 ;
0350 ;subroutine OPENIOCB
0360 ;
0370 ;called by OPEN macro
0380 ;
0390 OPENIOCB
0400      LDA #OPEN
0410      STA ICCOM,X
0420      JSR CIOV
0430      RTS
0440 ;
0450 ;-----------------------------
0460 ;
0470 ;if you use AMAC:
0480 ;
0490 ;some devices you might want
0500 ;to open - add your own if you
0510 ;use other custom handlers;
0520 ;you'll need to define a disk
0530 ;filename in full elsewhere in
0540 ;your program, such as:
0550 ;
0560 ;FILE1 .BYTE "D1:SCORES.DAT"
0570 ;
0580 S    .BYTE "S:"
0590 E    .BYTE "E:"
0600 C    .BYTE "C:"
0610 P    .BYTE "P:"
0620 K    .BYTE "K:"
0630 R    .BYTE "R:"
```

●

Listing 3.
Assembly listing.

```
0100 ;Example 1 for macro library
0110 ;by Karl E. Wiegers
0120 ;
0130      .OPT OBJ,NO LIST
0140      .INCLUDE #D:MACRO.LIB
0150      *=  $5000
0160       GRAPHICS  2
0170       SETCOLOR  2,6,4
0180       POKE  COLOR0+4,6
0190       POSITION  1,1
0200       PRINT  6,TEXT1
0210       PRINT  TEXT2
0220       CLOSE  6
0230 END JMP END
0240      .INCLUDE #D:SUBS.LIB
0250 TEXT1 .BYTE "THIS is a test",EOL
0260 TEXT2 .BYTE "Text window!",EOL
```

●

Listing 4.
Assembly listing.

```
0100 ;Example 2 for macro library
0110 ;by Karl E. Wiegers
0120 ;
0130      .OPT OBJ,NO LIST
0140      .INCLUDE #D:MACRO.LIB
0150      *=  $5000
0160       GRAPHICS  5+16
0170       SETCOLOR  0,1,12
0180       SETCOLOR  4,6,8
0190       COLOR  1
0200       PLOT  20,15
0210       DRAWTO  60,15
0220       DRAWTO  20,35
0230       DRAWTO  40,5
0240       DRAWTO  52,35
0250       DRAWTO  20,15
0260       CLOSE  6
0270 END JMP END
0280      .INCLUDE #D:SUBS.LIB
```

●

Batcher

A setup to help with the "housekeeping" on your Atari 8-bit.

by Mack McLeod

Batching is the process of automating keyboard input by allowing commands to be entered from a disk file. This is useful for DOS procedures requiring many responses, such as loading up a RAMdisk with work files, or using copy, rename and lock to maintain multiple versions of a file.

More than once I've found myself copying the same file twice—and, worse, destroying the most current version of a file with careless keystrokes. Once familiar with DOS, you start wishing for a "setup" procedure to take care of routine housekeeping chores, while you concentrate on that whizbang program. This is where batching takes over, relieving typing boredom and securing valuable files.

Batcher provides this feature, usually found in more expensive DOS systems, for any Atari 8-bit that uses a standard DOS. Since DOS itself is not modified, no compatibility problems arise using either release 2.0 or 2.5. Not only DOS prompts, but any editor input may be automated. Before going into the details, let's look at our interface to the new housekeeper.

I call the batch command files "scripts." They contain the text of commands just as you would enter them, each ending with a return character. While this format is produced by most text editors, a BASIC program to simply INPUTt and PRINT strings to a disk file would suffice for creating scripts.

Batch is invoked by typing a slash (/), followed by the filename containing your commands. **Batcher** then plays out the script, entering commands while you watch them zip by faster than your fingers could ever have responded. If you omit the device ID portion of the filename, **Batcher** inserts D:—making disk 1 the default device. For example: /BACKUP.BAT and /D1:BACKUP.BAT are equivalent commands. No file extender is assumed, allowing use of any desired name; however, an extender of .BAT is commonly used to identify batch files.

You'll soon be hooked on batching DOS commands, and **Batcher** lets you automate even more!

Features.

Batcher can provide input to the E: device from any other CIO-supported device. DOS and BASIC commands may be automated, as well as program prompts using the standard IOCB#0 for input. In BASIC, this is the INPUT statement with no channel number specified. The K: device is unaffected, allowing programs to obtain real keyboard input during a script.

A batch script may switch back and forth between DOS and BASIC modes, copying files, running BASIC or loading machine language programs with no restrictions on the number of commands "batched." For example, a single script could begin in DOS, copy files, invoke BASIC with the B command and run a program, automating some prompts and allowing others from the keyboard. The program could stop, the script return to DOS and perform more tasks, then enter BASIC again, automatically typing *CONT* to continue the program. While watching BASIC and DOS fly around unattended, you may wonder if your Atari still needs you.

I don't have a modem, so I couldn't evaluate the possibilities for bulletin board log-ons and the like.

You may have a script automatically execute when the first DOS menu or BASIC *READY* message appears after booting the system, by naming it AUTORUN.BAT on disk 1 and installing **Batcher** as AUTORUN.SYS.

Batching is turned off by pressing RESET from DOS or BASIC. It may be reactivated using the L option of DOS to load and run the install program. AUTORUN.BAT is not invoked when **Batcher** is loaded in this manner.

Installing *Batcher.*

The BATCH.BAS BASIC program provided creates a binary file named BATCH.OBJ. This machine language program may be run from the L option of DOS, or renamed to AUTORUN.SYS, to automatically install **Batcher** and make use of an AUTORUN.BAT script. A short message and delay informs you of a successful install. If you have an existing AUTORUN.SYS you still need, use the /A option of DOS copy to append **Batcher** to it.

Hints on using *Batcher.*

If you use an AUTORUN.BAT script, remember to boot the system in the mode that the script was written for. Entering DOS commands from BASIC does no damage, but is hardly useful. Consider that you may boot the system in either mode, and make DOS the first command in the script if DOS commands are to be done first. In this way, the script will act correctly—whether or not you boot with BASIC active. When booting without BASIC, this simply results in the *PLEASE TYPE 1 LETTER* message, in response to which the rest of the commands will be valid.

Typing a slash with an invalid filename (from the DOS menu) also results in this message. You need not type one letter, as the correct/filename command will still be accepted. An invalid entry in BASIC causes the usual error message, as would any other invalid command.

Remember that DOS will issue a prompt asking whether or not the program area may be overwritten when doing file copies. If *Y* is answered, subsequent copies will not be prompted again. If your script will return to BASIC and continue a program, it must answer *N* to all such prompts, so as not to erase the program in memory.

Some care must be taken when automating DOS commands, as **Batcher** will relentlessly answer prompts until the end of your script file is reached. Avoid using single-letter filenames—particularly *D*, *Y* or *I*, which are valid DOS commands and responses. Automated deletion of your favorite file is no improvement over making the mistake manually!

The procedure for automating any manual task is to carefully note each step, taking time to understand why it's being done. An efficient automation script may then be designed and tested under the various conditions it is expected to handle. Hastily typing up a batch script on your favorite disk can be a quick and bitter lesson on the efficiency of automated mistakes.

Start with small scripts until you get used to the process. Then amaze your friends with a graphic DOS program, creating scripts from joystick and icons, exiting to DOS to execute the commands and reload itself.

The following BASIC example shows how a program may start a batch script. The same technique will work in machine language.

```
10 REM PROGRAM INVOKED BATCH SCRIPT
20 TRAP 80
30 CLOSE #5
40 OPEN #5,4,0,"D:DEMO.BAT"
50 FOR  I=0 TO 15
60 POKE 1776+I,PEEK(912+I)
70 NEXT I
80 CLOSE #5:STOP
```

The technical notes at the end of this article explain what's happening. Note that the script takes over when Line 80 completes the STOP statement and the BASIC *READY* message appears.

Requirements and cautions.

Batcher uses all of page 6 and 32 bytes at the bottom of the page 1 machine stack. It is unlikely that any program will require so much stack space that it will overlay **Batcher**, but some programs intentionally overwrite this area as a step in copy protection.

You must avoid BASIC subroutines in page 6 or page 1. Remember to disable **Batcher** by pressing *RESET* prior to running such programs.

A batch script may not invoke another. Any text in a batch file is passed back as is.

Channel number 5 (IOCB#5) is generally reserved by **Batcher** during an active script. Programs may use it during batch for nondisk devices such as *K:* or *S:*.

An active batch script is an open disk file to DOS, so it reduces your allowed maximum of three concurrently open disk files to two. Error code 161 will occur if too many disk file opens are attempted.

Programs which add commands to BASIC may override **Batcher** if they do not use the Operating System (OS) tables correctly, or if the slash is used as a keyword.

You must also avoid program responses to the E: device that begin with a slash and are not batch requests. **Batcher** modifies such commands, and will likely return incorrect data to your program—or invoke a batch script if a valid, existing filename is found.

Technical notes.

If you don't wish to delve into the OS contructs in **Batcher**, move on to the BASIC listing and enjoy the power of batch processing that much sooner. The machine language is only about 524 bytes long, so it's not too much of a chore to type in. What follows is a description of the main points of interest. The heavily commented assembly listing may be consulted for more detail.

Batcher consists of an install program and the resident page 6 code, assembled together. The install portion is "editor trap" programming, often used to add commands to BASIC. Simply put, the Atari device handler tables are modified so that CIO calls page 6 instead of the Atari editor for GET operations.

The real editor GET and PUT 1 BYTE routine addresses are obtained from standard locations in the E: handler table and ICPTL field of IOCB#0. These are saved inside **Batcher** to allow nonbatch editor input and the echoing of batch data to the screen. The resident part of **Batcher** is then copied to page 6 and a short message issued. IOCB#5 is closed and copied to location $6F0 inside **Batcher**. This is the area modified by the BASIC sample above. The OS variable WARMST is then tested and, if the system is being booted, an attempt is made to open D:AUTORUN.BAT. The $6F0 area is swapped with IOCB#5 to reflect its success or failure. The install program then returns to DOS.

The $6F0 location is **Batcher**'s private IOCB, which I've dubbed the BIOCB. Its labels follow the convention of ZIOCB, used by CIO. This copy allows **Batcher** to run in both

DOS and BASIC. Normally, invoking BASIC or using the LOAD and RUN commands closes IOCB#5, which would terminate a script using it for disk I/O. The copying of a closed IOCB to the BIOCB initializes it with no file active.

Editor calls now go to **Batcher**, which tests the BIOCB to see if it's open. If so, the I/O subroutine is called, which swaps IOCB#5 and BIOCB contents before doing the batch read. The ZIOCB is also swapped with a save area. This is necessary as CIO is in the middle of an E: device handler operation, with its status maintained in the ZIOCB. The contents of ZIOCB when BATCIO actually calls CIO are irrelevant, as CIO initializes it for each new operation.

Once the batch disk read is complete, the swaps are done again to restore the ZIOCB for CIO and return the new status of batch I/O to BIOCB. The script file byte just obtained is returned to CIO, which then handles it as if it had come from the keyboard. Any error during batch I/O closes the BIOCB, terminating the current script.

When the BIOCB is not open, the Atari editor is called normally, returning each byte of text until RETURN is pressed. **Batcher** then checks the start of the entry for a slash. If one is found, an attempt is made to open the filename indicated. If successful, the E: copy of ZIOCB is modified to look as if no data were entered. This, in effect, erases the slash command and lets the batch data replace it. The first batch byte is then retrieved and returned to CIO, which treats it as the first byte entered on the keyboard.

Although **Batcher** is not a standard CIO calling program, the constructs and variables used are consistent with CIO operation in all Atari models.

Beyond *Batcher.*

The concept of a programmed batch poses many possibilities. Just remember to treat the BIOCB as a regular file, swapping it with IOCB#5 before and after I/O operations. If the BIOCB is open and you wish to terminate the current batch session and begin a new one, close it and reopen with the desired script file.

The ZIOCB may be ignored by your programming, since you're operating outside of CIO, unlike **Batcher** itself, which is inside CIO.

Many embellishments just wouldn't fit into page 6 and, since my design goal was to support DOS and BASIC modes, **Batcher** is missing such goodies as: parameter input to a script, IF logic inside a script, nested scripts, and a message trap allowing IF logic to determine whether or not a batch action was successful. An enterprising programmer could provide these functions, at least in BASIC mode, by expanding the program at MEMLO.

Put your imagination to work; you'll find many uses for batching, beyond the simple automation of DOS commands. In any case, you'll have more time to think about it now, while those file copies buzz along without you! ■

Mack McLeod is an Analyst/Programmer for a major Canadian bank. Even though he works with the largest of mainframes, he's amazed at the power and flexibility of Atari 8-bit machines. He believes they truly provide "Power without the Price." He programs his 130XE primarily in assembly, occasionally in BASIC, and now in C.

The two-letter checksum code preceding the line numbers here is *not* a part of the BASIC program. For further information, see the *BASIC Editor II*, in issue 47.

Listing 1.
BASIC listing.

```
UF 10 REM BATCH.BAS, CREATE BATCH.OBJ
ME 20 OPEN #1,8,0,"D:BATCH.OBJ"
RR 30 FOR I=1 TO 524
PR 40 READ A:PUT #1,A
IT 50 NEXT I
LM 60 CLOSE #1
IN 70 ? CHR$(125);"D:BATCH.OBJ CREATED"
YW 80 END
IW 900 DATA 255,255,0,64,255,65,169,0,133
   ,20,32,224,64,160,0,185,26,3,201,69,24
   0,8,200,200,200
KA 901 DATA 192,33,144,242,96,200,132,205
   ,185,26,3,133,203,200,185,26,3,133,204
   ,160,15,177,203,153,0
BH 902 DATA 1,136,16,248,173,4,1,24,105,1
   ,141,64,65,173,5,1,105,0,141,65,65,173
   ,70,3,24
MG 903 DATA 105,1,141,45,65,173,71,3,105,
   0,141,46,65,169,255,141,4,1,169,5,141,
   5,1,164,205
QQ 904 DATA 169,0,153,26,3,200,169,1,153,
   26,3,160,0,185,16,65,153,0,6,200,192,2
   40,208,245,169
BR 905 DATA 10,133,84,169,15,133,85,162,0
   ,169,250,141,68,3,169,64,141,69,3,169,
   8,141,72,3,138
QR 906 DATA 141,73,3,169,11,141,66,3,32,8
   6,228,162,80,169,12,157,66,3,32,86,228
   ,160,15,185,144
KV 907 DATA 3,153,240,6,136,16,247,165,8,
   208,33,162,80,169,2,157,68,3,169,65,15
   7,69,3,169,1
KW 908 DATA 157,73,3,169,4,157,74,3,169,3
   ,157,66,3,32,86,228,32,202,6,165,20,20
   1,112,144,250
QA 909 DATA 32,224,64,24,96,169,0,170,141
   ,68,3,141,69,3,141,72,3,141,73,3,169,1
   1,141,66,3
FR 910 DATA 169,125,32,86,228,96,160,194,
   193,212,195,200,197,210,68,58,65,85,84
   ,79,82,85,78,46,66
VU 911 DATA 65,84,155,173,240,6,48,38,169
   ,0,141,244,6,141,245,6,141,248,6,141,2
   49,6,160,7,32
EI 912 DATA 174,6,48,17,165,47,32,0,0,165
   ,47,24,144,2,169,155,162,0,160,1,96,16
   2,0,160,1
LE 913 DATA 32,0,0,192,128,240,244,201,15
   5,208,240,160,0,145,36,173,68,3,133,24
   3,141,244,6,173,69
ZF 914 DATA 3,133,244,141,245,6,177,243,2
   01,47,208,208,160,12,32,174,6,160,2,16
   9,58,209,243,240,32
ML 915 DATA 200,209,243,240,27,173,72,3,5
   6,229,40,168,177,243,200,145,243,136,1
   36,208,247,169,68,145,243
VM 916 DATA 200,169,58,145,243,208,8,238,
   244,6,208,3,238,245,6,169,1,141,249,6,
   160,4,140,250,6
RO 917 DATA 136,32,174,6,48,139,165,243,1
   33,36,165,244,133,37,173,72,3,133,40,1
   73,73,3,133,41,76
TN 918 DATA 5,6,140,242,6,32,202,6,162,80
   ,32,86,228,141,31,1,16,12,152,72,169,1
   2,157,66,3
CX 919 DATA 32,86,228,104,168,152,72,160,
   15,185,32,0,72,185,16,1,153,32,0,104,1
   53,16,1,185,144
PN 920 DATA 3,72,185,240,6,153,144,3,104,
   153,240,6,136,16,225,104,168,96,226,2,
   227,2,0,64
```

●

Listing 2.
Assembly listing.

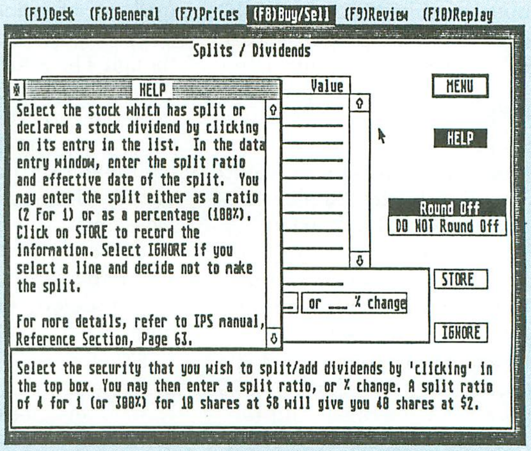
```
;
;       BATCHER
;       FOR DOS 2.x
;
WARMST   =    8          ;boot flag
CLOCK3   =    $14        ;timer
ZIOCB    =    $20        ;pg.0 IOCB,
ICBALZ   =    $24        ;addr-lo,
ICBAHZ   =    $25        ;addr-hi,
ICBLLZ   =    $28        ;len-lo,
ICBLHZ   =    $29        ;len-hi,
ICAX6Z   =    $2F        ;aux6.
ROWCRS   =    $54        ;Cursor row,
COLCRS   =    $55        ;and column.
EPTR     =    $CB        ;Work word,
INDEX    =    $CD        ;and byte.
INBUFF   =    $F3        ;Buffer addr
BATZIO   =    $0110      ;ZIOCB copy
HATABS   =    794        ;IO handlers
IOCB0    =    832        ;IOCB start,
ICCOM    =    834        ;Command.
ICSTA    =    835        ;Status.
ICBAL    =    836        ;Buffer,
ICBAH    =    837        ;address,
ICPTL    =    838        ;Put 1 byte,
ICPTH    =    839        ;vector.
ICBLL    =    840        ;buffer,
ICBLH    =    841        ;length.
ICAX1    =    842        ;aux. info
IOCB5    =    912        ;#5 hdler-id
CIOV     =    58454      ;I/O rtn.
;
;       Find and replace E: table
;
           *=  $4000
           LDA #0        ;timer set,
           STA CLOCK3    ;at 0.
           JSR CLEARS    ;Clear scrn
           LDY #0        ;Find,
FINDIT
           LDA HATABS,Y ;editor,
           CMP #'E   '          ;vector.
           BEQ FOUNDIT ;Found.
           INY           ;Skip this,
           INY           ;3 byte,
           INY           ;entry.
           CPY #33       ;Loop thru,
           BCC FINDIT    ;table.
           RTS           ;Failed.
FOUNDIT
           INY           ;Skip E
           STY INDEX     ;Save index
           LDA HATABS,Y ;Set addr,
           STA EPTR      ;of E: rtn,
           INY           ;table in,
           LDA HATABS,Y ;pointer,
           STA EPTR+1    ;word.
           LDY #15       ;Copy all,
COPYIT
           LDA (EPTR),Y ;16 bytes,
           STA $0100,Y ;of editor,
           DEY          ;vectors,
           BPL COPYIT   ;to PG.1.
           LDA $0104    ;E: GET,
           CLC          ;vector+1,
           ADC #1       ;saved in,
           STA BATMON+1 ;BATCHER,
           LDA $0105    ;for non-
           ADC #0       ;batch,
           STA BATMON+2 ;commands.
           LDA ICPTL    ;Save,
           CLC          ;editor,
           ADC #1       ;put 1,
           STA BATECO+1 ;byte rtn,
           LDA ICPTH    ;for echo,
           ADC #0       ;of batch,
           STA BATECO+2 ;commands.
           LDA #$FF     ;New E:,
           STA $0104    ;GET addr.,
           LDA #5       ;-1 in new,
           STA $0105    ;E: table.
           LDY INDEX    ;New E:,
           LDA #0       ;table,
           STA HATABS,Y ;address,
           INY          ;in device,
           LDA #1       ;handler,
           STA HATABS,Y ;table.
           LDY #0       ;Copy,
COPYBAT
           LDA BATCHER,Y ;code to,
           STA $0600,Y ;its pg. 6,
           INY          ;residence.
           CPY #BATLEN ;Len. must,
           BNE COPYBAT ;be < 256.
           LDA #10      ;Row and,
           STA ROWCRS   ;column,
           LDA #15      ;to put,
           STA COLCRS   ;message.
           LDX #0       ;E: IOCB.
           LDA #INMSG&255 ;Msg-low.
           STA ICBAL    ;Addr.low.
           LDA #INMSG/256 ;Msg-hi.
           STA ICBAH    ;Addr.hi.
           LDA #8       ;Length ,
           STA ICBLL    ;of msg.
           TXA          ;Zero hi,
           STA ICBLH    ;len.byte.
           LDA #11      ;Put data,
           STA ICCOM    ;I/O cmd.
           JSR CIOV     ;Send msg.
           LDX #$50     ;File #5,
           LDA #12      ;close,
           STA ICCOM,X ;command,
           JSR CIOV     ;to init,
           LDY #15      ;BIOCB.
INITBIOCB
           LDA IOCB5,Y ;Copy #5,
           STA BIOCB,Y ;closed,
           DEY          ;status,
           BPL INITBIOCB ;to BIOCB.
           LDA WARMST   ;Skip if,
           BNE WAITABIT ;not boot.
           LDX #$50     ;Bat. iocb
           LDA #AUTO&255 ;Autorun,
           STA ICBAL,X ;script,
           LDA #AUTO/256 ;file,
           STA ICBAH,X ;name.
           LDA #1       ;Enough,
           STA ICBLH,X ;length.
           LDA #4       ;Input,
           STA ICAX1,X ;mode I/O.
           LDA #3       ;Open,
           STA ICCOM,X ;command.
           JSR CIOV     ;try open.
           JSR BATSWAP ;BATSAVE.
WAITABIT
           LDA CLOCK3   ;Wait for,
           CMP #$70     ;a while,
          BCC WAITABIT ;to pass.
          JSR CLEARS   ;Clr scrn.
          CLC          ;Clean,
          RTS          ;return.
CLEARS
          LDA #0       ;Zero the,
          TAX          ;iocb and,
          STA ICBAL    ;address,
          STA ICBAH    ;and,
          STA ICBLL    ;length,
          STA ICBLH    ;for 1,
          LDA #11      ;byte put,
          STA ICCOM    ;command,
          LDA #$7D     ;from A,
          JSR CIOV     ;register.
          RTS          ;Return.
;
INMSG     .BYTE " BATCHER"
AUTO      .BYTE "D:AUTORUN.BAT"
          .BYTE $9B
;
;       BATCHER
;       E: device GET CHAR rtn.
;       (non-relocatable labels,
;        set as pg. 6 offsets.)
;
BATCHER
          LDA BIOCB    ;active BAT?
          BMI BATEDI   ;No-real E:
;
BATGET    =   $0600+*-BATCHER
          LDA #0       ;Set BIOCB,
          STA ICBALB   ;up for 1,
          STA ICBAHB   ;byte GET,
          STA ICBLLB   ;to be,
          STA ICBLHB   ;returned,
          LDY #7       ;in A reg.
          JSR BATCIO   ;Real E:,
          BMI BATEDI   ;if error.
          LDA ICAX6Z   ;saved byte
BATECO    JSR $00      ;CRT echo.
          LDA ICAX6Z   ;restore.
          CLC          ;data,
          BCC BATXIT   ;return.
BATX9B    LDA #$9B     ;return EOL
BATXIT    LDX #0       ;E: IOCB.
          LDY #1       ;Return,
BATRTS    RTS          ;status ok
;
;       Allow real E: input,
;       Watch for / commands.
;
BATEDI    LDX #0       ;E: IOCB
          LDY #1       ;Stat.o.k
BATMON    JSR $00      ;User key.
          CPY #$80     ;Break?
          BEQ BATRTS   ;Yes-exit.
          CMP #$9B     ;EOL ?
          BNE BATRTS   ;No-exit.
          LDY #0       ;Save EOL,
          STA (ICBALZ),Y ;for CIO.
          LDA ICBAL    ;Caller,
          STA INBUFF   ;buff ptr,
          STA ICBALB   ;to pg.0,
          LDA ICBAH    ;and BIOCB
          STA INBUFF+1 ;in case,
          STA ICBAHB   ;needed.
          LDA (INBUFF),Y ;1ST byte,
          CMP #'/      '  ;is BATCH?
          BNE BATX9B   ;No-return
;
;       Start a BATCH script
;
BATGO     LDY #12      ;close,
          JSR BATCIO   ;BIOCB.
          LDY #2       ;Test if,
          LDA #':      '      ;D:, or,
          CMP (INBUFF),Y ;Dn:,
```

```
         BEQ BATINC  ;format,
         INY         ;was,
         CMP (INBUFF),Y ;keyed.
         BEQ BATINC  ;Yes-open.
         LDA ICBLL   ;Only a,
         SEC         ;name was,
         SBC ICBLLZ  ;keyed,
         TAY         ;so we,
BATL1    LDA (INBUFF),Y ;must,
         INY         ;shift,
         STA (INBUFF),Y ;right,
         DEY         ;overlay,
         DEY         ;slash,
         BNE BATL1   ;with D,
         LDA #'D     '         ;and,
         STA (INBUFF),Y ;insert a,
         INY         ;colon,
         LDA #':     '  ;after it,
         STA (INBUFF),Y ;then go,
         BNE BATOPN  ;open file
BATINC   INC ICBALB  ;Skip,
         BNE BATOPN  ;the / to,
         INC ICBAHB  ;filename.
BATOPN   LDA #1      ;Set up,
         STA ICBLHB  ;BIOCB,
         LDY #4      ;for Open,
         STY ICAX1B  ;input,
         DEY         ;cmd and,
         JSR BATCIO  ;try open.
         BMI BATX9B  ;failed.
         LDA INBUFF  ;Tell CIO,
```

```
         STA ICBALZ  ;that the,
         LDA INBUFF+1 ;editor,
         STA ICBAHZ  ;buffer,
         LDA ICBLL   ;is empty,
         STA ICBLLZ  ;ignoring,
         LDA ICBLH   ;/ cmd,to
         STA ICBLHZ  ;replace,
         JMP BATGET  ;with BAT.
;
BATCIO   =   $0600+*-BATCHER
         STY ICCOMB  ;Save cmd.
         JSR BATSWAP ;Swap I/O.
         LDX #$50    ;BAT IOCB.
         JSR CIOV    ;Do I/O.
         STA BATZIO+15 ;Save data
         BPL BATCIX  ;I/O o.k.
         TYA         ;Status,
         PHA         ;saved.
         LDA #12     ;Close #5,
         STA ICCOM,X ;restore,
         JSR CIOV    ;status,
         PLA         ;and fall,
         TAY         ;thru to,
BATCIX   =   *       ;BATSWAP.
BATSWAP  =   $0600+*-BATCHER
         TYA         ;Save Y,
         PHA         ;reg. Set,
         LDY #15     ;copy len.
BATSWL   LDA ZIOCB,Y ;ZIOCB,
         PHA         ;byte save
         LDA BATZIO,Y ;copy old,
         STA ZIOCB,Y ;to ZIOCB.
         PLA         ;Get ZIO,
         STA BATZIO,Y ;save it.
         LDA IOCB5,Y ;File #5,
         PHA         ;saved.
         LDA BIOCB,Y ;BAT IOCB,
         STA IOCB5,Y ;to #5.
         PLA         ;Old #5,
         STA BIOCB,Y ;saved.
         DEY         ;loop for,
         BPL BATSWL  ;16 bytes.
         PLA         ;Restore,
         TAY         ;Y reg,
         RTS         ;and retn.
;
;        Private IOCB
;
BIOCB    =   $0600+*-BATCHER
ICCOMB   =   BIOCB+2 ;cmd.
ICBALB   =   BIOCB+4 ;buffer,
ICBAHB   =   BIOCB+5 ;address.
ICBLLB   =   BIOCB+8 ;buffer,
ICBLHB   =   BIOCB+9 ;length.
ICAX1B   =   BIOCB+10 ;I/O mode
                      BATCHER,
;
BATLEN   =   *-BATCHER ;length
         .END
```

●

Isgur Portfolio System



Select the stock which has split or declared a stock dividend by clicking on its entry in the list. In the data entry window, enter the split ratio and effective date of the split. You may enter the split either as a ratio (2 For 1) or as a percentage (100%). Click on STORE to record the information. Select IGNORE if you select a line and decide not to make the split.

For more details, refer to IPS manual, Reference Section, Page 63.

Select the security that you wish to split/add dividends by 'clicking' in the top box. You may then enter a split ratio, or % change. A split ratio of 4 for 1 (or 300%) for 10 shares at $8 will give you 40 shares at $2.

by Lee Isgur and Mark Skapinker
ELECTRONIC ARTS/BATTERIES INCLUDED
1820 Gateway Drive
San Mateo, CA 94404
(415) 571-7171
High or medium resolution $199.95

by Steve Panak

Although I'm best known for my reviews of recreational software, my life is *not* filled exclusively with fun and games. In fact, in my other identity, I'm a Trust Attorney in charge of pension administration in a large midwestern bank. In this role, I take part in the trading of stocks and bonds held in relatively large pension portfolios. Naturally, I jumped at the opportunity to take a look at a portfolio manager for the ST.

The **Isgur Portfolio System** is targeted at individual investors and money managers who wish to avoid the high-priced, mainframe-based subscription services. In testing it, I took full advantage of the vast differences in sizes of our accounts, which can range from IRAs of less than $100,000 to large corporate pension funds in excess of $30 million. A sampling of these accounts were brought up to date from past statements, then the program was used to chart growth.

When you receive **Isgur**, needless to say, it's *not* ready to run. Like all professional business programs, it must be configured to your system. The number and type of drives is just one factor which influences the program's behavior. And how a program handles the installation process often makes or breaks it.

Busy people who buy software to save time don't want to spend that time (and often much more) figuring out how to use it. In most PC-compatible software, this installation process is handled through a batch program, which copies various files onto various disks, to be inserted into various devices after various and numerous prompts. **Isgur** for the PC has such a program. **Isgur** for the ST does not.

This is not fatal, though. Installation goes fairly smoothly if you read and carefully follow the instructions given in the manual's Appendix C. There's an installation screen on which you specify where the program files are to be located. Answering a couple of simple questions creates a file INSTALL.IS, which you must copy into folders on each of the four disks you'll be using.

Once you finish the installation process, it's time to get down to some serious work. The package itself consists of three programs, each of which is fully capable of standing alone. The **I*S Bridge** is your gateway to **The Isgur Portfolio**, as is **I*S Talk**, a communications program reviewed in **ST-Log** 11, February 1987. You begin all your work sessions from **Bridge**.

However, once you become familiar with **Bridge**, you'll find it much more than simply a stepping stone to the other **I*S** programs. Its premier feature is the memo calendar. This useful utility allows you to make memos and tie them to certain dates and times. You can then read these messages back at your leisure. This is the perfect tool to create a tickler file or an audit trail.

To create a memo, you just click on "MAKE." A window opens which lets you input a date and time, the subject of the memo, and its origin (such as **Bridge**, or **Isgur**, which will automatically create its own memos). This gives the illusion that you are able to input memos from any **I*S** program. This is not the case, and if you wish to make a memo while working in **Isgur**, you must exit to the **Bridge** in order to do it. You may then enter up to six lines of text.

Once your note is entered, all you do is click on the calendar date whose notes you wish to view (the date will be marked as one which has notes attached), then click on "View today's memos." You can also choose to view all memos, or only waiting memos, which are those with dates less than the current date. Marking a memo as "Read" will cause it not to be listed as a waiting memo.

I felt the memo portion of this program was very useful, except for one *slight* drawback. I was not given the ability to search, electronically, for memos with a certain subject, or to search for a character string in the body of the memo. This becomes a more necessary feature as your memo database becomes larger. Finally, you're able to print your memos.

After you're through making memos, it's time to get to the heart of this powerful package. Clicking on the Isgur icon from the **Bridge** starts the **Isgur Portfolio System**. When **Isgur** takes over the display, you have six main menu choices, each with a number of sub-choices. "Desk" provides access to desktop accessories, and "General" allows you to control certain program parameters and customize the package. The "Prices" menu lets you update prices of securities, while "Buy/Sell" does pretty much what you would expect it to do. Finally, the "Replay" option is actually a powerful macro facility, to let you play back a sequence of keystrokes with a single key. Each menu is dropped by touching it with the pointer, or by pressing a function key.

After customizing the program, perhaps choosing to have the prices displayed as decimals rather than fractions, and telling it what type of printer you're using, you choose a portfolio to work with. Unfortunately, at the outset you'll have no portfolios available (other than a sample one created for you). So your logical first step would be the Buy menu.

This window contains a summary of important data (such as number of units and cost) about the securities in your portfolio. In this, as in all other windows, a scroll bar is used to indicate additional items, and dragging it scrolls you through them. Below this display is a similar area, but blank, into which you insert new buys. The bottom of the screen contains other option icons, including useful (although not extensive) help screens. When you enter a security for the first time, a second window opens to allow input of additional security information. This will then apply to all additional units of this security purchased. Although it's tedious to place an entire portfolio into the system, once you've finished the result will be more than worth it.

Sales are handled in a manner very similar to buys, and an additional feature lets you specify an amount of money you need to raise—the program will suggest the sells to meet your goal. These suggestions are based on certain system preferences you enter on the general information screen (such as favoring short-term losses). As with buys, you can review past sales. The Buy/Sell menu also records stock splits, and keeps track of cash and margin accounts.

Using the Prices menu, you're able to update the prices of the securities in your portfolios. This is probably the most crucial aspect of this type of program, as ease of entering current prices determines just how often these prices will be updated. And this determines how recent the values in your portfolios will be.

Fortunately, **Isgur** allows for automatic updates of prices, over the phone line from any of a number of sources. Although The Source sometimes improperly downloads prices, Dow Jones and CompuServe provide good, fast updates. You simply specify the service and various equipment settings, and the program will suck in the latest prices—assuming you have given the correct symbols for each of the securities. You can then check to see if any of your holdings were improperly updated, and these can be posted manually.

The Review menu will probably become your favorite, because it's your gateway to evaluating your portfolio. This menu allows you to examine portfolios, and to create reports and graphs evaluating their worth.

Three reports are available. The first contains portfolio values, and you can choose to look at one or any number of portfolios. This report lists each security, along with its price, cost, and other information. It also gives the total account value. The second reports on securities, either in one or in many portfolios. The third shows a summary of all real gains and losses from sales, again, in any number of portfolios. All these reports can be printed out easily, in an attractive format. Each report screen also has various options, to change the way you view your data.

Isgur also creates graphs. The first compares current value to cost for all portfolios, or for those selected. The second creates a pie graph, showing the diversification of the securities in a given portfolio. While all these graphs were easily created, I was disappointed that there was no way to print them out, other than perhaps with a screen dump.

The manual for the **Isgur Portfolio System** is the best for any Atari (8-bit or ST) program I've ever seen. Contained in a thick, notebook-style ring binder, the 200-odd pages cover each of its three programs. The format is that of a program overview, followed by a detailed reference section (although **Bridge** lacked the second). **Isgur**'s overview contained a five-session tutorial which eased the user into the program.

All sections are indexed adequately. When combined with the many help screens available (which typically refer you to the correct pages in the manual) it's pretty difficult to get lost. Numerous illustrations reassure you that you're not in the wrong place.

This was an extremely easy program to learn and use. I never even bothered with the tutorials, as the many menus and help screens allowed me to rapidly begin work. Clever program design allows the use of both the mouse and the keyboard to control all program functions.

One problem I saw was the inability to easily work with bonds. I had trouble putting both the government and corporate bonds into the portfolios, and I would have liked to see a field that keeps track of interest rates. The preference screens were well designed, to provide a lot of flexibility. The currency exchange was a nice feature, although I have yet to number this among one of my concerns. Perhaps a different situation exists in Canada, the homeland of this program.

Probably my biggest gripe with the **Portfolio** was its memory requirement. It needs an entire 520K of memory to operate. This means first that TOS must be in ROM—not much of a burden, as I think everyone wants TOS in ROM. However, you also cannot, on the 520, have any desk accessories active if **Isgur** is to load. This means the accessory file FUNKEY.ACC, which lets you use function keys to control the programs, cannot be loaded. Many program functions must be loaded from disk when you access them: time wasted.

But on the whole, the **Isgur Portfolio System** is a very nice package, and has a lot more features than I was able to discuss here. Indeed, it is the best business-related Atari program I've ever looked at. It rivals those offered for the PC compatible market (and, as I said earlier, a PC version is available).

Although its price is high, that is the nature of all high-quality business programs. The important thing is that, despite its shortcomings, the **Isgur Portfolio System** helps you get the job done. And, in software such as this, that is the most important quality.

Steve Panak is a Trust Attorney and a free-lance writer living in northeastern Ohio. He holds a B.S. in B.A. and a J.D. He currently oversees computer operations in his department, where he develops software to teach complex legal concepts. In his spare time, he enjoys computer games.

The End User

THIS MONTH:

A good year for the Atari crop, super ST speculation, and response to readers.

Arthur Leyenberger is a human factors psychologist and free-lance writer living in New Jersey. He has been an Atari enthusiast for almost four years. When not computing, he enjoys playing with robotic toys.

CompuServe — 71266,46
Delphi — NJANALOG

by Arthur Leyenberger

Where is that program? Where *is* that program? The one that I, uh, "put away" so I would be sure to find it. Drat! Oh, hi. We're still in the thick of unpacking. Before we moved, I had set aside some 8-bit software that I wanted to talk about this month, and now, of course, I don't have the foggiest notion of where it is. But there's a lot of other information worth discussing here.

We've been in the new house for almost a month now. I've tripled my commuting time—so much for the joys of living in the country. I'm glad the snow is gone. I didn't notice before we bought the place that everything is uphill from here. At least we have a 250-foot driveway, to get a running start.

Speaking of a running start, have you been following the Atari stock prices lately? Just check the AMEX (American Stock Exchange) listing of any daily newspaper. Atari made a public offering of stock in November 1986, with the stock opening at $11 per share. Since then, the price has hovered in the mid-twenty-dollar range, with a high of $29 per share. By Wall Street standards, Atari is doing just fine.

Atari's bottom line.

Atari had a record-breaking year in 1986. Sales came in at $258 million, versus $142 million for 1985. This resulted in a net 1986 income of $45 million, compared to a loss of $14 million for 1985. Clearly, Atari under the stewardship of Jack Tramiel and family has prospered well. Not only does Atari look good on the balance sheet, but Wall Street and the investors think highly of the company, too.

According to Atari, this strong growth in 1986 was due to the ST's steadily gaining market share, in both the United States and Europe. Sales for video games were also strong, as reflected by Atari's proclamation that the 1986 Christmas season was one of the best in recent years. With the IBM PC clone, the new, higher memory STs and a laser printer, Atari seems poised for even more growth in 1987. In Sam Tramiel's words, "We are operating from a position of strength now. We are virtually debt free, and have a solid equity base on which to build the company's future."

Congratulations, Atari, on your successes. We all look forward to your bigger and brighter future.

Miscellaneous ramblings.

Atari Corp. is still planning to introduce a 32-bit-based ST workstation, originally dubbed the TT. Purported to arrive in the second half of the year (which year?), the "super computer" will use a 68020 microprocessor and a 68881 math coprocessor, and will run UNIX System V software—or something close to it. You may recall that, almost two years ago, AT&T (holder of the UNIX copyright and corporate parent of UNIX developer Bell Laboratories) and Atari signed a mysterious joint agreement. Rumor has it that Atari has purchased a license to the UNIX Operating System, but so far, neither Atari or AT&T will spill the beans.

The End User *continued*

This number-crunching computer is said to interface with the DMA port of either the current ST or a future "enhanced resolution" ST. The workstation will use the ST as a front end for the keyboard, graphics and other I/O (Input/Output) devices. With Atari's still relatively small design staff working on such projects as the laser printer and multi-megabyte STs, who is to say whether or not this super ST workstation will debut this year?

From the mail pouch.

It's been almost a year since I've responded to letters in print, so I figured the time was ripe—to boldly go where no computer column has gone before. And we've had some real doozies appear recently. Unlike some magazines, we don't make these things up. Ah, well, it takes all kinds.

Michael Rainy spoiled my day recently with a poison-pen letter. He's upset with me for taking *Consumer Reports* to task in this very column (back in January), for their lack of computer knowledge and slowness to appreciate Atari products. Mr. Rainey is concerned with my "irresponsible vendetta against *Consumer Reports*."

Mr. Rainey, you say writers like me "seriously erode the credibility of **ANALOG Computing** magazine. . . because my observations are grossly inaccurate, misleading. . . [and] biased." Further, you call for "objective discussion in the pages of **ANALOG**." First of all, writers like yours truly are not unbiased; we *are* biased. We get paid for speaking our minds in an informed, understandable and (we hope) entertaining way. I have been both Atari's most loyal supporter and harshest critic, for the past five years. I have the respect of my readers because I *do* "call 'em as I see 'em." What other magazine can you name whose writers are allowed to be so honest?

I am objective in discussing the specifications of a product, but *subjective* when I discuss the use of that product. Is the product easy to use? For whom is it designed? Does it feel right when I use it? Does it accomplish what it set out to? As these questions relate to *my* use of the product or *your* use of the product, we're talking impressions here. Two word processors for the ST can have the same features and cost about the same amount, but, in many cases, the programs have a different "feeling." This is a subjective thing, and the extent to which I can communicate my opinion on it determines how good a writer I am. Further, I always try to make it clear (not necessarily explicitly) when I'm being objective and when I'm being subjective.

I am not blind to the strengths and weaknesses of the Macintosh, Amiga, MS-DOS clones, or other computers. I have written about them and have acknowledged that, in some respects, each is superior to the Atari products. However, I write from an *informed* position when I mention products in this column or in reviews. Knowledge and hands-on use is a much better combination for evaluation of products than is the "consumer-babble" often seen in *Consumer Reports*.

Unlike you, I do remember when *Consumer Reports* either neglected Atari computers altogether, or printed inaccurate or misinformed information. And I said so in print, in this magazine and others (I won't relive the gory details here). For washing machines and toasters, I go to *Consumer Reports*. For computers, I read **ANALOG Computing**, **ST-Log**, *PC Magazine*, *InfoWorld*, *Byte*, *PC Tech Journal* and, yes, even other Atari-specific magazines.

I hope you understand what I'm saying here and can, in the future, appreciate what I'm trying to do. If so, you will avoid the pitfall of thinking I'm on a vendetta. We're all in this great Atari Adventure together—and my aim is to help you get the most out of Atari computing.

Ken Jennings from Milwaukee, Wisconsin had previously been in a stew about my attitude toward Atari. He thought I was (my words) just a mouthpiece for Atari's "vaporware" products, rumors and inaccurate specifications. In a recent letter, he congratulated me for "finally realizing what a jerk Jack [Tramiel] is" and "getting back in sync with the rest of the Atari community."

Ken says he no longer believes rumors about new Atari products, and that (this is a good one) "Atari has become the Grand Central Station for Fictitious Products" by introducing at trade shows products that never make it to market. Ken also goes on to criticize Atari's advertising and inflated sales figures, among other things.

To begin with, Jack is *not* a jerk. Aside from the fact that I never said that in print, my feeling is this: he is a businessman—and quite a good one. I don't always agree with everything he does with the new Atari, but there is no denying that he and his sons have turned the company around. As far as new products announced at trade shows, I learned long ago to simply do my job as a reporter and tell readers what Atari showed.

I *know* that some of these products won't make it to market. So does Jack. One of the reasons they're shown is to gauge reaction to a potential new product. If there's favorable interest, production is encouraged. One thing you soon realize at the shows: a new Atari product will become available when it becomes available.

As far as advertising is concerned, wouldn't you, as head of a company, want your ads to arouse interest, even controversy? Some people feel the Atari ads are in bad taste. Who cares? The only real question is whether or not they're effective in generating interest in the products. I don't always believe everything I read in print advertising. Do you?

One thing I've criticized Atari for (and still do, for that matter) is their minimal customer support after the sale. Every one of the top Atari execs—Jack, Sam, Leonard, Neil Harris, *et. al.*—seems to be a deal maker. That is, they seem more interested in striking a good deal with a supplier, retailer, etc., than in providing the Atari user with new software, new hardware and support. Look at the new ST machines shown at the Winter CES in Las Vegas: more STs, more memory, more expensive. Where is the memory upgrade for the ST user who bought his computer in October 1985, believing Atari's promise of future upgrades and support?

At every Atari press conference I've attended in the last two years, one or more of the Tramiels invariably says something about the user being the most important thing to Atari. Hogwash. It seems, from their behavior, that the *potential* user—the one who may be persuaded to buy an Atari computer—is the most important thing, not the *existing* user. I could go on about the mediocre ST BASIC and ST Logo languages packed with the STs, or the poor manuals accompanying the computers, but you get my point.

Back Issues

All back issues are priced at $4.00 each.

Send your check or money order to **ANALOG Computing Back Issues**, P.O. Box 625, Holmes, PA 19043. MasterCard and VISA orders, call 1-800-345-8112 (in Pennsylvania, 1-800-662-2444).

ISSUE 32 • Supereversion • DOS III to DOS 2 conversion • Color the Shapes • Home-made Translator • Cosmic Defender • 520ST

ISSUE 33 • An Intro to MIDI • Note Master • Syntron • BASIC Bug Exterminator • Assemble Some Sound • C.COM • Mince (ST)

ISSUE 34 • Dragon's Breath • Multiple Choice Vocabulary Quiz • Elevator Repairman • Assemble Some Sound Part 2

ISSUE 35 *(also on disk)* • Hide and Seek • Printers Revisited • Bonk • Turtle 1020 • G:

ISSUE 36 *(also on disk)* • Sneak Attack • Maze War • Nightshade • Solid Gold Input Routine • Rafferty Run

ISSUE 37 *(also on disk)* • Speedski • Index to **ANALOG Computing** (15-36) • Master Disk Directory • Halley Hunter • Bank Switching for the 130XE

ISSUE 38 *(also on disk)* • Color Alignment Generator • Incoming! • DLI Maker • Air Hockey • ST Color Palette

ISSUE 39 *(also on disk)* • Super Pong • Unicheck (updated) • C-Manship Part 1 • Program Helper • Adventurous Programming Part 1 • ST Software Guide

ISSUE 40 *(also on disk)* • Clash of Kings • Micro-Mail • Koala Slideshow Program • Adventurous Programming Part 2 • Mouser

ISSUE 44 *(also on disk)* • RAMcopy! • The 8-Bit Parallel Interface • Arm your Atari • Blast! • D:CHECK in Action! • ST-Log 4

ISSUE 45 *(also on disk)* • Stencil Graphics • Roll 'Em! • RAM DOS XL • LBASIC • Using BASIC XL's Hidden Memory • ST-Log 5

ISSUE 46 *(also on disk)* • Magic Spell •Moonlord • Soft Touch •La Machine • June CES • Launch Code • ST-Log 6

ISSUE 47 *(also on disk)* • DLIs: A minute to learn • Deathzone • BASIC Editor II • The ANALOG Database • DiskFile • ST-Log 7

ISSUE 48 *(also on disk)* • M-Windows • Cosmic Glob • DLIs - Part 2 • Modem Chess • Status Report • ST-Log 8

ISSUE 49 *(also on disk)* • The Atari 8-bit Gift Guide • Brickworks • TechPop • Fortune-Wheel • Smiles and other facial wrinkles • ST-Log 9

ISSUE 50 *(also on disk)* • Krazy Katerpillars • Atari Picture Storage Techniques • Trails in Action! • Scroll-It • Screen Scroller

ISSUE 51 *(also on disk)* • The BBK Monitor • Slither • The Tablet Typist • An Introduction to the Vertical Blank Interrupt (Part 1) • ST Color Tuner

ISSUE 52 *(also on disk)* • The Devil's Doorway • Rambug II • The Vertical Blank Interrupt: Scrolling (Part 2) • Midas Maze • Matching Shapes (ST)

Issues 12, 14, 15, 16, 17, 18, 19, 20, 21, 22 and up are also still available.

Back issues on 5¼-inch disk
$12.95 each, plus $3.00 shipping and handling.
Issues 35 and up are available in this format.

THE #1 MAGAZINE FOR ATARI® COMPUTER OWNERS™
A.N.A.L.O.G.
COMPUTING
P.O. BOX 23 WORCESTER, MASSACHUSETTS 01603

Lyco Computer
Marketing & Consultants

Since 1981

HOURS
Mon.-Thurs. 9AM-8PM
Fri. 9AM-6PM
Sat. 10AM-6PM

SAVE ON THESE IN STOCK PRINTERS!

Panasonic

1080i

$195.00

★ 120 cps Draft Mode
★ 24 cps NLQ
★ Word Process
★ Friction Feed

2-year warranty

PANASONIC

1080i	$195
1091i	$269
1092i	$335
1592	$359
1595	$499
3131	$259
3151	$379

SEIKOSHA

SP-1200AS	$195
SP-1200AI	$195
BP-5420	$1075
BP-5420 ribbon	$12.50
MP-5300 AI	$549
Color Kit	$119
NP 1300AI	$369

STAR MICRONICS

NX10	$195
NL-10	$209
NP-10	$169
SD-10	$259
NB-15	$889
ND-15	$425
NX-15	$339
NR-15	$529
NB 24-5	$729

EPSON

LX86	$209
EX-800	$355
EX-1000	$579
LQ-800	$449
LQ-1000	$659
FX-286e	$519
FX-86e	$369

CITIZEN

120-D	$179
MSP-10	$285
MSP-15	$385
MSP-20	$325
MSP-25	$485
Premier 35	$469

OKIDATA

Okimate 20	$129
Plug n Print	$CALL
120 NLQ	$209
182	$245
192+	$355
193+	$539

LEGEND

808	$159
1080	$199
1380	$229

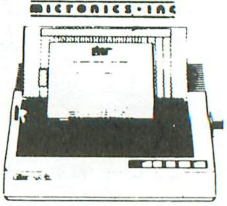
JUKI

6100	$369
5510 w/color	$435
6300	$739

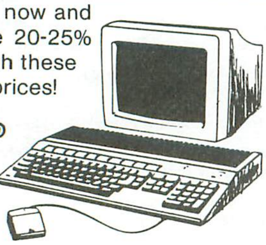
SILVER REED

EXP 420P	$209
EXP 600P	$539

star MICRONICS•INC



NX-10
$175.00
with purchase of printer stand
Limited offer



Buy now and save 20-25% with these prices!

ATARI®

Lyco provides 90-Day warranty on Atari products at no additional cost.

1040 ST Mono SYSTEM $715.95

520 ST Mono SYSTEM $515.95

1040 ST Color SYSTEM $879.95

520 ST Color SYSTEM $699.95 (limited offer)

Box of 10

DISKETTES

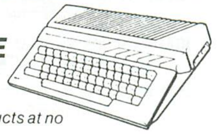
	5¼	3.5
MAXELL		
SSDD	$9.99	$16.99
DSDD	$12.99	$23.99
BONUS		
SSDD	$6.99	—
DSDD	$7.50	—
SKC		
SSDD	$8.50	$14.99
DSDD	$9.50	$16.99
DSHD	$16.99	—
VERBATIM		
SSDD	$9.99	$16.99
DSDD	$12.99	$24.99

Call for generic diskette pricing!

Limited Quantities! *Limited Quantities!*

130 XE COMPUTER SPECIAL $419.95

with
ATARI 1050 DISK DRIVE
and a
STAR NP-10 PRINTER



ATARI

Lyco offers 90-Day Warranty on all Atari products at no additional cost.

130 XE and 1050 Package $255.95

ATARI HARDWARE

INTERFACING

Microprint	$39.95
Xetec AT	$39.95
Atari 850	$109.95
PR: Connection	$59.95

COMPUTERS

130 XE	$125.95
65 XE	$99.95

DRIVES

Indus GT	$185.95
Atari 1050	$139.95
SF 314	$219.95
SF 354	$129.95
SHD 204	$589.95

MONITORS

CALL FOR AVAILABLE MODELS FOR YOUR SYSTEM!

MODEMS

Atari XM 301	$42.95
US Robotics 1200	$189.95
Avatex 1200	$89.95
Avatex 1200 hc	$119.95
SX 212	$89.95

WE STOCK OVER 5000 SOFTWARE TITLES!

SSI

Warship	$34.75
Wizard's Crown	24.75
Gettysburg	34.75
Gemstone Healer	24.75
Phantasie	24.75

Broderbund Software®

Print Shop	$25.75
Print Shop Comp	$22.75
Graphic Lib I, II or III	$15.75
Karateka	$18.75
Bank St. Writer	$29.75

subLOGIC

Pinball	$22.75
Flight Sim II	$31.95
Scenery DSK	$15.95

EPYX

Elddon	$22.95
Karate Champ	$18.95
Koronis Rift	$22.95
Summer Games	$23.95

MicroProse

Conflict in NAM	$25.95
Decision in Desert	$25.95
F-15 Strike Eagle	$22.95
Kennedy Approach	$19.95
Silent Service	$22.95
Top Gunner	$19.95

QUICKVIEW

Zoomracks	$54.95
Zoomracks II	$79.95

OSS™
Precision Software Tools

Action	$46.95
Action Tool Kit	$18.95
Basic XE	$46.95
Basic XL	$36.95
Basic XL Tool Kit	$18.95
Mac 65	$46.95
Mac 65 Tool Kit	$18.95

ACCESS
Software Incorporated

Leader board	$24.95
Tournament #1	$14.95
Triple Pack	$14.95
10th Frame	$24.95

Datasoft Inc.

Gunslinger	$20.75
Conan	$14.95
Alternate Reality The Dungeon	$22.75

TO ORDER
CALL TOLL FREE
1-800-233-8760
in PA
(717) 494-1030
Customer Service
(717) 494-1670
Mon-Fri 9AM-5PM

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

MasterCard

VISA®

Risk Free Policy

In-stock items shipped within 24 hrs. of order. NO deposit on C.O.D. orders. Free shipping on prepaid cash orders within the Continental U.S. Volume discounts available. PA residents add sales tax. APO, FPO and international orders add $5.00 plus 3% for priority mail. Advertised prices show 4% discount for cash, add 4% for MasterCard and VISA. Personal checks require 4 weeks clearance before shipping. We cannot guarantee compatibility. We only ship factory fresh merchandise. Ask about USP Blue and Red Label shipping. All merchandise carried under manufacturers' warranty. Return restriction applicable. Return authorization required. We check for stolen credit cards. We accept purchase orders from educational institutions. All items subject to change without notice

1-800-233-8760

CIRCLE #134 ON READER SERVICE CARD

ST notes

by D.F. Scott

Newsworthy happenings in the ST world.

Ready-Set-Go! at "Set" Stage for ST.

Manhattan Graphics spokesperson Ken Abbott has confirmed that the desktop-publishing software package **Ready-Set-Go!** version 3 is being ported over to the ST. **RSG3** currently ranks near Aldus's PageMaker as one of the top-selling DP packages for the Macintosh.

Currently, Letraset USA Corp. of Paramus, New Jersey—a division of a major Swedish typesetting equipment company—has the marketing rights to the Mac version of **RSG3**. Still, Letraset's Marketing Communications Director, Curtis Dwyer, has told us his company has not yet decided to make a deal with Manhattan Graphics over the rights to the ST version.

A Letraset news release claims **RSG3** uses internal multiple-document word-processing features, so that the text can be composed and laid out using the same program. The Mac version, however, does import text from other word processors; and Ken Abbott states such features will *not* be dropped in the ST version. In fact, says Abbott, the ST **RSG3** should sport enhancements and improvements over the Mac edition.

Letraset states **RSG3** includes the ability to set long articles over several pages in one fell swoop, and also to base page formats upon user-created layout templates, which may be used repetitively. Automatic column layout features kerning and optional hyphenation, as well as text reflow around graphics—so if you had an odd-shaped graphic, text could be set to let the margins flow around that graphic.

Whether or not the ST **RSG3** will use GDOS is unspecified, but it should support PostScript for use with laser printers equipped for it. Final price and release date are also unknown, although the Mac version currently sells for a whopping suggested retail of $395.

GFA A-OK, RSVPs PDQ.

Finally, rejoices the world, there is a BASIC for the ST that *works*. MichTron, which released **GFA BASIC** to North America for GFA Systemteknik in West Germany, is responding in several ways. MichTron Customer Service Director Mark Bruttell tells us that **GFA BASIC 2.0**, and the subsequent revision of the **GFA Compiler**, was released on April 27th with all-new packaging. Version 2.0 mainly features some bug corrections, as well as new commands which allow direct access to the ST blitter chip. The new edition of the compiler will be able, reports Bruttell, to compile permutations of source code that were not compilable with the older edition.

GFA BASIC 2.0 will also feature a completely new manual in paperback form, written by **ANALOG Computing** contributor David Plotkin.

As if that weren't enough, MichTron is soon to be releasing (if not by press time) its second BASIC accessory, **GFA Vector** which, when merged with the interpreter, will provide new commands for *three-dimensional vector-plotted object-oriented graphics.* Graphic objects will be interpreted as whole entities, and will be able to be moved as such. Vector plotting is achieved angularly, by rotation along an axis, or by positioning relative to an axis. Projected list price for **GFA Vector** is $59.95.

New scrolls on 'old themes.

Every once in a while, I come across a computer game that rests peacefully along the back of the store shelf behind **Starglider** and **Sundog**, pretending to be another lackluster product produced on a whim, though hiding within its modest packaging is a real gem. I found two such games recently.

Goldrunner is a game for the person who sneaks to the furthest corner of the arcade, passing up all the modern stereo tri-screen games for the Galaga machine. Its directions, given to you at the beginning of the game, are almost completely unnecessary—in fact, here they are: "**Goldrunner**—The Game of Destruction. If it moves—shoot it! If it is still—blast it!"

Actually, this game is a cross between Galaga and Xevious: you are a spaceship, flying over ground areas called "rings," each of which have energy-generating stations that you are to destroy if you're to earn your way to the *next* ring. Leaving the demo game on awhile, I discovered there are at least 98 such rings in this game, each with extensive graphic detail—although all are variants of each other. Between each ring, there are Galaga-like bonus rounds.

What really makes this game work,

INDEX TO ADVERTISERS

READER SERVICE #	ADVERTISER	PAGE #
102	Abacus Software	8
115	Advan Language Design	55
103	American TV	15
—	ANALOG Publishing	2, 7, 70
116	At-A-Glance	55
113	B & C Computervision	49
118	Banana Software	57
123	Beckemeyer Development Tools	66
104	Computability	16, 17
106	Computer Creations, Inc.	24
107	Computer Mail Order	28, 29
125	Computer Repeats, Inc.	73
110	Computer Software Services	35
—	Delphi	67
105	Electronic One	22
136	Elfin Magic	66
135	G.E. Information Systems	IBC
124	Industrial Machine Products	66
—	Jesse Jones	26
120	L & Y	57
134	Lyco Computer	94
128	Megamax, Inc.	80
126	MichTron	73
129	Microlab	81
114	Microtyme	53
130	Midisoft Corp.	81
127	Newell Industries	74
—	Paradox Enterprises	80
121	Protecto	58, 59, 60, 61
108	S & S Wholesalers	30
112	Side-line Computer	44, 57
131	Sierra Services	81
119	Six Forks Software	57
122	Software Discounters	63
132	Southern Software	82
101	subLOGIC Corp.	IFC
109	TDC Distributors	35
111	Tevex	40
133	Twenty-fifth Century	87

This index is an additional service. While every effort is made to provide a complete and accurate listing, the publisher cannot be responsible for inadvertent errors.

// ST notes *continued*

though, is its fine scrolling of those well designed rings. Even with "turbo-thrusters" engaged, there's absolutely no flicker at all. The scroll is so smooth you'll be taking apart your color monitor in search of the axle grease.

Written by Steve Bak and Pete Lyon of 2-bit Software. Released by MichTron for Microdeal Ltd., P.O. Box 68, St. Austell, Cornwall PL25 4YB, U.K. List price is $29.95.

Trailblazer by Mindscape has a simpler though more unique premise (listen up, XE owners, this is for you, too): you are a soccer ball(!) You roll and bounce along a brightly-colored checkered landscape, in which some colors of checkers may be detrimental to your health, and may even deflate your ego. If you're on blue-green checkers, you're safe; but a turquoise one makes you skid, a yellow one slows you down, a yellow-green one *really* slows you down, a pink one can send you backwards, a white one makes you bounce, and where there's no checker at all—just a black, starry field—you drop into the infamous black hole (or maybe the Voit Void), only to be kicked back out by what may very well be a goalie at the edge of the universe.

The game is played race-course style, so the object of the game is to set the fastest time along a course. **Trailblazer** offers twenty courses, as well as a random-course mode. And, if you think the random courses are unfair, you can *design your own courses* with the included utility program. You may race just to qualify against yourself, or against another person or the computer side-by-side in split-screen mode.

Although the graphic appearance and even the name of this game were obviously inspired by **Ballblazer** from Lucasfilm and the premise may have been borrowed from Synapse's **Rainbow Walker**, this has an original feel to it that places it in the Realm of Uniqueness. By Gremlin Software for Mindscape, 3444 Dundee Road, Northbrook, IL 60062. Retail is $39.95.

The reason I've mentioned **Goldrunner** and **Trailblazer** together in one section is that both include an original musical score, as well as voice-synthesized opening titles. I feel music and sound have been relatively unexplored qualities of microcomputer games up until now, and these games both utilize those qualities imaginatively. //

Batteries recharged.

A final note: within weeks of the acquisition of the Batteries Included brand by Electronic Arts, BI's inventory was shipped to California for immediate distribution through EA's own network. As of now, EA-distributed BI products—for all computer brands—are available through retail outlets. Some distribution companies believe, however, EA may decide to distribute BI products exclusively, thus locking out the distribution channels of the former Batteries Included. //

"Ordinary online services left me flat..."

"With GEnie™ I discovered vast new horizons."

$10.00+ per hour

$5 per hour

There's only so far you can go with most ordinary online information networks. But with GEnie—the General Electric Network for Information Exchange—there's virtually no end to the exciting new interests you can explore. Discover the value of GEnie's vast range of services and explore new territory with guides from our Special Interest Groups.

Join the friendly crew with GEnie's Atari† ***RoundTable™ Special Interest Group*** for exciting and informative discussions. Benefit from the expertise of experienced individuals. Compare notes on hardware, software, games or any other topic you choose. And, for the simple cost of downloading, collect thousands of valuable public domain software programs for your personal computer. Over 1000 new programs are added to our libraries every month. And remember, UPLOADS ARE FREE on GEnie!

No extra charge for 1200 Baud.

Make new friends, schedule meetings, gather facts and figures, set up travel plans, get the news, be entertained and even shop online for a fraction of what other information services charge, because GEnie's non-prime rate for 300 or 1200 baud is only $5 an hour. GEnie always comes out ahead of the competition in savings, in some cases by as much as 60%.

You get even more for your money.

In addition to user support groups, you can meet new people with ***GEnie's Live Wire™ CB Simulator.*** Exchange messages with ***electronic mail*** service. Find the best vacation deals and make your reservations with ***American Airlines travel service.*** Shop at home with ***American Express Merchandise Shop.*** Learn the fun way with a subscription to ***Grolier's electronic encyclopedia.*** Play classic and ***multiplayer games.*** Use GEnie's ***Financial Services*** to track stock market quotes, check market indicators and maintain an automatically updated personal portfolio online. All this ***plus*** new services added each and every month!

Map out your own voyage of discovery with GEnie as your guide. . .enrich your tomorrows by signing on with GEnie today.

Sign up from your keyboard today.
4 Easy Steps:

1. Have your major credit card or checking account number ready.
2. Set your modem for local echo (half duplex)—300 or 1200 baud.
3. Dial **1-800-638-8369.** When connected, enter HHH
4. At the U#= prompt enter **XJM11877,GEnie** then RETURN.

Need help or more information? No modem yet? We can help. In U.S. or Canada call **1-800-638-9636.**

† *Atari is a registered trademark of Atari Corporation*

Backed in the GE Tradition

GEnie is brought to you by General Electric, whose reputation is built on attention to detail and customer satisfaction. We're continuing that tradition with GEnie.

Compare & Save*	**Services**						**Pricing**			
	Travel & Shopping	*SIGs/User Groups*	*CB & Mail*	*Financial Services*	*News*	*Games*	*Registration Fee*	*Monthly Minimum*	*Non-prime time rates* *300 baud*	 *1200 baud*
The Source	X	X	X	X	X	X	$49.95	$10.00	$8.40	$10.80
CompuServe	X	X	X	X	X	X	$39.95	none	$6.00	$12.50
GEnie†	**X**	**X**	**X**	**X**	**X**	**X**	**$18.00**	**none**	**$5.00**	**$5.00**

**Basic rates and services shown in effect 2/87. †Non-prime time applies Mon.–Fri., 6pm–8am local time, all day Sat., Sun., and nat'l holidays. Subject to service availability. Additional charges apply for 2400 baud and financial services.*

ATARI CORP.
NOW ONLINE!

GEnie™

Stay online longer, for less.

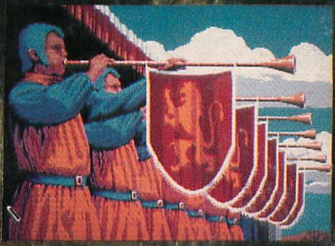
INFORMATION SERVICES

General Electric Information Services Company, USA

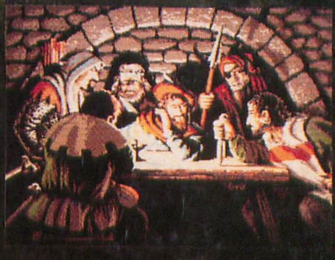
GEnie rates and services subject to change. Uploads are free during non-prime hours at 300 or 1200 baud. Some services offered on Genie may include additional charges.

CIRCLE #135 ON READER SERVICE CARD

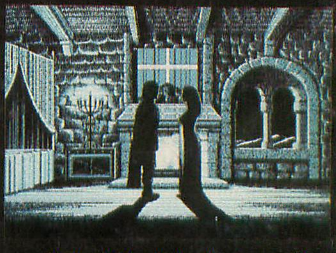
TO THE VICTOR GO THE SPOILS
"What an experience!
Like being in a movie!"
Computer Entertainer, Dec. 1986
CINEMAWARE™
PRESENTS
DEFENDER OF THE CROWN
NOW PLAYING AT A SOFTWARE DEALER NEAR YOU



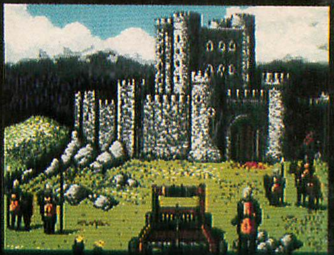
Actual Atari ST screen!



Actual Apple IIGS screen!



Actual Macintosh screen!



Actual Commodore 64 screen!

Exclusively distributed by Mindscape. 3444 Dundee Road Northbrook, IL 60062
Call Toll Free in Continental U.S. (Except IL): 1-800-443-7982 Illinois: 1-312-480-7667