# ST-LOG

THE ATARI ST MONTHLY MAGAZINE

U.S.A. $3.50
CANADA $4.75

JUNE 1987

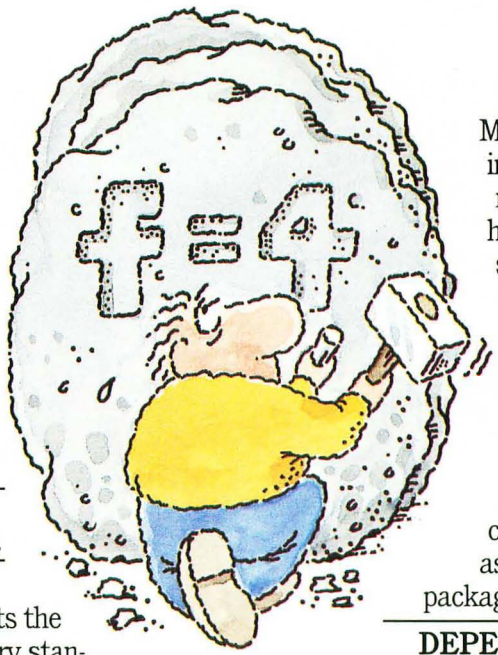
ISSUE 15

**SPECIAL PROGRAMMING ISSUE:**

**Language overview**
**ST Labelmaker**
**PrinterFont**

**NEW COLUMN:**
**GFA Basics**

# MARK WILLIAMS C. AN ENLIGHTENING DEVELOPMENT FOR ATARI ST USERS.



If you've tried your hand at developing applications on the Atari ST, you know the problem. Programming tools aren't only hard to come by, they're hard to use. One might even say primitive. But now for some enlightening news: you can have all the power, portability and versatility of the C language from a leader in professional C programming tools, Mark Williams.

## BRING YOUR PROGRAMMING UP TO SPEED.

The Mark Williams C compiler produces fast, dense code and supports the complete Kernighan & Ritchie industry standard C. You'll have access to GEM's AES and VDI libraries for programs using graphics, icons and the Atari mouse. And Mark Williams C lets you take advantage of the full 16 megabytes in Atari's 68000 microprocessor.

## STREAMLINE DEVELOPMENT WITH POWER UTILITIES.

Mark Williams C is loaded with everything you'll need for professional development. Bring the power of the UNIX environment to your Atari ST with our Microshell Command Processor including pipes, I/O redirection and more. Edit your program with the highly acclaimed MicroEMACS full screen editor. Accelerate and simplify compiling with *make* which finds and recompiles only those modules affected by your changes. Then, when you're ready for debugging, call on our db Symbolic Debugger with single step, breakpoint and stack traceback functions. Over 40 commands, including a linker and assembler, provide a total development package for your Atari ST.

### NEW VERSION 2.0 FEATURES

- ★ 50% faster compile time
- ★ Improved code: 20% faster dhrystone
- ★ Full access to AES/VDI libraries now with 200 new pages of documentation and dozens of examples
- ★ New integrated edit-compile cycle: editor automatically points to errors
- ★ Source and object code for RAM disk cuts compile time in half
- Complete Kernighan & Ritchie C plus extensions
- Microshell Command Processor, a powerful UNIX style shell
- MicroEMACS Full Screen Editor with commented source code
- Make Program Building Discipline
- Complete symbolic debugger
- Assembler, linker and archiver
- Powerful Utilities Package: one-step compiling/linking with cc command, egrep, sort, diff and more
- Over 600 pages of documentation including 120 sample C programs
- Not copy protected

Sieve benchmark*:
Compile and link time in seconds.

| | |
|---|---|
| Mark Williams C | 46.0 |
| Megamax | 78.5 |

**MARK WILLIAMS C FOR THE ATARI ST: $179.95**
**60 DAY MONEY BACK GUARANTEE**

*Sieve benchmark from Byte, 1983, done on one double-sided floppy disk and included RAM disk.

## DEPEND ON A NAME WITH A HISTORY OF PERFORMANCE.

Mark Williams C for the Atari ST is part of our growing line of C compilers. A line that includes the C compiler chosen by DEC, Intel, Wang and thousands of professional programmers. Now our Atari C compiler is earning its own reputation:

"Finally a great C compiler that exploits the power of the ST"–Sigmund Hartmann, President, Atari Software Group

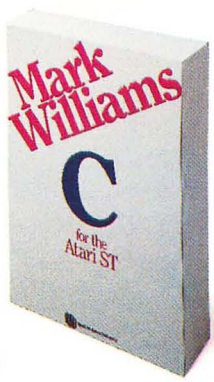
"The all-around best choice for serious software development on the ST."–Douglas Weir of *ANALOG COMPUTING*

## GET WHAT YOUR ATARI ST HAS BEEN WAITING FOR.

Mark Williams C is just what your Atari ST was made for: powerful, professional programming. So now that you can have Mark Williams C for just $179.95, what are you waiting for?

Ask your Atari dealer about Mark Williams C or order today by calling **1-800-MWC-1700.***

*In Illinois call: 312-472-6659




**Mark Williams Company**

1430 West Wrightwood, Chicago, Illinois 60614



CIRCLE #101 ON READER SERVICE CARD

THE ATARI ST MONTHLY MAGAZINE

# ST-LOG

**ISSUE 15**
**JUNE 1987**

## FEATURES



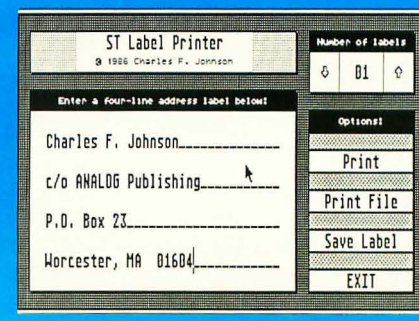
## REVIEWS



## COLUMNS




21




39

77

**ST-Log** (ISSN 0890-9601) 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 **ST-Log**, P.O. Box 625, Holmes, PA 19043. 

# 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

ANALOG COMPUTING

P.O. BOX 23, WORCESTER, MA 01603

# MOVING?

## DON'T MISS A SINGLE ISSUE

Let us know your new address right away. Attach an old mailing label in the space provided below and print your new address where indicated.

**DO YOU HAVE A QUESTION ABOUT YOUR SUBSCRIPTION?**

Check the appropriate boxes below:

- ☐ New subscription. Please allow 4 to 8 weeks for your first copy to be mailed.
- ☐ Renewal subscription. Please include a current address label to insure prompt and proper extension.
- ☐ 1 year — $28.00. This rate limited to the U.S. and its possessions.
- ☐ Payment enclosed. ☐ Bill me.

MAIL TO: **ST-LOG,** P.O. BOX 625, HOLMES, PA 19043

Name _____

Street Address _____

City _____ State ___ Zip ___

ATTACH LABEL HERE

(IF LABEL IS NOT HANDY, PRINT OLD ADDRESS IN THIS SPACE.)

## ST-Log Staff

*Editors/Publishers:* Michael J. DesChenes, Lee H. Pappas. *Managing Editor:* Diane L. Gaw. *East Coast Editor:* Arthur Leyenberger. *Midwest Editor:* Matthew J.W. Ratcliff. *West Coast Editor:* Charles F. Johnson. *Contributing Editors:* Ian Chadwick, Arnie Katz, Bill Kunkel, Maurice Molyneaux, Steve Panak, D.F. Scott, Joyce Worley. *Cover photography:* Jim Langone. *Technical Division:* Charles Bachand, Clayton Walnum, Douglas Weir. *Production Editor:* Kathy Wiesner. *Graphics:* Connie Moore. *Type production:* Edythe Stoddard, Lisa Openshaw. *Advertising Manager:* Michael J. DesChenes. *Accounting/Circulation:* Robin Levitsky. *Production/Distribution:* Lorell Press, Inc. *Contributors:* Richard J. Bourne, Frank Cohen, Rusel DeMaria.

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



## Where to write

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

Correspondence regarding subscriptions, including problems and changes of address, should be sent to: **ST-Log**, 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.

## Advertising Sales

| | |
|---|---|
| **East Coast** | Gerald F. Sweeney & Associates — (212) 242-3540<br>P.O. Box 662, New York, NY 10113 |
| **West Coast** | J.E. Publishers Representatives — Los Angeles: (213) 467-2266<br>San Francisco: (415) 864-3252, Chicago: (312) 445-2489,<br>Denver: (303) 595-4331<br>6855 Santa Monica Blvd., Suite 200, Los Angeles, CA 90038 |
| **ST-Log** offices | Michael DesChenes, National Advertising — (617) 892-9230 |

Address all advertising materials to: Michael DesChenes — Advertising Production, **ST-Log**, 565 Main Street, Cherry Valley, MA 01611

## Permissions



## Subscriptions

**ST-Log**, P.O. Box 625, Holmes, PA 19043; or call toll-free: 1-800-345-8112 (in Pennsylvania, 1-800-662-2444). Payable in U.S. funds only. U.S.: $28.00-1 year; $52.00-2 years; $76.00-3 years. Canada: $36.00-1 year; $68.00-2 years; $99.00-3 years. Foreign: $39.00-1 year; $72.00-2 years; $104.00-3 years. 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

Computers come and go, but the languages remain.

It's anybody's guess what hardware will look like ten years from now, but one thing is pretty certain: if you want to, you'll be able to program in BASIC on it. Or Pascal, or C, or FORTRAN.

But what kind of BASIC? Languages are changing, even though we're calling them by the same names. Sophisticated operating systems have grown so fast that the languages have had to adapt willy-nilly. C source code written to use GEM's windowing and other features almost looks like a new language. The situation is "worse" with the similar systems used by the Macintosh and Amiga.

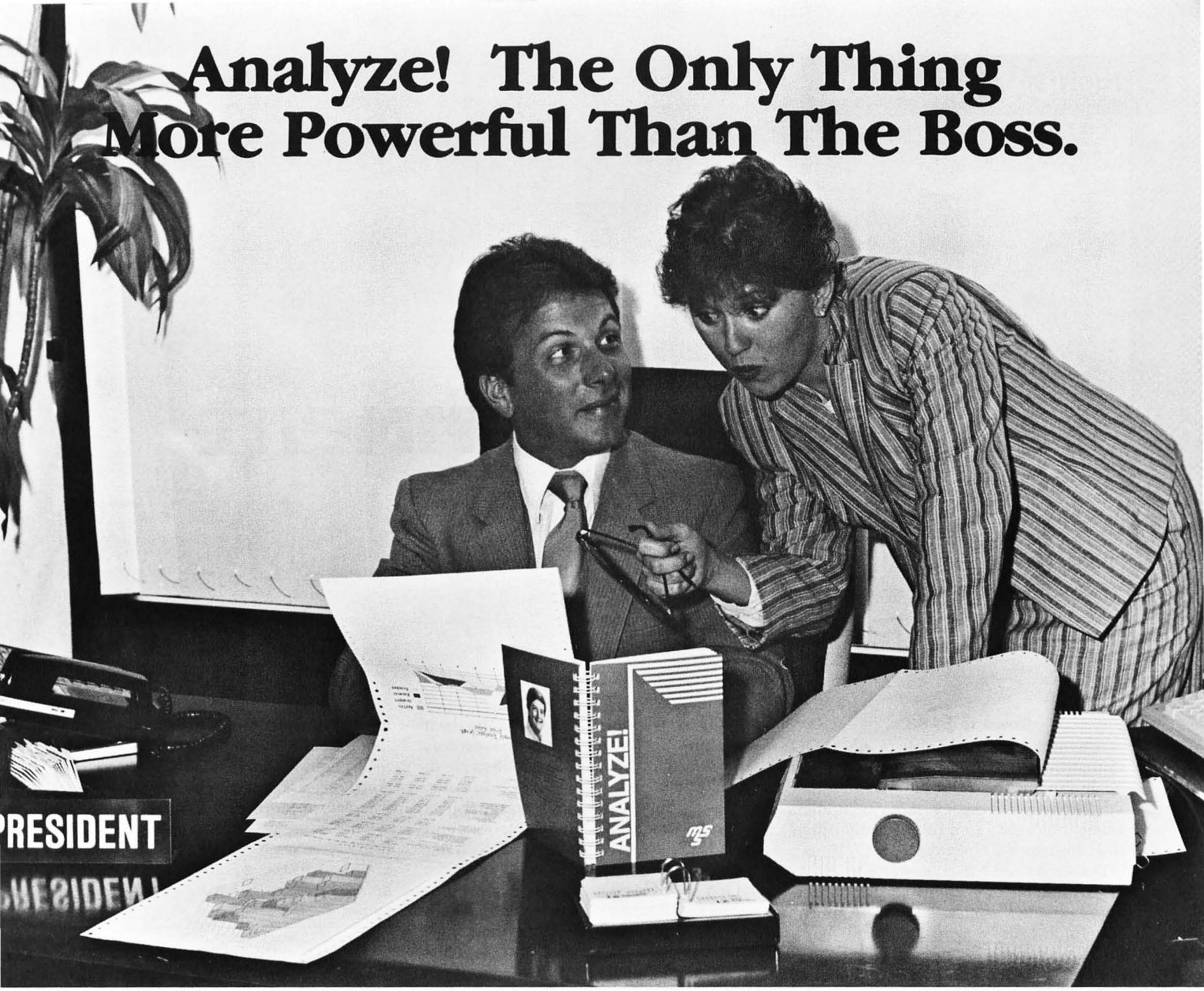
We'll be seeing more windows and menus in the future, not less. Will we also see more and more specialized (and haphazard) language dialects to control them, or will we eventually have environments that look absolutely "normal" to the programmer? A language system that would let you set "standard output" to mean "a window," and then forget about it, would make it a lot easier to write those full-featured word processors and database managers and wargames we've been waiting for. It's been a long time since programmers were forced to handle keyboard debounce and scan routines. Is there any real difference between debouncing a keypress and processing a "resize window" message?

It's interesting to look at the many fine compilers and interpreters available for the ST and see how they've begun to come to grips with this problem in very different ways. The C compilers have by and large ignored it. But that's what you would expect from C. After all, if you want to deal with GEM and TOS at this level, you should be able to. Other languages have taken the same approach, but less appropriately. You might expect BASIC to show the most innovation in this area, but the exciting things so far seem to have come from a couple of Pascal packages. Then there are the assemblers. You don't have much choice about how to interface with the operating system in assembly language—you take it as it is. Which makes it seem all the more bizarre that almost every assembler available for the ST comes smothered in a package of GEM-based editor/supervisor programs that end up being more hindrance than help in serious software development.

Not very long ago I learned to program using Waterloo Pascal on a creaky IBM mainframe timesharing system. It was called WYLBUR, although it usually behaved more like Mister Ed. Things have changed greatly since then. The ST is a much nicer machine to work with and, since one has it to oneself, practically as powerful. Whether the same becomes true of languages will depend less on integrated editors and fanciful extensions than it will on serious efforts at fitting the language to the environment and leaving the programmer to worry about more important things. If this does happen on the ST, it will be another first for this revolutionary series of computers.

Douglas Weir
*Technical Editor*
**ST-Log**

# Analyze! The Only Thing More Powerful Than The Boss.



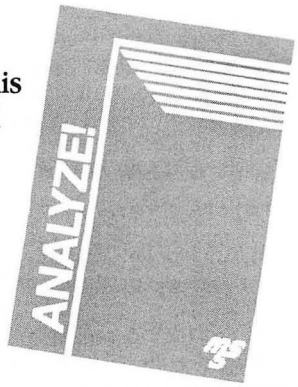

**Analyze!** is a powerful electronic spreadsheet that enforces the reasons you have an Atari ST.

It provides compatible sophistication that refreshes the dreary "monkey see, monkey do" products of today. Controlled via mouse, keyboard, or both, **Analyze!** will work with the major spreadsheet programs popular in the work place.

Take your Atari 520 ST or 1040 ST to the limit with the full featured Macro language, outrageous 4 or 8 color graphs, sorts and extensive mathematical functions.

**Analyze!** is easy to learn, supported by an informative reference manual and our expertly trained Technical Support Division.

Ask for a dealer demonstration of this fine product, or call our Technical Question Line at (305) 790-0772




Sales Order Line: (800) 327-8724
For a dealer near you call:
Brown Wagh Publishing
(800) 451-0900

**MICRO-SYSTEMS SOFTWARE, INC.**

**12798 West Forest Hill Blvd. West Palm Beach, FL 33414**

**MICRO-SYSTEMS SOFTWARE, INC. ...7 years of quality software and still growing strong!**

CIRCLE #102 ON READER SERVICE CARD

# Reader comment

### Down to BASICs.

Having been a subscriber to **ANALOG Computing** magazine for many years when I owned an Atari 800XL, now that I'm the happy owner of an ST computer, I must confess: I'm a little disappointed in regard to ST coverage.

On occasion, I had to write to **ANALOG** for help or technical information. Your staff always answered me promptly, for which I am very grateful and thank you for your reader support.

I received two free **ST-Log** magazines with my two remaining months' subscription to **ANALOG**, and I feel **ST-Log** needs to improve with age.

I am mailing in a check for a year's subscription to **ST-Log**. I'm subscribing to **ST-Log** merely based on **ANALOG**'s past excellent performance. I am subscribing with the hope that **ST-Log** will continue to improve.

Frankly, I don't care about programming *in C, Modula-2*, Pascal, or what have you. I still enjoy programming in BASIC, and I feel there are many like myself, who would like to see more coverage in BASIC programs, short graphics demos and utilities.

Very few graphics demos have appeared in any ST magazine in BASIC, or even tried to tap the excellent graphics capabilities of the ST.

I can buy all the game programs I want, and wish ST-related magazines would issue fewer game programs. I believe a sad mistake is being made by all the ST magazines in this respect, other than *ST Applications*, which I feel is the best magazine for the ST. Not all ST owners are youngsters. Many ST computers are in the hands of mature adults. Sure, I like games. . . but I also enjoy good graphics programs, fancy and interesting titles and sound effects, as well as good utilities.

I would like to see continued coverage of GFA BASIC and maybe some short programs in GFA, so I'll be tempted to go out and buy. I realize other subscribers have other choices in languages, and you must please all segments of your reader population. I realize C, Pascal and Modula-2 are here to stay, but, gosh, fellows. . . help keep BASIC alive. Even a few short programs each month would be a big improvement.

As I previously stated, **ANALOG Computing** has always been a super magazine with respect to the 8-bit Atari computers. Now, with your new **ST-Log** magazine, I am hopeful you will expand and realize a lot of people out here still like to tinker with BASIC.

There are many features I enjoy in **ST-Log**, such as the **Reader comment** and reviews sections. I am hoping, by making myself heard, you will better know what your audience wants to see more of in upcoming issues.

Respectfully,
Mario Sala
Torrance, CA

*We've tried to provide a good sampling of BASIC programs for the ST; in our first separate **ST-Log**, January 1987, you'll find **Spellbinder**. February brought **Appointment Calendar**, and—while our March "entertainment" issue had only an ST BASIC game—**Escher Cubes** graced the April pages. Last month's BASIC offering was **Music Steps and Triads**.*

*This issue, we begin a column on GFA BASIC (see page 73). ST BASIC programs are present, too: **PrinterFont** and **ST Labelmaker**. How's* that *for service with a smile? —Ed.*

---

### Cover copy.

*Our May staff listing attributed that cover's computer graphics to André Molyneux. . .wrong! The art was by Vincent Reynolds. Our apologies, Vincent. —Ed.*

---

### Getting in Step.

I'm writing this letter in regard to the **Step 1** articles in the past issues of **ST-Log**.

First, let me give you a whole-hearted thank-you for including a series for the beginner. You would not *believe* how many magazines I've purchased over the past three months (not to mention the countless masses of books), in search for something that doesn't require a doctorate to understand all the terminology of the computer. Mr. Molyneaux's articles took the time to explain all the jargon that no one else thought was important enough to mention.

My husband and I only recently purchased a 520ST, and both of us are novices, at best, when it comes to computers. When we found our first **ST-Log**, I honestly thought that it was like all the

# 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 1000 | 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 |

### 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 |

| | |
|---|---|
| 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 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.

rest. . .until I found the introductory **Step 1**. It helped me start understanding what everyone else is talking about.

I'm proud to say we're now subscribers to **ST-Log**—so we won't miss an issue. Please keep these and other articles like them in future issues.

Sincerely yours,
Jennifer Coleman
Chula Vista, CA

**Mail-order mayhem.**

I would like to first say that **ST-Log** has been a welcome addition to my reading library. I have found **ST-Log** very informative and hope to possibly see some articles on, or using, Personal Pascal.

Now. . .There is a practice among some of **ST-Log**'s advertisers to advertise a product or program months before it is ready to be released. I have two such products ordered at this time. One has been ordered since November 1986, and the other since February 1987. Both of these orders have been paid, and my money has been tied up for months.

Both vendors are prominent in the ST products line and have produced some very good programs and utilities for the ST. However, I feel that this practice is unfair to the public. Maybe all advertisers should be required to put an availability date in their ads.

I realize that prerelease info is important to sales, but the public should be *informed* as to the availability dates. Due to the lack of local support by Atari Corp. or Atari dealers, I must buy most of my programs mail order. It should be kept fair and aboveboard, for those of us who must mail order.

Sincerely,
George R. Millward
York, PA

*We understand—and agree with—your point of view. To be fair to the mail-order companies, though, you should know that ad copy is sometimes submitted three months before you finally see it. They rely on release dates given to them by the software producers and distributors. And, as any software author will tell you, Murphy's Law rules programming. Most mail-order companies are as reliable as possible. We hope they (and producers) are listening—and trying to improve matters.*

*Oh—you may enjoy* ***Animation with Pascal*** *on page 17.* —*Ed.*

**Promoting Atari—*us?***

I have been an **ANALOG Computing** subscriber for over three years. Since the introduction of the STs, I sometimes wonder if **ANALOG Computing** and **ST-Log** have just become Atari rags, similar to the new *Infoworld*.

In the past, the problem with **ANALOG Computing** was the lack of current news and developments in the Atari world. When Atari underwent many changes, I—and probably many owners—had to read *Infoworld* (the non-Apple/IBM version) for the latest information. **ANALOG Computing** was simply too slow in delivering the information. Currently, this problem is corrected, and the magazine is bringing the latest news and developments.

However, since the introduction of the STs, **ANALOG Computing** has become more like a promotion rag of the new machines. ANALOG Publishing has never emphasized the problems and limitations of the STs, nor have the magazines been very critical about Atari Corp.

First off, Atari Corp. simply goofed in choosing the microprocessor for the STs. It should have been National Semiconductor's 32016 or 32032. The problems of memory management and floating point would have been solved. In addition, ST owners would have the benefits of virtual memory. A graphic chip should have been in the ST in the first place, as well. A system based on the NS32000 series would have outgunned the Amiga and IBM PC AT (with floating-point option). Who would buy such a system? Engineers, hackers, programmers, students and business people are just *some* of the possible consumers. The cost of such a system would be a maximum of a couple hundred dollars more than the present ST prices. **ANALOG Computing** and **ST-Log** have never pointed out the limitations of the vaunted MC68000.

Second, **ANALOG** and **ST-Log** never emphasized the problems of GEM. GEM is horribly coded (great for compatibility, but not on the MC68000, which does not support high-level languages) and pathetically slow, not to mention very buggy. GEMDOS is in a sad state. The ST disk I/O speeds are pathetic. Part of the problem may be attributed to the IBM for-

*(continued on page 64)*

# Language overview

Much has changed since the early days of microcomputing, when BASIC and assembly were all that programmers had to choose from in the way of languages. An ST owner can explore artificial intelligence with LISP or PROLOG, draw on three decades' worth of scientific programming expertise with FORTRAN and its libraries, or chase pointers with C and a copy of Kernighan and Ritchie. Here's a list of all the languages we could find currently available for the ST series. For the most part, only products available through a U.S. distributor are listed, though there are a couple of exceptions. We've also included thumbnail sketches of the more common languages.

## ASSEMBLY

Each processor has its own assembly language, describing the operations it can perform. Assembly language uses names and syntax not too far removed from the forms which the machine handles directly, but which are, at the same time, much more easily understandable to human beings. By a relatively simple process, assembly language is translated into machine codes—the only language the machine understands—and describes the only things the machine can really "do."

**A-Seka ..... $34.95**
The Catalog
544 Second St., San Francisco, CA 94107
(415) 957-0886

**Assembler ..... $59.95**
(requires Pecan P-system—$99.95)
Pecan Software Systems, Inc.
1410 39th St., Brooklyn, NY 11218
(718) 851-3100

**AssemPro ..... $59.95**
Abacus Software
P.O. Box 7219, Grand Rapids, MI 49510
(616) 241-5510

**AS68 Assembler ..... $300.00**
(included in the ST Developer's Kit)
Atari Corp.
1196 Borregas Ave., Sunnyvale, CA 94086
(408) 745-2000

**DevpacST ..... $129.95**
HiSoft
The Old School, Greenfield
Bedford MK45 5DE, U.K.
(0525) 718181

**GST-ASM ..... $59.95**
The Catalog
544 Second St., San Francisco, CA 94107
(415) 957-0886

**MCC Macro Assembler ..... $79.95**
Tenchstar Inc./Metacomco
26 Portland Sq., Bristol BS2 8RZ, U.K.
(0272) 428781

**OS/9 Assembler Linker Debugger. . . $150.00**
(requires OS/9 operating system)
Microware Systems
1900 NW 114th St., Des Moines, IA 50322
(515) 224-1929

## C

C is the result of an evolutionary process (B was an immediate ancestor) at Bell Labs in the early 1970s. It combines many features of a Pascal-like high-level language with others which make it seem at times more like a very powerful assembler. C was designed as a convenient and easy tool for operating system development, although nowadays it's used for just about anything.

**Alcyon C ..... $300.00**
(included in the ST Developer's Kit)
Atari Corp.
1196 Borregas Ave., Sunnyvale, CA 94086
(408) 745-2000

**GSTC Compiler ..... $79.95**
The Catalog
544 Second St., San Francisco, CA 94107
(415) 957-0886

**Lattice C ..... $149.95**
Tenchstar Inc./Metacomco
26 Portland Sq., Bristol BS2 8RZ, U.K.
(0272) 428781

**Mark Williams C ..... $179.95**
Mark Williams Company
1430 West Wrightwood, Chicago, IL 60614
(312) 472-6659

**Megamax C ..... $199.95**
Megamax, Inc.
Box 851521, Richardson, TX 75085
(214) 987-4931

**OS/9 C ..... $500.00**
(requires OS/9 operating system)
Microware Systems
1900 NW 114th St., Des Moines, IA 50322
(515) 224-1929

## FORTRAN

The first FORTRAN (FORmula TRANslation) compiler appeared in 1956, and is thus the grandaddy of all compilers. The original design was greatly influenced by the scientific and mathematical needs of its developer and users, and this bias has remained despite the addition of string processing and other services. FORTRAN is preeminently a "scientific" language. For example, a coding error in a FORTRAN DO loop reputedly enabled a NASA space probe to miss Venus sometime in the 1970s.

**AC/FORTRAN ..... $199.95**
Absoft
4268 N. Woodward, Royal Oak, MI 48072
(313) 549-7111

**FORTRAN 77 ..... $149.95**
(with UCSD Pascal)
TDI Software, Inc.
10410 Markison Rd., Dallas, TX 75238
(214) 340-4942

**FORTRAN 77 ..... $79.95**
(with required P-system $99.95)
Pecan Software Systems, Inc.
1410 39th St., Brooklyn, NY 11218
(718) 851-3100

**OS/9 FORTRAN 77 ..... $750.00**
Microware Systems
1900 NW 114th St., Des Moines, IA 50322
(515) 224-1929

**Pro FORTRAN-77 v1.15 ..... $199.95**
Prospero Software
190 Castelnau, London SW13 9DH, U.K.
01-741-8531

## LISP

LISP is an acronym for LISt Processing. It's a much older language than it looks, dating back to 1959, when it was first implemented at MIT. LISP programming technique is heavily dependent on recursion, and the programs are unusual also in that they're built up of (and are reducible to) LISP data structures. The language is easily extensible.

**Cambridge Lisp 68000 ..... $199.95**
Tenchstar Inc./Metacomco
26 Portland Sq., Bristol BS2 8RZ, U.K.
(0272) 428781

**LisPas II ST ..... $162.70**
TommySoftware
Mainzer Landstrasse 147,
6000 Frankfurt/M., 1/West Germany
069-736917

**XLisp ..... Public Domain**
Written by David Betz, Christopher F. Chabris
15 Sterling Rd., Armonk, NY 10504
(914) 273-8828

## PROLOG

PROLOG (from PROgramming in LOGic) is a relatively new nonprocedural, object-oriented language—which means that it looks quite different from what most programmers are used to. PROLOG is being used more and more in artificial intelligence applications, among them the publicized "Fifth Generation Computer" project in Japan.

**Prolog ..... $39.95**
Rational Visions
7111 W. Indian School Rd., Ste. 131,
Phoenix, AZ 85033
(602) 846-0371

**MPROLOG ..... $199.00**
Logicware
Park Avenue Atrium
237 Park Ave., Ste. 2136, New York, NY 10017
(212) 551-3536

## BASIC

BASIC is the most popular language on microcomputers, mostly due to its simplicity, ease of use and wide distribution. In its simplest form, it uses line numbers to organize program statements, and thus is highly lacking in structure. BASIC, mostly suited for novice programmers, is easy to learn, but in most cases is severely restrictive. There are exceptions, however, particularly GFA BASIC and Fast BASIC, which are more fully implemented languages, allowing easy access to GEM and other special features of the ST. In fact, some consider these new BASIC "hybrids" not to be BASIC at all.

**BASIC Compiler** ..... **$79.95**
Pecan Software Systems, Inc.
1410 39th St., Brooklyn, NY 11218
(718) 851-3100

**Fast BASIC** ..... **$149.95**
Computer Concepts, distributed by G-Plus
130 Albert St., Ottawa
Ontario K1P 5G4, Canada
(613) 230-7750

**Fast BASIC-M Compiler v1.35** ..... **$129.00**
Philon, Inc.
641 Avenue of the Americas
New York, NY 10011
(212) 807-0303

**GFA BASIC Interpreter** ..... **$79.95**
MichTron, Inc.
576 South Telegraph, Pontiac, MI 48053
(313) 334-5700

**GFA BASIC Compiler** ..... **$79.95**
MichTron, Inc.
576 South Telegraph, Pontiac, MI 48053
(313) 334-5700

**LDW BASIC Compiler** ..... **$69.95**
Logical Design Works, Inc.
780 Montague Expwy., Ste. 205
San Jose, CA 95131
(408) 435-1445

**OS/9 BASIC** ..... **$275.00**
(available soon; requires OS/9 OS)
Microware Systems
1900 NW 114th St., Des Moines, IA 50322
(515) 224-1929

**Real BASIC** ..... **$69.95**
CCL Computer
516 5th Ave., Ste. 507, New York, NY 10036
(212) 644-2591

**ST BASIC** ..... **Free w/ST purchase**
Atari Corp.
1196 Borregas Ave., Sunnyvale, CA 94086
(408) 745-2000

**Softworks BASIC** ..... **$79.95**
Softworks Limited
2944 N. Broadway, Chicago, IL 60657
(312) 975-4030

**True BASIC** ..... **$99.00**
True BASIC, Inc.
39 South Main St., Hanover, NH 03755
(800) TRBASIC

## PASCAL

Pascal is a highly structured, compiled language. Its use of English-like keywords makes source code readable and easy to follow. Pascal is a more powerful language than BASIC, but is more restrictive than languages such as assembly and C. Though Pascal is suitable for any level programmer, it's especially recommended for novices who wish to learn structured programming techniques without having to deal with the more cryptic syntax of C.

**Alice** ..... **$79.95**
Looking Glass Software
124 King St., N., Waterloo,
Ontario N2J 2X8, Canada
(519) 884-7473

**MCC Pascal** ..... **$99.95**
Metacomco
26 Portland Sq., Bristol BS2 8RZ, U.K.
(0272) 428781

**OS/9 Pascal** ..... **$500.00**
(available soon; requires OS/9 OS)
Microware Systems
1900 NW 114th St., Des Moines, IA 50322
(515) 224-1929

**Personal Pascal** ..... **$74.95**
Optimized Systems Software, Inc.
1221 B Kentwood Ave., San Jose, CA 95129
(408) 446-3099

**UCSD Pascal** ..... **$79.95**
TDI Software, Inc.
10410 Markison Rd., Dallas, TX 75238
(214) 340-4942

**UCSD Pascal** ..... **$199.95**
Pecan Software Systems, Inc.
1410 39th St., Brooklyn, NY 11218
(718) 851-3100

## MODULA-2

Modula-2 is a close cousin to Pascal, although it provides expert programmers the ability to "get closer to the machine." Most of the restrictions of Pascal have been removed, providing a programming environment more suitable for professional use.

**Modula-2** ..... **$99.95**
Pecan Software Systems, Inc.
1410 39th St., Brooklyn, NY 11218
(718) 851-3100

**Modula-2** ..... **$99.95**
TDI Software, Inc.
10410 Markison Rd., Dallas, TX 75238
(214) 340-4942

**Modula-2** ..... **$39.95**
Jefferson Software
12416 N. 28th Dr. #18-236,
Phoenix, AZ 85029-2434
(602) 243-3106

## FORTH

First introduced in 1971, FORTH has gradually attracted a devoted band of followers. It is primarily an interpreted (but often also compiled), stack-oriented, easily extensible language. Using this last feature, programmers quickly develop their own "version" of FORTH. The name is an abbreviation of *fourth*, for "fourth generation language." FORTH is quite different from all the other languages listed here, although it has some similarities to LISP.

**4XForth** ..... **$99.95**
The Dragon Group
148 Poca Fork Rd., Elkview, WV 25071
(304) 965-5517

**Mach 2** (Forth-83) ..... **$99.95**
Palo Alto Shipping
P.O. Box 7430, Menlo Park, CA 94026
(800) 443-6785

**Multi Forth** ..... **$89.00**
Creative Solutions, Inc.
4701 Randolph Rd., Ste. 12
Rockville, MD 20852
(800) 367-8465

**ST Forth/MT** ..... **$49.95**
Abacus Software
P.O. Box 7219, Grand Rapids, MI 49510
(616) 241-5510

## MISCELLANEOUS

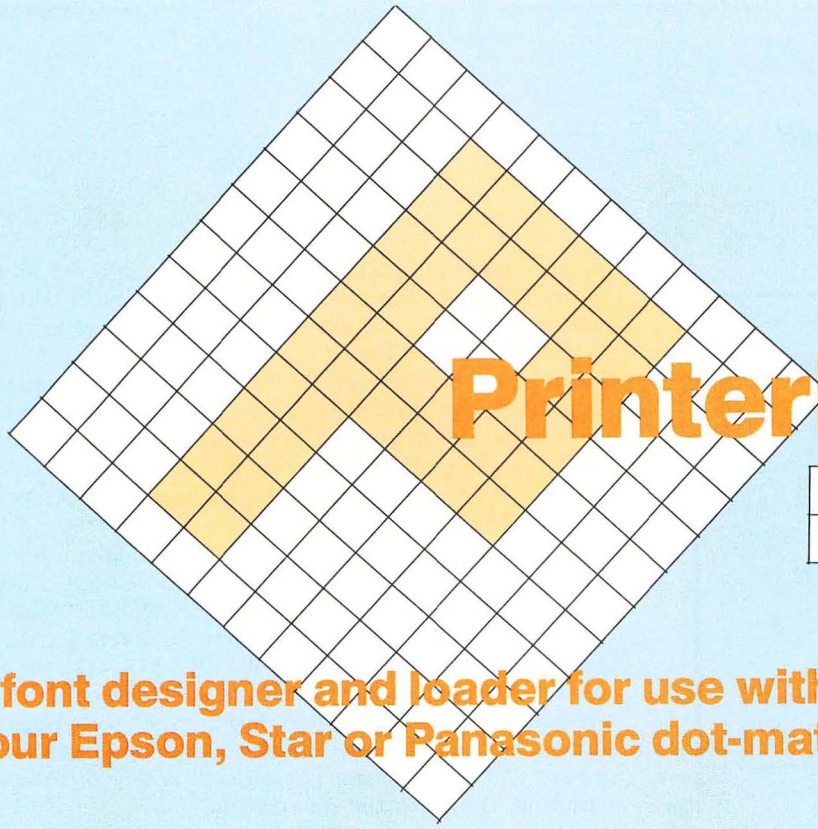
**APL.68000** ..... **$295.00**
Spencer Organization, Inc.
P.O. Box 248, Westwood, NJ 07675
(201) 666-6011

**BCPL** ..... **$149.95**
Tenchstar Inc./Metacomco
26 Portland Sq., Bristol BS2 8RZ, U.K.
(0272) 428781

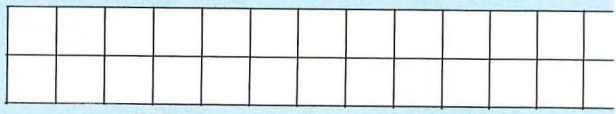
**BOS/MicroCOBOL** ..... **$550.00**
BOS National
2607 Walnut Hill La., Ste. 200, Dallas, TX 75229
(214) 956-7722

**Icon** ..... **Public Domain**
(Call for information)
Icon Project, Dept. of Computer Science
University of Arizona, Tucson, AZ 85721
(602) 621-6613

**ST Logo** ..... **Free with ST purchase**
Atari Corp.
1196 Borregas Ave., Sunnyvale, CA 94086
(408) 745-2000



# PrinterFont



## A font designer and loader for use with your Epson, Star or Panasonic dot-matrix printer.

by Richard J. Bourne

No single accessory increases the usefulness of your computer more than a good printer. Atari ST owners recognize this, and many have purchased high-end personal printers such as the Epson, Star or Panasonic brands. All these printers offer a powerful but little-used feature called "character downloading," which allows you to create custom characters and store them in the printer's RAM. Everything from script and Old English to digital fonts and graphic shapes can be sent to your printer, and used in place of the default characters.

However, the method described in most printer manuals for accessing character downloading seems calculated to discourage its use. You must draw each character on a grid, calculate each dot value in binary, total up each of nine to twelve columns as a byte value, and then use BASIC's LPRINT command followed by a long series of CHR$ values. This procedure is repeated for *every* character. And, when your printer is turned off, you lose all your work and must start over!

My solution is a font creation program, **PrinterFont**, which allows you to edit your characters on a grid, sized automatically to your printer's needs. Toggling dots on and off is as simple as putting the mouse pointer in the correct box and clicking the left mouse button. Menu choices — saving or loading a font set, testing it, or downloading it to the printer—are also performed with the mouse. The keyboard is used only to enter a filename or select a cell to edit.

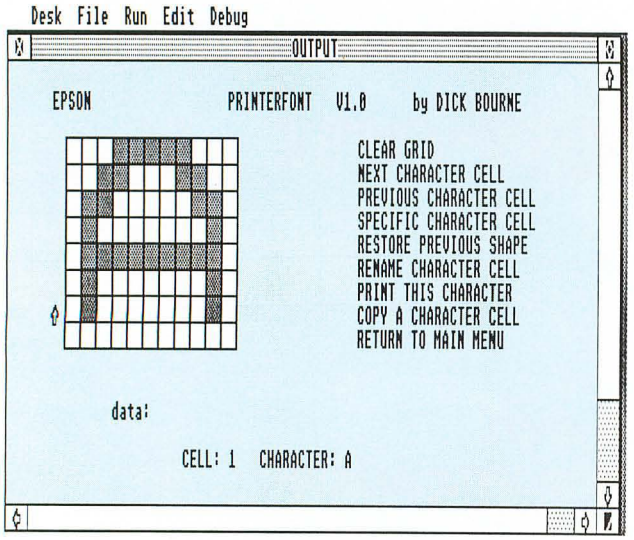
### How to use *PrinterFont*.

To type in the program, first load BASIC and disable buffered graphics—unless you have TOS in ROM or a 1040ST. Select the "edit" window, so that you can make corrections as you proceed. When you reach the debugging stage, you will appreciate the modular approach of **PrinterFont**. Subroutines are all named in accordance with their function. Most problems will be easily traceable to such a subroutine. Also, if certain functions are not important to you, you can easily scale down the size of the program by typing in just the subroutine name, followed by a colon and a RETURN. You can easily add such functions at a later date.

When you run the program, the first thing you're asked is the printer type. Once you select from the choices offered, you cannot change your decision unless you stop and rerun the program. The reason is that each printer has slight differences in the grid shape required for editing, as well as in the total number of characters which can be downloaded.

After you choose your printer, a grid will be drawn and nine main menu choices will appear on the right side of the screen. Position your mouse pointer on CREATE CHARACTER SET and click the left button. Your choice will be highlighted in red. After a brief pause for clearing font memory, the program will ask you for a character name (the usual choice would be *A* or *a*). Once you press a character key and RETURN, an edit submenu will replace the main menu, and a lower-screen message will identify the cell number and character name. You can now begin to draw in Cell 1. Move your pointer to any space on the grid, and click your left button to toggle a dot on or off. ON dots will appear in color (default = green), while OFF dots will be the screen color.

There are few rules to follow when designing a character. You will normally make letters and numbers nine dots wide, to match your printer's standard characters. Letters without descenders and numbers sit on a "baseline" at the bottom of the grid (Star) or one row up (Epson, Panason-




ic). Capitals are normally seven dots high, while lowercase letters are five dots high. The Panasonic printers, with a 9x8 grid, allow you only a one-dot descender on your g, j, p, q and y. The Star printers, although they use only a 9x7 grid, allow you to shift characters down two spaces. Therefore, design descender characters *two rows* above the baseline, then click on the up-arrow symbol to the left of the grid. When it is pointing down, the "descender mode" will be activated when that character is printed. Epson printers use an 11x8 grid and allow for a one-dot downshift. Therefore, design Epson descender characters *one row* above the baseline, then click on the up-arrow symbol to invert it and activate the downshift.

Do *not* place any two dots side-by-side. If you do, the dot to the right will not be printed, as the pins can't fire that quickly. A little experimentation will help you get the look you want. At any time, click on the submenu choice PRINT THIS CHARACTER, and you'll be able to see your design on paper next to the default print character.

Once you've completed a character, click your mouse arrow on NEXT CHARACTER CELL. You'll see a "saved" message at lower screen (the character is saved to RAM, not disk), and you'll be asked for another character name. The usual choice would be *b* or *B* and RETURN. (You are not required to maintain alphabetic or numeric sequence as you build a font, although I find it helpful.) Again, you'll see a lower-screen identifier for cell number and character name. The previous character shape remains on the grid . . . handy for matching the style of one character to another. You can change *E* to *F* with just four clicks! You may select CLEAR GRID to start the character from scratch, if you wish.

You can move to a previous character cell to redefine it, or to any specific character cell (by number or by character name). If you choose to move by character name, you'll get an error message if the name isn't found. If you choose to go by number, it must be in the range *1* through *40* for Panasonic, and *1* through *96* for Epson or Star printers. You can actually download 256 characters to an Epson, but I didn't want to run out of ST memory and only allowed 96 to be stored in RAM. When you move back into a cell which has been designed, you'll get a nice bonus. . .the 9 to 11 bytes which define that shape to your printer will be displayed on the lower screen. You may wish to jot them down to use in a BASIC program of your own. You'll still need to refer to your printer manual, to see exactly how to use them in an LPRINT command. Moving to a new cell or returning to the main menu automatically saves the cell you're leaving (in RAM).

If after working on a character you decide you liked the old version better, click on RESTORE PREVIOUS SHAPE before you move to a new cell. RENAME CHARACTER CELL prompts you for a new character name for the current cell, while COPY CHARACTER CELL asks for a cell name or number whose pattern is to be copied to the current cell. You do not have to design every cell. When you download your font, the printer will keep default shapes for any characters you haven't defined. When you're finished creating, return to the main menu.

At this point, it's wise to SEND CHARACTERS TO DISK. You'll be prompted for a filename, which must: be no longer than eight characters, start with a letter, and contain only alphanumerics. I usually use names like COMP1, COMP2, etc., to indicate revisions of my fonts. Once you press RETURN, the extension .PFT will be appended and the font saved under that name.

Now the real power of **PrinterFont** is about to show itself! Click on DOWNLOAD SET TO PRINTER (a confirming message will appear in the lower screen). Now select PRINT CHARACTER SET. Your printer will spring to life and you'll get a complete chart, containing cell numbers, character names in default italics and your character designs. Undefined cells will be indicated by number only. If you've designed an alphanumeric set, set your printer at the top of the form and click on PRINT TEST TEXT to see an entire page of text printed with your font in various printer modes: Pica, Elite, bold, double-wide and condensed, in various combinations. You may cancel your downloaded shapes at any time by selecting RESTORE PRINTER, or by turning the printer off and on.

You may wish to go back and fine-tune your designs. Select EDIT CHARACTER SET, and, once you select the cell to edit in (by name or number), you'll see the edit submenu reappear—and you can set to work. The only difference between CREATE CHARACTER SET and EDIT CHARACTER SET is that the first clears font RAM memory and starts you in cell 1, while the latter keeps font RAM memory and lets you choose the cell to start working on. ST BASIC does not clear font RAM, even if the program is stopped and run again (nice to know, especially if an untrapped error stops it for you). Return to the main menu when you're finished and resave your font to disk. If you use the same filename as before, you'll be advised of the existing file and asked if you wish to replace it.

If you have another font on disk, you may then LOAD CHARACTERS FROM DISK. As in saving, you do *not* have to add the .PFT extension to the filename. You don't even have to choose a font created specifically for your printer.

When **PrinterFont** saves fonts, it includes a byte identifier of the printer type you're currently using. When it loads, it checks the font type and, if necessary, *converts it* to conform with the printer you're now using! This is a powerful feature, as fonts can be shared among ST users with different printers (although minor modifications will need to be made to some characters). Panasonic users, however, will only be able to load the first 40 characters of a 96-character Epson or Star file.

### A font farewell.

You'll be amazed at how quickly your collection of font files will grow. Remember that it takes a tenth as long to redesign as to build characters from scratch. You can get ideas for font styles from newspapers and magazines, or from lettering catalogs available from drafting supply stores. I'm limited to uppercase fonts only, as my Panasonic hasn't enough memory for both upper- and lowercase. But even a new set of "caps" gives a fresh look to your printouts. I'm constantly finding new applications. Using my FAT.PFT font when listing BASIC programs makes the capitals and numbers very thick and dark. My subroutine labels and line destinations then stand out from the other printing, and I can follow the program logic much better!

Advanced BASIC users will find some useful subroutines, such as the VDISYS call for mouse control, and the menu and grid selection subroutines. The speed of the program is impressive, partly due to the 8 MHz speed of the processor, and partly due to the grouping of time-critical subroutines near the start of the program. ///

---

*Richard J. Bourne, B.Ed., is a television instructor at the Southern Alberta Institute of Technology, where six DEC VAX 11s are available to the staff. He also utilizes computers for videotape editing and for titling. Richard's love affair with computers has led him from a TI-994A to an Atari 800, a Commodore 64 and an Atari 520ST.*

---

Listing 1.
ST BASIC listing.

```
10 gosub INIT:goto MAINMENU
20 ' ******************   PRINTERFONTER
   ***************
30 ' ****************** BY RICHARD BOUR
NE ***************
40 ' ****************** CALGARY,  ALBER
TA ***************
50 ' design and send custom characters
 to your Star, Epson
60 ' or Roland/Panasonic printer. Also
 allows sets to be
70 ' saved to disk and reloaded from d
isk.
80 GRID: for M=60 to 204-(brand=3)*32
step 16
90 linef M,28,M,108+(brand=2)*10
100 linef M-1,28,M-1,108+(brand=2)*10:
next
110 for n=28 to 108+(brand=2)*10 step
10
120 linef 60,n,204-(brand=3)*32,n:next
:return
130 PEEKMOUSE:poke contrl,124
140 poke contrl+2,0:poke contrl+6,0
150 bflag=0:mflag=0:vdisys(0)
160 x=peek(ptsout):y=peek(ptsout+2)
170 key=peek(intout):if key=down then
130
180 if key<>1 then 220
190 if x>320 and x<640 and y>48 and Y<
129 then mflag=1:gosub MENUSLCT:goto 2
20
200 if main=1 then 220
210 if x>44 and x<252 and y>30 and Y<1
40+(brand=2)*10 then gosub GRIDSLCT
220 down=key:if mflag=1 then return el
se 130
230 DRAWCHAR: gotoxy 12,14
240 print"data:
                    "
250 gotoxy 18,14:if pm(cn,id)=0 then r
eturn
260 for m=1 to mtop:if brand=1 then pr
int pm(cn,m);:goto 280
270 t=cn:gosub CONVERT:print tv2%;
280 for n=1 to ntop
290 bd(m,n)=0:colr=0
300 if pm(cn,m)and 2^(8-n) then bd(m,n
)=1:colr=3
310 color 1,colr:fill x1+m*16+1,y1+n*1
0+1
320 next:next:print
330 if brand=2 then if pm(cn,1) and 1
then flip=1 else flip=0
340 if brand=3 then if pm(cn,12) and 1
28 then flip=1 else flip=0
350 if brand<>1 then gosub FLIPDES
360 return
370 GRIDCLR:for m=1 to mtop:for n=1 to
 ntop
380 bd(m,n)=0:color 1,0
390 if brand=2 and m>1 and n=ntop then
 410
400 fill x1+m*16+1,y1+n*10+1
410 next:next:return
420 GRIDTOPM:for m=1 to mtop-(brand=3)
430 pm(cn,m)=0:for n=1 to ntop
440 pm(cn,m)=pm(cn,m)+2^(8-n)*bd(m,n)
450 next:next
460 if brand=2 then pm(cn,1)=pm(cn,1)+
bd(1,8)
470 return
480 GRIDTOTC:for m=1 to mtop-(brand=3)
490 tdat(m)=0:for n=1 to ntop
500 tdat(m)=tdat(m)+2^(8-n)*bd(m,n)
510 next:next:return
520 CONVERT: tv%=pm(t,m)/2
530 tv2%=0:for n=1 to 7:tv2%=tv2%+(tv%
 and 2^(7-n))*(2^((n-4)*2))
540 next:return
550 SAVDATA:on error goto 2680
560 gosub CLRLINE
570 input "name of file: limit 8 chara
cters";name$
580 if len(name$)=0 then 640
590 if len(name$)>8 then 560
600 name$=name$ + ".pft"
610 open "O",#1,name$,128:print #1,bra
nd
620 for t=1 to high:for m=1 to id
630 print#1, pm(t,m):next:next:close #
1
640 gosub CLRLINE:return
650 LOADATA:on error goto 2680
660 gosub CLRLINE:input "name of file"
;name$
670 t=1:if len(name$)=0 then 750
680 if right$(name$,4)<>".pft" then na
me$=name$ + ".pft"
690 open "I",#1,name$,128:input #1,br
700 gosub CLRLINE:print "    "br$(br)"
 FILE"
```

```
710 while not (EOF(1) or t>high)
720 for m=1 to 9-3*(br=3):input #1,pm(
t,m):next
730 input #1,pm(t,id):t=t+1
740 wend:close #1
750 gosub CLRLINE:return
760 INIT:x1=44:y1=20:ntop=8:menlm=9:lp
=1:hp=11
770 dim bd(12,8),pm(96,13):fullw 2:cle
arw 2:e$=chr$(27)
780 z$=chr$(0):o$=chr$(1):r$=chr$(13):
color 1,1
790 br$(1)="ROLAND/PANASONIC":br$(2)="
STAR GEMINI":br$(3)="EPSON"
800 width lprint 255:return
810 MENUPRINT:for t=1 to menlm
820 read menu$(t):gotoxy 40,2+t
830 print menu$(t):next:return
840 data "CREATE CHARACTER SET
"
850 data "EDIT CHARACTER SET
"
860 data "SEND CHARACTERS TO DISK
"
870 data "LOAD CHARACTERS FROM DISK
"
880 data "DOWNLOAD SET TO PRINTER
"
890 data "RESTORE PRINTER
"
900 data "PRINT CHARACTER SET
"
910 data "PRINT TEST TEXT
"
920 data "END PROGRAM
"
930 MENUSLCT:if temp=0 then 950
940 gotoxy 40,2+temp:color 1,1:print m
enu$(temp)
950 t=int(((y-48)/9)+1):if t>menlm the
n return
960 gotoxy 40,2+t:color 2,1
970 print menu$(t):color 1,1:temp=t:re
turn
980 GRIDSLCT:m=int((x-44)/16):if m>mto
p then return
990 n=int((y-20)/10)-2:if n<1 or n>nto
p then return
1000 if brand<>1 and m<1 then flip=1-f
lip:gosub FLIPDES:return
1010 bd(m,n)=1-bd(m,n):gosub FILLBOX:r
eturn
1020 CLRMAT:for t=1 to high:for m=1 to
 mtop+1-(brand=3)
1030 pm(t,m)=0:next:next:return
1040 FILLBOX:colr=0:if bd(m,n)=1 then
colr=3
1050 color 1,colr:fill x1+m*16+1,y1+n*
10+1:return
1060 TESTPRTR: cn=0:temp=0:lprint
1070 lprint "NAME: ";name$:lprint
1080 for n=1 to high/4:for m=1 to 4
1090 cn=cn+1:if cn<10 then lprint " ";
1100 lprint cn"; ";
1110 if pm(cn,10)=0 then lprint "
";:goto 1140
1120 lprint e$"4"chr$(pm(cn,10))" ";
1130 lprint e$"5"chr$(pm(cn,10))"  ";
1140 next:lprint:next:return
1150 CLRLINE:gotoxy 20,16
1160 print"                    "
1170 gotoxy 20,16:return
1180 SHORT:q$=left$(q$,1):if q$="Y" th
en q$="y"
1190 return
1200 SENDDATA: gosub CLRLINE
1210 print "SENDING CHARACTER DATA"
1220 if brand=2 then lprint e$"*"z$;'c
opy ROM set to RAM
1230 if brand=3 then lprint e$":"z$z$z
$;'ditto
1240 for t=1 to high:if pm(t,id)=0 the
n 1350
1250 if brand=1 then lprint e$"y"chr$(
pm(t,id));:goto 1310
1260 if brand=3 then lprint e$"&"z$chr
$(pm(t,id))chr$(pm(t,id)); else 1280
1270 lprint chr$(pm(t,id-1));:goto 131
0'proport. & descender code
1280 lprint e$"*"chr$(1)chr$(pm(t,id))
;
1290 desc=0:if pm(t,1)/2-(int(pm(t,1)/
2)) then desc=1
1300 lprint chr$(desc);'descender code
1310 for m=1 to mtop
1320 if brand<>2 then lprint chr$(pm(t
,m));:goto 1340
1330 gosub CONVERT:lprint chr$(tv2%);
1340 next:lprint r$;
1350 next:gosub CLRLINE: print "DATA S
END COMPLETE"
1360 if brand=2 then lprint e$"$"o$;'a
ctivate RAM set
1370 if brand=3 then lprint e$"%"o$z$;
'ditto
1380 return
1390 CLRPRTR: gosub CLRLINE
1400 print "RESTORING NORMAL CHARACTER
S"
1410 if brand=2 then lprint e$"$"z$;:g
oto 1440'activate ROM set
1420 if brand=3 then lprint e$"%"z$z$;
:goto 1440'ditto
1430 for t=1 to high:lprint e$"z"chr$(
pm(t,id));r$;:next
1440 gosub CLRLINE: print "PRINTER RES
TORED":return
1450 TRUNCATE:c$=chr$(pm(cn,id)):retur
n
1460 MAINMENU:boxlocate=0:temp=0
1470 gotoxy 25,1: print "PRINTERFONT
 V1.0";
1480 print tab(25)"by DICK BOURNE":gos
ub PBRAND:gosub GRID
1490 mtop=9-(brand=3)*2:id=mtop+1-(bra
nd=3):ntop=8+(brand=2)
1500 gotoxy 5,1:print br$(brand)
1510 if brand>1 then gotoxy 5,10:print
 "╞"
1520 main=1
1530 restore 840:gosub MENUPRINT
1540 gosub PEEKMOUSE:main=0
1550 on TEMP gosub CREATE, EDCHAR, SAV
DATA, LOADATA
1560 on TEMP-4 gosub SENDDATA, CLRPRTR
, TESTPRTR,PRINTEXT,ALLEND
1570 goto 1520
1580 EDCHAR:ret=0:restore 1670:flip=0
1590 gosub MENUPRINT:gotoxy 40,menlm+4
1600 if cflag=1 then cflag=0:cn=1:gosu
b RENAME:gosub IDCHAR:goto 1620
1610 gosub SPECHARB
1620 gosub PEEKMOUSE
1630 if mflag<>1 then 1660
1640 on temp gosub GRIDCLR,NXTCHAR,PRV
CHAR,SPECHAR
1650 on temp-4 gosub DRAWCHAR,RENAME,P
RINT1,COPYCHAR,RETURMN
1660 if ret=1 then main=1:return else
1620
1670 data "CLEAR GRID
 "
1680 data "NEXT CHARACTER CELL
 "
1690 data "PREVIOUS CHARACTER CELL
 "
```

```
1700 data "SPECIFIC CHARACTER CELL
 "
1710 data "RESTORE PREVIOUS SHAPE
 "
1720 data "RENAME CHARACTER CELL
 "
1730 data "PRINT THIS CHARACTER
 "
1740 data "COPY A CHARACTER CELL
 "
1750 data "RETURN TO MAIN MENU
 "
1760 NXTCHAR:gosub SAVECHAR
1770 cn=cn+1:if cn>high then cn=1
1780 gosub DRAWCHAR:gosub IDCHAR:retur
n
1790 PRVCHAR:gosub SAVECHAR
1800 cn=cn-1:if cn<1 then cn=high
1810 gosub DRAWCHAR:gosub IDCHAR:retur
n
1820 SPECHAR:gosub SAVECHAR
1830 SPECHARB:
1840 gosub CLRLINE: INPUT "1.BY NUMBER
  2.BY NAME";wh$
1850 on val(wh$) goto 1870,1890
1860 goto 1840
1870 gosub CLRLINE:input "CELL# to EDI
T";cell
1880 if cell<1 or cell>high then 1870
else 1930
1890 gosub CLRLINE:input"CHARACTER NAM
E";char$
1900 found=0:for t=1 to high
1910 if char$=chr$(pm(t,id)) then foun
d=1:cell=t:t=high
1920 next:if found=0 then gosub CLRLIN
E:print"   NOT FOUND":goto 1840
1930 cn=cell:gosub DRAWCHAR:gosub IDCH
AR:return
1940 SAVECHAR: gosub GRIDTOPM
1950 gosub CLRLINE:print "     CHARACT
ER SAVED":return
1960 PRINT1:gosub TRUNCATE:on brand go
to 1970,1990,2010
1970 lprint e$"z"c$"character "c$": ";
1980 lprint e$"y"c$;:goto 2030
1990 lprint e$"$"z$"character "c$": ";
2000 lprint e$"*"o$c$chr$(bd(1,8));:go
to 2030
2010 lprint e$"%"z$z$"character "c$":
";
2020 lprint e$"&"z$c$c$chr$(bd(12,1)*1
28);
2030 gosub GRIDTOTC
2040 for m=1 to mtop
2050 if brand<>2 then lprint chr$(tdat
(m));:goto 2070
2060 t=cn:gosub CONVERT:lprint chr$(tv
2%);
2070 next:lprint c$
2080 if brand=1 then lprint e$"z"c$;:r
eturn
2090 if brand=2 then lprint e$"$"z$;:r
eturn
2100 lprint e$"%"z$z$;:return
2110 COPYCHAR:tempcell=cn
2120 gosub CLRLINE: INPUT "1.BY NUMBER
  2.BY NAME  3.CANCEL";wh$
2130 on val(wh$) goto 2150,2170,2760
2140 goto 2120
2150 gosub CLRLINE:input "CELL# to COP
Y FROM";cell
2160 if cell<1 or cell>high then 2150
else 2210
2170 gosub CLRLINE:input"CHARACTER NAM
E";char$
2180 found=0:for t=1 to high
2190 if chr$(pm(t,id))=char$ then foun
d=1:cell=t:t=high
2200 next:if found=0 then gosub CLRLIN
E:print"   NOT FOUND":return
2210 cn=cell:gosub DRAWCHAR
2220 cn=tempcell:gosub IDCHAR:return
2230 RETURMN: gosub SAVECHAR:ret=1:tem
p=0:return
2240 IDCHAR:if pm(cn,id)=0 then gosub
RENAME
2250 gosub CLRLINE: print"CELL:"cn;"
CHARACTER: ";
2260 print chr$(pm(cn,id)):return
2270 CREATE:gosub CLRLINE:lp=1:hp=11
2280 print "CLEARING MEMORY":name$="":
gosub CLRMAT
2290 gosub GRIDCLR:gosub CLRLINE
2300 cflag=1:gosub EDCHAR:return
2310 RENAME:gosub CLRLINE:input"CHARAC
TER NAME";wh$
2320 if len(wh$)<1 then pm(cn,id)=0: r
eturn
2330 if len(wh$)>1 then 2310
2340 pm(cn,id)=asc(wh$):return
2350 PRINTEXT:lprint:lprint "NAME: ";n
ame$:lprint:goto STYLES
2360 TEXT:lprint:lprint"THE QUICK BROW
N FOX JUMPED OVER THE LAZY DOGS"
2370 lprint"The Quick Brown Fox Jumped
 Over The Lazy Dogs"
2380 lprint"1234567890 !@#$^&*()_+-=(♦
♠↖[];"'<,>.?/|\"
2390 lprint:return

2410 BLDOFF:lprint e$"F"o$;:return
2420 DBL:lprint e$"G";:return
2430 DBLOFF:lprint e$"H";:return
2440 WD:lprint e$"W"o$;:return
2450 WDOFF:lprint e$"W"z$;:return
2460 ELITE:lprint e$"M";:return
2470 PICA:lprint e$"P";:return
2480 SELITE:lprint e$"B"chr$(2);:retur
n
2490 SPICA:lprint e$"B"o$;:return
2500 CON:lprint chr$(15);:return
2510 CONOFF:lprint chr$(18);:return
2520 NLQ:lprint e$chr$(110):return
2530 STYLES:lprint "PICA":gosub TEXT:g
osub BLD:lprint"PICA BOLD"
2540 gosub TEXT:gosub BLDOFF
2550 gosub WD:lprint "WIDE TEXT":gosub
 TEXT
2560 gosub BLD:lprint "WIDE BOLD":gosu
b TEXT:gosub WDOFF:gosub BLDOFF
2570 gosub DBL:lprint "DOUBLE":gosub T
EXT:gosub BLD
2580 lprint "DOUBLE BOLD":gosub TEXT
2590 gosub DBLOFF:gosub BLDOFF
2600 if brand=1 then gosub ELITE else
gosub SELITE
2610 lprint "ELITE":gosub TEXT
2620 gosub WD:lprint "ELITE WIDE":gosu
b TEXT
2630 gosub WDOFF:if brand=1 then gosub
 PICA else gosub SPICA
2640 gosub CON:lprint "CONDENSED":gosu
b TEXT
2650 if brand=1 then gosub CONOFF else
 gosub SPICA:goto 2670
2660 gosub NLQ:lprint "NEAR LETTER QUA
LITY":gosub TEXT:gosub PICA
2670 lprint:return
2680 ERRCHECK:gosub CLRLINE
2690 if err=61 then print "DISK FULL O
R WRITE PROTECT ON!"
2700 if err=61 then close 1:RESUME 840
2710 if err=53 then print "       FILE
 NOT FOUND"
2720 if err=53 then close 1:RESUME 940
2730 if err=52 then print "     INVALI
D FILENAME"
```

```
2740 if err=52 then close 1:RESUME 840
2750 print " LOAD ERROR - TRY AGAIN":t
=40:m=10:resume 730
2760 PBRAND:gosub CLRLINE
2770 input "1.ROLAND/PANASONIC  2.STAR
  3:EPSON";brand$
2780 brand=val(brand$):if brand<1 or b
rand>3 then 2760
2790 high=96:if brand=1 then high=40
2800 gosub CLRLINE:return
2810 FLIPDES:if brand=2 then bd(1,8)=f
lip
2820 if brand=3 then bd(12,1)=flip*(12
8)+10
2830 gotoxy 5,10:print chr$(1+flip):re
turn
2840 ALLEND:end
```

●

ST CHECKSUM DATA.

*(see page 8)*

```
 10 data  765, 82, 422, 286, 852, 663,
 165, 264, 711, 777, 4987
 110 data  538, 536, 875, 496, 703, 85
6, 425, 512, 713, 513, 6167
 210 data  734, 719, 323, 274, 206, 78
8, 596, 392, 415, 730, 5177
 310 data  847, 655, 273, 606, 487, 35
2, 584, 686, 187, 185, 4862
 410 data  900, 959, 642, 332, 783, 11
2, 357, 979, 806, 218, 6088
 510 data  902, 306, 366, 112, 970, 32
8, 973, 155, 174, 7, 4293
 610 data  733, 131, 956, 331, 923, 57
4, 790, 999, 343, 346, 6126
 710 data  1, 688, 336, 189, 336, 928,
 366, 376, 981, 581, 4782
 810 data  91, 852, 201, 400, 374, 650
, 860, 680, 360, 464, 4932
 910 data  255, 767, 944, 341, 678, 33
7, 669, 609, 311, 182, 5093
 1010 data  415, 805, 56, 475, 854, 39
7, 165, 322, 534, 246, 4269
 1110 data  415, 964, 54, 936, 90, 474
, 82, 417, 449, 655, 4536
 1210 data  416, 5, 685, 291, 117, 915
, 91, 419, 301, 887, 4127
 1310 data  556, 435, 728, 539, 801, 6
12, 418, 454, 536, 248, 5327
 1410 data  803, 541, 159, 817, 963, 2
80, 660, 628, 377, 19, 5247
 1510 data  38, 272, 292, 508, 403, 72
8, 572, 686, 109, 817, 4425
 1610 data  362, 642, 930, 838, 101, 1
53, 681, 399, 733, 636, 5475
 1710 data  838, 594, 499, 529, 465, 6
43, 28, 225, 642, 980, 5443
 1810 data  221, 586, 753, 736, 474, 5
87, 743, 683, 18, 502, 5303
 1910 data  905, 767, 505, 19, 865, 98
0, 97, 574, 950, 6, 5668
 2010 data  151, 547, 423, 552, 486, 4
90, 343, 570, 360, 653, 4575
 2110 data  303, 917, 926, 558, 561, 6
33, 997, 488, 891, 613, 6887
 2210 data  530, 830, 430, 706, 190, 1
37, 698, 577, 320, 309, 4727
 2310 data  795, 419, 226, 660, 362, 4
68, 432, 726, 240, 875, 5203
 2410 data  308, 655, 88, 647, 106, 85
2, 519, 0, 46, 108, 3329
 2510 data  547, 381, 916, 288, 772, 3
4, 227, 326, 582, 121, 4194
 2610 data  591, 980, 689, 976, 716, 3
0, 247, 867, 284, 467, 5847
 2710 data  130, 474, 474, 472, 24, 69
7, 379, 554, 532, 53, 3789
 2810 data  196, 949, 906, 76, 2127
```

●

# Animation with Pascal

## Real-time image animation using *Personal Pascal*.

by David Plotkin

**Personal Pascal** from OSS is a very powerful language, both due to the built-in GEM libraries and because it allows access to all GEM routines, even those not directly supported. This article shows you how to use **Personal Pascal** to animate multicolored, complex figures—and how to move them around the screen.

### Principles of animation.

Animation in computer graphics is surprisingly similar to that seen in Saturday-morning cartoons. A series of images is put on the screen. These vary slightly from one another, and if the images are flashed on the screen fast enough, your eye and brain will be fooled into seeing smooth, continuous motion. This is the way movies work, too. Each image on the film is projected onto the screen, and your eye "sees" continuous motion.

You have an option—to move your animated figure around the screen. This gives another range of motion to your creation.

### Animation figures.

The first question we need to address is: where do you get the data which makes up the series of pictures for on-screen display? You can generate pictures from graphics programs, although this introduces a complication—you'll need to know the structure of the file produced by the graphics program. This method also makes it difficult to produce a program for publication in a magazine.

The other method is the one used in the two demo programs included with this article to illustrate the concepts of animation. Most Microsoft languages have commands such as "Screen GET" and "Screen PUT." Our method works in a similar way.

First, draw your picture on the screen, using the various GEM drawing commands. Then pick the data up off the screen (actually, out of screen memory) and store it in an array. Later, you can put the data back on-screen from the array. Your ST can get the data on-screen so fast that realistic animation is easily achieved.

### The programs.

The heart of the animation process lies in Listing 1, GEMRAST.PAS. This was written by Mark Rose of OSS, publishers of **Personal Pascal**. What this program does is use **Personal Pascal**'s generic GEM calls to access the VDI function known as raster_op.

While a complete discussion of VDI's raster_op is beyond the scope of this article, it is this function which picks up and deposits data to and from the screen. Notice that the GEMRAST listing is compiled separately from the main program. Use it *exactly* as listed and compile it with the "Chain to Linker" option turned off under "Compile Options." This will produce a file called GEMRAST.O. You must compile GEMRAST separately and place it in "Additional Link Files" under "Link Options" for your main program, rather than including the elements of the listing in the main program.

This is because, due to the requirements of VDI, PROCEDURE Copy_Opaque needs integer arguments when compiled, but needs the address of the "Memory Form Definition Block" when called. This requirement for different types of arguments will cause an error if you include the elements of GEMRAST in your main program, but won't give you any problems if GEMRAST is compiled separately.

Listing 2, BLITSCRN.PAS, must also be compiled separately and not linked. Here, the problem is PROCEDURE Blit_Screen, which needs a long integer for scrn_ary when compiled, but an array address (where the data is stored) when called.

GEMRAST and BLITSCRN illustrate how to access VDI functions with **Personal Pascal**. The Port_Inquire call de-

termines the screen resolution. While there are other ways to get the screen resolution, Port_Inquire is guaranteed to work, even if Atari later increases the available resolution of the ST.

The first of the two demo programs (Listing 3) is called PAINTBRS.PAS. One of the uses for these routines is to let you pick up any section of the screen and use it as a brush in a graphics program or in setting up a screen.

PAINTBRS draws a rectangle on-screen, puts it in the buffer, then erases it. Move the mouse pointer around the screen, and press the left button anywhere you want to put down a copy of the design. The program is fast enough to let you paint a continous design if you want to.

The program itself is quite simple. The necessary PROCEDURE Blit_Screen is declared as external, then the design is painted on the screen with GEM commands and erased. Note that the mouse is hidden (hide_mouse) before any drawing is done. It is an excellent idea to always hide the mouse before drawing on-screen, since GEM will foul up the screen if you don't.

The last part of the program is a simple loop for mouse button clicks and messages from GEM. If the mouse button is clicked, you put down a copy of the pattern wherever the mouse pointer is currently located. If you click in the window close box (upper left corner), then GEM sends the WM_Closed message, causing the program to fall out of the WHILE loop and end. Remember, you must pull down "Link Options" and place both GEMRAST and BLITSCRN in additional link files before attempting to compile PAINTBRS. This is also true of the next demo.

Listing 4, the second demo, is called ANIMATE.PAS. It puts four figures on-screen and moves them around to follow the mouse pointer. The figures are animated and will be familiar to any fan of **Pac-Man**. The speed of their animation can be adjusted by pressing the up (slower) and down (faster) arrow keys on the keyboard.

ANIMATE is similar in many ways to PAINTBRS. The procedures Draw_Ghost, Ghost and Erase put the four views of the animated figures on the screen and erase them. The main program makes the various GEM calls and calls the procedures necessary to actually place the figures on-screen and store them in the four arrays. Then it goes into a loop, looking for the WM_Closed message, a timer event or a key press. Notice that, even though we aren't looking for a mouse button event, the location of the mouse pointer is returned by the GET_EVENT call. The variable flag keeps track of which view to put on-screen to effect the illusion of animation.

There are a few things you'll need to know in order to use the routines included in this article effectively in your own programs. The first is that you must provide enough white space around your figure to erase the previous figure, when you're moving a character across the screen. If the figure can take sizeable jumps (as it does when it follows the mouse pointer), it's advisable to have a "blank" array, to erase whatever is left behind before drawing the new figure.

Also, you must ensure that your array is large enough to hold *all* the data you're trying to pick up. If not, you could crash your machine. To calculate how large you must make your array, use the following:

(1) Take the width of your figure in pixels, divide by *16*, then round up to the nearest integer number of Words. Multiply by *2* to convert to bytes.

(2) Multiply the number obtained in (1) by the height of the figure in pixels. This is the minimum size to make your array so as to hold the data.

You have your instructions—now let's see some animation!

*David Plotkin has his Masters in Chemical Engineering and works as a Design Engineer for Chevron U.S.A. He owns a 130XE and a 520ST, and is currently a heavy Pascal user on the ST. His interests (on computers) lie in programming, games and tutorials.*

Listing 1.
Pascal listing.

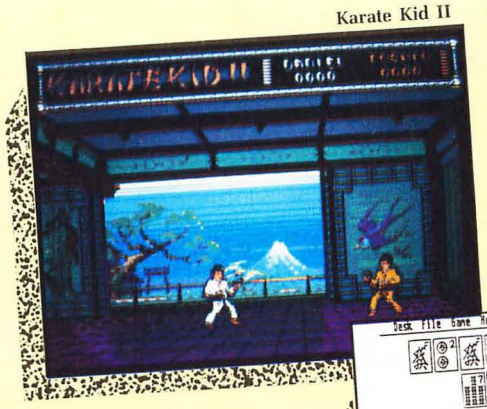
```
{$M+,E+}
PROGRAM Gem_Raster ;

{ gemrast.pas - Raster operation support for Personal Pascal library.

    The following routines are declared in this module:

    Copy_Opaque - Opaque raster copy operations.  The source and destination
                  bitmaps must be of the same color/monochrome format, but
                  need not be the same device format (?).
}

  CONST
    {$I gemconst.pas}

  TYPE
    {$I gemtype.pas}

    Form_Format = ( Device_Format, Standard ) ;
    Mem_Form_Def = RECORD
                     mem : long_integer ;
                     pw : integer ;
                     ph : integer ;
                     width : integer ;
                     format : Form_Format ;
                     planes : integer ;
                     res1 : integer ;
                     res2 : integer ;
                     res3 : integer ;
                   END ;
    Raster_Op = ( ROP_0, ROP_And, ROP_AndNot, ROP_Replace, ROP_NotAnd, ROP_NOP,
                  ROP_Xor, ROP_Or, ROP_Not_Or, ROP_Not_Xor, ROP_NotD,
                  ROP_OrNot, ROP_Not5, ROP_NotOr, ROP_Not_And, ROP_1 ) ;

    Ctrl_Parms = ARRAY[0..11] OF Integer;
    Int_In_Parms = ARRAY[0..15] OF Integer;
    Int_Out_Parms = ARRAY[0..45] OF Integer;
    Pts_In_Parms = ARRAY[0..11] OF Integer;
    Pts_Out_Parms = ARRAY[0..11] OF Integer;
    Inq_type = (Open_Vals, Ext_Vals);

    PROCEDURE VDI_Call(cmd,sub_cmd,nints,npts : Integer; VAR ctrl : Ctrl_Parms;
                       VAR int_in : Int_In_Parms ; VAR int_out : Int_Out_Parms;
                       VAR pts_in : Pts_In_Parms ; VAR pts_out : Pts_Out_Parms;
                       translate : Boolean);
       EXTERNAL;

  PROCEDURE Copy_Opaque( source_l, source_h, dest_l, dest_h, op,
                         sx, sy, sw, sh, dx, dy, dw, dh : integer ) ;

    VAR
      ctrl : Ctrl_Parms ;
      int_in : Int_In_Parms ;
      int_out : Int_Out_Parms ;
      pts_in : Pts_In_Parms ;
      pts_out : Pts_Out_Parms ;

    BEGIN
      ctrl[7] := source_h ;
      ctrl[8] := source_l ;
      ctrl[9] := dest_h ;
      ctrl[10] := dest_l ;
      int_in[0] := op ;
      pts_in[0] := sx ;
      pts_in[1] := sy ;
      pts_in[2] := sx+sw-1 ;
      pts_in[3] := sy+sh-1 ;
      pts_in[4] := dx ;
      pts_in[5] := dy ;
      pts_in[6] := dx+dw-1 ;
      pts_in[7] := dy+dh-1 ;
      VDI_Call( 109, 0, 1, 4, ctrl, int_in, int_out, pts_in, pts_out, false ) ;
    END ;

  BEGIN
  END.

{ End of gemrast.pas }
```

Listing 2.
Pascal listing.

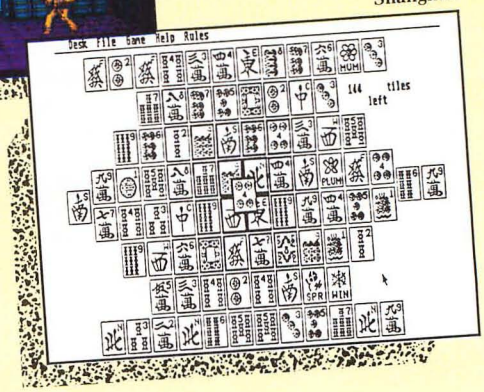
```
{$M+,E+}
PROGRAM Blit_Screen ;


  CONST
    {$I gemconst.pas}

  TYPE
    {$I gemtype.pas}

    Form_Format = ( Device_Format, Standard ) ;
    Mem_Form_Def = RECORD
                     mem : long_integer ;
                     pw : integer ;
                     ph : integer ;
                     width : integer ;
                     format : Form_Format ;
                     planes : integer ;
                     res1 : integer ;
                     res2 : integer ;
                     res3 : integer ;
                   END ;

    Raster_Op = ( ROP_0, ROP_And, ROP_AndNot, ROP_Replace, ROP_NotAnd, ROP_NOP,
                  ROP_Xor, ROP_Or, ROP_Not_Or, ROP_Not_Xor, ROP_NotD,
                  ROP_OrNot, ROP_NotS, ROP_NotOr, ROP_Not_And, ROP_1 ) ;

    Ctrl_Parms = ARRAY[0..11] OF Integer;
    Int_In_Parms = ARRAY[0..15] OF Integer;
    Int_Out_Parms = ARRAY[0..45] OF Integer;
    Pts_In_Parms = ARRAY[0..11] OF Integer;
    Pts_Out_Parms = ARRAY[0..11] OF Integer;
    Inq_type = (Open_Vals,Ext_Vals);

  {$I gemsubs.pas}

  PROCEDURE Copy_Opaque( VAR source, dest : Mem_Form_Def ; op : Raster_Op ;
                         sx, sy, sw, sh, dx, dy, dw, dh : integer ) ;
    EXTERNAL ;

  FUNCTION Get_Port : Integer;
    EXTERNAL ;

  PROCEDURE Port_Inquire(Port_handle : Integer; VAR Int_Out : Int_Out_Parms;
                         VAR Pts_Out : Pts_Out_Parms; which : Inq_type);
    EXTERNAL;

PROCEDURE Blit_Screen(x,y,w,h,dir : integer ; scrn_ary : long_integer ) ;

    VAR
      screen,screen_form : Mem_Form_Def ;
      int_out : Int_Out_Parms ;
      pts_out : Pts_Out_Parms ;

    BEGIN
      WITH screen DO
        BEGIN
          mem := 0 ;
          pw := w;
          ph := h;
          width := (pw+15) DIV 16 ;
          format := Device_Format ;
          Port_Inquire( Get_Port, int_out, pts_out, Ext_Vals ) ;
          planes := int_out[4] ;
          res1 := 0 ;
          res2 := 0 ;
          res3 := 0 ;
        END ;
      WITH screen_form DO
        BEGIN
          mem := scrn_ary ;
          pw := w;
          ph := h;
          width := (pw+15) DIV 16 ;
          format := Device_Format ;
          Port_Inquire( Get_Port, int_out, pts_out, Ext_Vals ) ;
          planes := int_out[4] ;
          res1 := 0 ;
          res2 := 0 ;
```

(Listing continued on page 80)

Karate Kid II



Shanghai




# Software on the Orient Express

## The Eastern influence in ST gaming.

**by Bill Kunkel, Arnie Katz and Joyce Worley**

Just before Australia, and considerably after England, the Orient (in particular, Japan) became "hip," the subject of intense, faddish scrutiny. Black silk dressing gowns, oriental pictographic symbols on T-shirts, and a surfeit of dragons, gigantic robots and karate-wielding ninjas characterized the "orient-ation" of the culture vultures.

If something looked Eastern, it sold. Chinese cuisine was reborn through the popularity of Szechuan restaurants, and Japanese-style tempura/sushi bars sprouted like mushrooms after a rainfall. Harried businessmen studied esoteric martial arts like kung fu and tai kwan-do. And everyone bought Sonys and Subarus. One company, started by an American original, even adopted a Japanese-sounding name before venturing into, and eventually redefining, the then-burgeoning video game business. The American original was Nolan Bushnell, and his company was called Atari, from the ancient oriental game Go.

Nothing enters the pantheon of pop culture without being reflected in the games society plays, and this oriental obsession is currently on view in several ST entertainment programs.

The most authentic expression of Eastern thought in an occidental game is **Shanghai**, published by Activision (P.O. Box 7287, Mountain View, CA 94039 — (800) 227-9759) and written by Brodie Lockard. A modern evocation of the ancient game of mah-jongg, **Shanghai** consists of a multilayered game surface with 144 "tiles"—108 suit tiles, 12 dragon tiles, 16 wind tiles, 4 season tiles and 4 flower tiles. The player removes tiles by matching up pairs, but there's a rub: only "free" tiles can be removed. Free tiles are those that can slide from the formation to the left or right. Tiles are free because they're either on the perimeter of the formation, or stacked in one of the higher rows. The centerpiece tile in the formation, for example, constitutes the fourth level. There are 4 tiles on the third level, 30 tiles on the second level, and the remaining 109 tiles make up the first level.

**Shanghai** allows the player to experience the oriental mindset to an unusual degree. The game cannot be played at high speeds, even when working under time constraints. Patience is **Shanghai**'s prime requirement. It develops in the user an ability to see the "big picture," to consider all the possible moves—and the implications of each. (If you remove that pair of bamboos, you may never get to liberate the season tiles underneath them.)

**Shanghai** is a beautiful and compelling game, crafted with a gentle, poetic sensibility. The tiles themselves are miniature works of art, whether featuring simple dots, elaborate pictograph characters or symbolic representations of the elements.

The gentle side of orientalia has rarely been captured so well in a computer game. More often, quite naturally, a game creator's eye falls upon the more active elements of Eastern culture.

**Karate Kid II**, for example, does *not* require the player to deal with the philosophical question of when violence is required—if you wanna play this game, it's necessary. MichTron (576 S. Telegraph, Pontiac, MI 48053 — (313) 334-5700) has, however, provided more than just another clone of Data East's Karate Champ coin-op.

**Karate Kid II** recreates several scenes from the popular film, including the obligatory *mano-a-mano* between the player, controlling the **Kid**, and the totally contemptible bully. This is, of course, the game's centerpiece, and it works pretty well; but the program also includes some more offbeat challenges. In one, the player must use the controller to manipulate a pair of chopsticks in hopes of capturing a very frisky housefly, while another recreates the scene in which the **Kid**'s aged mentor takes an ornamental wind fan out for a spin.

The visuals incorporate several stunning backgrounds and one or two "digitized" renderings that greatly enhance the graphic realism.

Speaking of stunning backgrounds and Karate Champ clones, Epyx (600 Galveston Drive, Redwood City, CA 94063 — (415) 366-0606) has also joined the martial arts madness with a contest that fits both characterizations perfectly.

**World Karate Championship** is a virtual

duplicate of the Data East program, but for one thing: it features the most eyeball-popping background graphics ever seen on the ST. Whether the combatants punch one another out in Paris, rumble in Rome or square off in Switzerland, there's as much to see in the background as there is in the action-packed foreground. Each new match is fought in a different locale, and players can leap and thrust and generally practice the ancient art of the "empty hand" in all the major cities on Earth. The nightlights and vivid neon of Paris, the snowy Alps, the Brooklyn Bridge in New York City, Tokyo and Mt. Fuji—all of these and more turn up as gorgeous backdrops, complete with two moving objects per screen (not counting the combatants, of course).

The combat itself is excellent, if unoriginal. The player uses joystick commands to summon eleven different maneuvers. In addition to the standard combat screens, there are several bonus racks as well. These include brickbreaking, leaping and dodging contests that will, as with most of **World Karate Championship**, be familiar to experienced gamers.

Chinese history and legend provide the material for **Golden Path** (Firebird Software, P.O. Box 49, Ramsey, NJ 07446 — (201) 444-5700), an action-adventure first published in the United Kingdom. In this oriental fantasy, Y'in Hsi, a 16-year-old novice in the Brotherhood, takes the first steps down the road to becoming the Golden Emperor.

The monks send him on the Path of Enlightenment, so that he may profit from its lessons. As the hero travels around the countryside, he solves puzzles, fights enemies with martial arts, interacts with characters and seeks knowledge.

The mouse controls a Chinese symbol on the screen. Moving this ornate cursor away from the hero and clicking the left button causes the character to walk toward it. If Y'in Hsi reaches a fork in the road, the position of the symbol determines which path he takes. The monk treads the high road if the symbol is above him, but takes the low one if it's near the bottom of the screen.

Other mouse movements allow the character to pick up and drop objects, kick or punch a foe, or use objects. The latter involves a particularly charming visual device. When positioning the Chinese symbol on the hero while he has an object in his hands, the player may see a question mark above Y'in Hsi's head. This indicates the protagonist is trying to apply the object to the situation. If he accomplishes anything, the question mark changes to an exclamation point.

The artwork on the numerous, nonscrolling game screens is appropriate to the theme and quite good, but its very lushness might make it hard for some computer users to pick out small objects. Some of the items are hard to spot, due to the profusion of colors used in the background. Fortunately, the descriptions in the Book of Knowledge, at the click of a mouse button, resolve any ambiguity.

Directly below the main visual display is the Vine. This withers as the game progresses, to show Y'in Hsi's waning lifeforce. Unenlightened actions (kicking an old man) diminish lifeforce faster. Solving a puzzle slows the energy drain.

Below the Vine, from left to right, are the Book of Knowledge, boxes which represent the monk's four pockets, and a small screen which highlights the Path of Enlightenment in the area shown in the main visual display.

The Book of Knowledge corresponds to the room descriptions in most other adventures. The text, especially the boldface words, gives the character an indication of what should be done at that location.

Each pocket holds one item. Clicking on a full pocket transfers its contents to the monk's hands. Clicking on an empty pocket while the monk is already holding something puts that object away.

The music attempts to evoke the mysterious East. More melodic variety would have improved the overall effect, but what's here is nicely programmed.

As with too many British software products, the documentation is a little sketchy. The introductory story is surprisingly readable and informative—a model for inclusions of this type. The actual game instructions, though, should have discussed each aspect in more depth. They cover most of the *essentials*, but examples of play—and, possibly, a couple of diagrams—would make it easier to learn.

**Golden Path** goes one step beyond the martial arts action quests—it offers the player a rich feast of intriguing encounters and situations.

The oriental influence on computer entertainment runs a lot deeper than karate simulations. Reversi contests, like MichTron's **Flip Side**, are based on the ancient Chinese Go. Moreover, two of the most important computer games ever designed (Taito's **Space Invaders** and Namco's **Pac-Man**) were created in Japan.

The Eastern mystique continues to grow, and when our perceptions eventually move beyond a fascination with split-toed ninja shoes and silken headbands, our games will be ready to reflect that, too. //

---

## Arcticfox

**ELECTRONIC ARTS**
**1820 Gateway Drive**
**San Mateo, CA 94404**
**Low resolution $39.95**

**by Bill Kunkel**

Combat tank contests have been a staple of electronic gaming since Atari's **Battlezone** rolled over the arcades several years ago, and Dynamix's **Arcticfox** (ST version by Michael Edwards) brings a new level of excitement to this proud genre.

It seems these nasty aliens have set up a base within an all but impenetrable force field in Antarctica, in the year 2005. Ensconced within, they're busily reconverting our atmosphere to a lovely mix of ammonia, methane and chlorine gas.

Our scientists have weakened the force field (I said it was "*all but* impenetrable") at one crucial point, creating a space large enough for a *single vehicle* to pass through. The candidate: the Slye-Hicks MX-100 or **Arcticfox**, a beauty of an ATV especially styled for arctic conditions. The "Fox" can dig in and bury itself in the snow, fire missiles, drop mines and, on iced surfaces, travel better than 100 kph.

The user selects skill level, scenario and starting location, then jumps behind the controls. The console is keyboard activated, with warning lights and speed gauges, plus six primary controls. The first lets the pilot reorient the cannon position; the second lets him toggle between a secondary screen's radar and rearview display; the third drops mines; the fourth is a relative reverse, which works in concert with

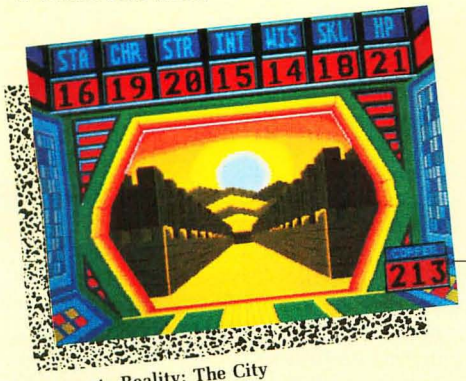
# Alternate Reality The City

**DATASOFT**
**19808 Nordhoff Place**
**Chatsworth, CA 91311**
**(818) 886-5922**
**Low resolution $39.95**

**by Bill Kunkel**

**Alternate Reality: The City** is a role-playing fantasy contest with science fiction overtones, nice graphics and considerable play value.

The science-fiction pretext has aliens kidnapping humans and depositing them in a standard sword-and-sorcery-driven setting, "Xebec's Demise." The player, cast as one of the prisoners, awakens in a strange, hi-tech vestibule overlooking the city. Atop the doorway is a series of constantly-changing numbers, representing values to be assigned to the player. The SPACE BAR stops the numbers and sets stamina, charisma, strength, intelligence, wisdom and skill.



Alternate Reality: The City

The character moves into Xebec's Demise, seen in a center-screen window. Player attributes, etc. are listed across the top of the screen, as well as experience, hit points (damage he can absorb before dying), food and water. The current location appears above the visual display and the macro-location (city) and conditions appear below it.

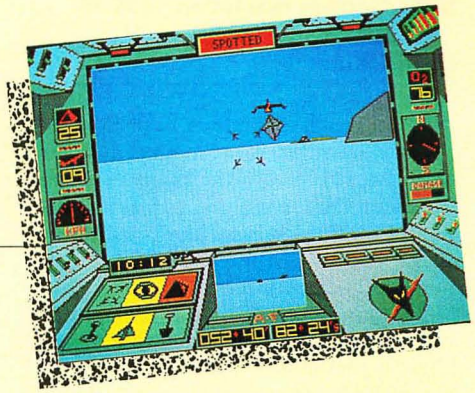
The city consists mostly of primitive stone structures. You can visit shops; enter banks for transactions; drop by a tavern; or even get a job. Just walking the streets, the player will encounter muggers, maniacs, roving magicians and worse. Interaction is handled, as are all game situations, with menus. In a confrontation, the player can fight, charm or trick his way out of danger, based on that character's strength, charisma and intelligence —or lack of same. In some situations, the character could be helpless, facing a vastly superior adversary, in which case the message *No Option* appears.

**Alternate Reality: The City** plays in real time; you can't wait long to make a decision, since events continue, independent of action or inaction on your part.

The visual window presents the character's view of the environment, with movement actuated via direction keys or access to an on-screen, compass-style rosette. Graphics are impressive and full of variety, vivid colors and limited animation. Interiors are all nicely realized.

Robert Mirsky, Jim Ratcliff, Phillip Price and Steve Hoffman, creators, have slickly mined territory previously explored in role-playing D&D-type programs, but with far more emphasis on making the game an appealing audio-visual experience. As **The City** is, theoretically, the first in a series, it is hoped future installments will move beyond the all-too-familiar swords-'n'-spells trappings which define this first episode.

The program includes a prelude, an ex-



Arcticfox

the Cannon Inclination control and lets the Fox reverse its direction; the fifth fires missiles; and the final control accesses the "Dig In" function.

The joystick-controlled cannon fires tank shells, which are periodically reloaded by the system (the gunsight alters to indicate loading status). The guided missiles are also under total joystick control, from launch to impact.

When the player fires a guided missile, the secondary display screen shifts to a view from a camera mounted *on the missile itself*, one of the most unique and visually powerful ideas ever executed in a game of this type. The result is not only a weapon that can fire over and around mountains but one that provides constant surveillance information as it moves along.

The graphics are uniformly excellent, effectively communicating not only the gradual white-out conditions during the onset of a sudden blizzard, for example, but also the glistening, frosty aftermath, as even the normally blue ice is coated with a patina of white crystal.

The topography is wonderfully realized, with Dynamix living up to their boast that "We build worlds!" **Arcticfox** is internally consistent. Park your vehicle on an inclining hillside and it will roll down. There are deep crevices from which no craft ever returns, mud flats (which are hell to travel over), ridges, rocks, mountains and snowfields. In addition to blizzards and fog, savage lightning storms occur with greater frequency as more oxy gen is converted by the invaders; and lightning knocks out even radar.

**Arcticfox** mixes the juggernaut power and heavy-metal rumble so essential to these games with a dollop of strategic nuance that keeps it playable for *many* hours. Unlike EA's earlier game of similar title, the rather soggy **Skyfox** (which was not designed by Dynamix), this game delivers everything the user could reasonably expect—and then some.

It's a cold world out there, but you've got the perfect tank to face it in. //

tended introductory sequence which not only visually sets up the kidnapped-by-aliens motif, but also displays credits in motion-picture fashion, complete with theme song (the dreadful lyrics appear on-screen as it plays). Unfortunately, there seems to be no way to avoid this wah-wah on subsequent plays, a traditional annoyance in DataSoft products.

**Alternate Reality: The City** is a role-playing adventure for those who've been turned off by the lack of a visual component in earlier contests of this type. Moreover, the ST version represents a considerable improvement, visually, over previous editions. This speaks well of DataSoft's commitment to the system and makes one anxious to see sequels produced with the 16-bit technology in mind. //



## Golden Oldies

**ELECTRONIC ARTS**
**1820 Gateway Drive**
**San Mateo, CA 94404**
**Medium resolution $34.95**

**by Bill Kunkel**

History puts everything in perspective. We see how far we've come, and how close we still are to the point of origin.

**Golden Oldies** is a great idea: a collection of four classics for the price of one. These capsulize the early days of computerized entertainment, from the prototype adventure (titled, aptly enough, **Adventure**) to an electronic shrink named **Eliza**, a metaphorical game of **Life**, and the very first major video game, **Pong**.

Unfortunately, they must be *more* than mere historical benchmarks and curiosities; they must stand as games. And these venerable classics crumble to the dust of antiquity with their first exposure to modern sunlight.

**Adventure** was a favorite of mainframe mavens back when only MIT techies had access to computer entertainment. Its simple virtues, have hardly endured the passage of time. **Adventure** is a text adventure so primitive that only historians and sociologists could have *any* interest in actually playing it beyond the first five minutes.

**Eliza**, the computer psychiatrist, is a shrewd and knowing parody of the more witless practitioners of psychoanalysis. A wag of a programmer observed that many techniques of psychoanalysis could be practiced with only the most passing of interactions between patient and analyst. So he concocted a collection of noncommittal phrases ("How do you feel about that?") and all-purpose reactions ("You say [repeat whatever patient said]. Why do you feel that way?"), to produce a virtual databank of psychobabble.

The result is an amusing program that drew smiles when displayed on home computers in the late 70s. Once the contemporary user pierces this rather transparent parlor trick, there's no reason to continue. Unlike its evolutionary descendant, Mindscape's **Racter**—which at least *attempts* to create the illusion of interaction with a computer intelligence, albeit clumsily—**Eliza** is a parade of cynical cliches without even the pretense of genuine dialog.

**Life** was another early staple, especially on color systems like the Atari 400/800. The player lays down a series of multicolored pixels on-screen, then hits a "run" command and watches as the processes of life are displayed. Concepts as metaphysical as entropy (the process by which all things wear down) can be perceived in terms of physical reality, as generations of pixels are born, live and die. The "population" and "generation" are tracked on-screen.

This is all lovely, educational and even marginally entertaining, but the Joshua Light Show it most definitely is not. No ST owner is going to spend any significant amount of time watching these tiny clusters go through their none-too-eye-popping life cycles.

Finally, we have **Pong**, its origins clouded by litigation (Nolan Bushnel "invented" the game and gave it its distinctive name, but only after an assistant saw a demonstration of a similar program by Ralph Baer), but still strong in terms of play value. Unfortunately, the version here (by Software Toolworks) lacks the sophistication of the coin-op. It doesn't even compare favorably with the dedicated-chip **Pong** clones which flooded the home market in the early 70s. Virtually every family should have a video game system somewhere in their closet that plays a better game of computer table tennis than this one, with its limited angles and easy-to-beat AI opposition.

From a historical point of view, **Golden Oldies** is an interesting collection of obsolete forms of computer entertainment. Unfortunately, from an entertainment point of view, it is a gallery of outdated oddities, bereft of even the slightest potential for user enjoyment. //

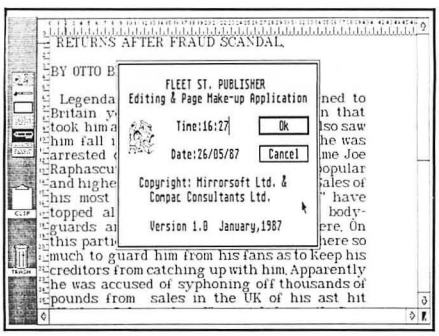
# ST news

## From Fleet Street.

The third desktop publishing package available for the ST is brought to you by Fleet Street of England. Their powerful package gives users the ability to create multicolumn newsletters, publications, business reports, brochures, ads, flyers—or anything else which requires text or text mixed with graphics.

**Fleet Street Publisher** highlights features such as full-page composition and multiple columns, text editing in page makeup, variable-sized text blocks, picture sizing and cropping, auto-word-wrap and graphics handling. Some of its many other functions include: a library of ready-to-use graphic images, dot-matrix output upgradeable to laser and GEM compatibility. **Fleet Street Publisher** works on monochrome systems only, with an Epson FX-80 or compatible printer, on a 520ST or larger.

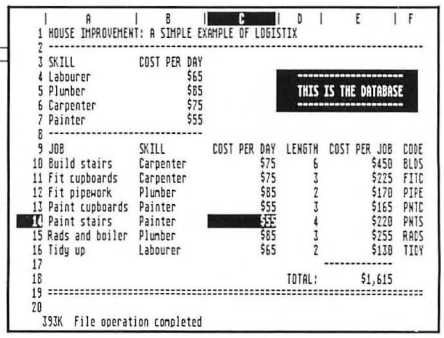
Priced at $119.95. When laser printer drivers are installed, the **Publisher** will be $149.95. From Mirrorsoft Limited—distributed through Spectrum HoloByte, 495 Third Street, San Francisco, CA 94107 — (415) 882-7381. Reader Service #126.




## All in one. . .

**LOGiSTiK** combines a powerful spreadsheet and database with a time and product manager, plus color graphics capability. The spreadsheet matrix is 2048 rows by 1024 columns, with many commands. The database lets you store, sort, find and edit data, as well as integrate information with the spreadsheet, graphics and time/product manager. Other formats—like Lotus, dBase, ASCII and Supercalc—can be read.

The management end of **LOGiSTiK** can be integrated with cash flow projections, income statements, etc. Time planning and calculation of "Critical Planning" can also be accomplished. Graphics turn raw data into easily understood charts and diagrams.

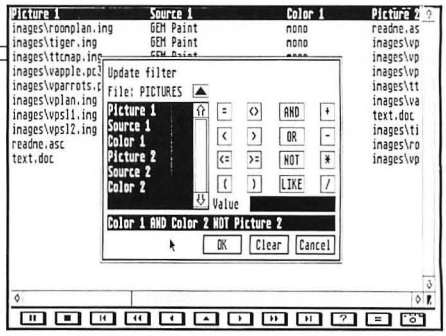



Retail is $149.95. A junior version, without graphics capability, is $99.95. From Progressive Peripherals & Software, 464 Kalamath Street, Denver, CO 80204—(303) 825-4144. Reader Service #127.

## Relational database.

Precision Software of England now offers **Superbase Personal**, an easy-to-set-up database with multiple windows, pull-down menus and click-and-point selection "buttons." Up to 16,000,000 (!) records are stored per file, as many as 999 indexes per record. Number of files, file size, open files and number of fields per record are determined by the memory of your ST. Calendar functions with multiple date styles, 13-digit numeric precision and input range checking are also provided.

Reports up to 255 characters wide can have time/date stamping and page numbering, plus multicolumn formats, sorting and variable-format mailing labels. Note, too, that **Superbase** can access a special "picture reference facility" with data handling, so a picture/data library can be formed.




The **Superbase** software works with medium (color) and high (monochrome) resolution monitors.

For more information, contact Precision Software Ltd., Park Terr., Worcester Park, Surrey KT4 7JZ, U.K. — 01-330 7166. Reader Service #128.

## Floyd is back!

Remember Floyd, your ol' robot pal in **Planetfall**? Do you remember **Planetfall**? Well, Infocom's latest interactive fiction puts you on another assignment in outer space: pilot a spacetruck to a nearby station to pick up a load of trivial forms. Sound exciting? Nope, it's not. . .until you arrive at the station to find it deserted—with an eerie alien ship in the docking bay and some weird ETs about. Oh—the name of this latest entertainment software is **Stationfall**.

Also released is **The Lurking Horror**, which is said (by Infocom) to give you the flavor of an H.P. Lovecraft and/or a Stephen King story. Many puzzles, chilling corridors, strange sounds and shadowy creatures await you in the damp old campus basements of G.U.E. Tech. You've heard the myths; now it's time to explore.

In traditional Infocom. . .well, tradition, the packages come with goodies and (intentionally) little instruction. They can be yours for $39.95 each. Infocom, 125 CambridgePark Drive, Cambridge, MA 02140 — (617) 492-1031. Reader Service #129.

## Other news.

☑ Artworx is shipping its latest program for the ST: **Minigolf**, a miniature golf game. The player can select from the three different courses, complete with various obstacles that'll cut down opportunities to make a hole-in-one.

A score card displays the low score for each hole and tracks up to four players. In addition, an editor permits you to create your own course or modify an existing one.

For $19.95, from Artworx Software Company, Inc., 1844 Penfield Road, Penfield, NY 14526 — (800) 828-6573. Reader Service #131.

☑ The creators of **Analyze!** call their program a powerful electronic spreadsheet, which allows the user to enter any type of mathematical data—permitting one to keep track of anything from checkbook records to a company's general ledger.

Data is entered in rows and columns, and can then be graphed if desired. Pull-down menus control commands such as insert, copy, name, label, erase and lock—to keep a spreadsheet from accepting additional data. The same commands can also be entered directly from the keyboard.

Other features include: sorting; print functions like boldface, italics and underline; graphs—pie charts, bar and 3-D; as well as margin settings with headers and footers.

At $149.95. Contact Christopher DeBracy, Micro-Systems Software, Inc., 4301-18 Oak Circle, Boca Raton, FL 33431 — (305) 391-5077. Reader Service #102.

☑ **Custom fonts** for SoftLogik's **Publishing Partner** are available in a series of four disks, with three fonts per disk, retailing for $15.95 each.

Soon to be released is a **Clip Art** series, also for **Publishing Partner**, with over 100 pictures on each $19.95 disk.

For more details, contact The Font Factory at P.O. Box 17422, Phoenix, AZ 85011. Reader Service #132.

☑ The Pierstorff Company's latest release is **More! Graphics ST**, a collection of 128 clip-art icons, letters and symbols, all of which can be used in **DEGAS** and **DEGAS Elite**, **PrintMaster** and **PrintMaster Plus**, **Typesetter Elite**, **PM Interface** and **Publishing Partner**.

The graphics can be added to hard disk or backed up onto a floppy. Compatible with monochrome or color systems. Priced at $14.95, from The Pierstorff Company, 131 W. Main Street, Woodland, CA 95695 — (916) 666-3530. Reader Service #133.

☑ Micro League Sports Association announces the release of **MicroLeague Baseball II**. Utilizing the ST's large memory capacity and enhanced graphics, the authors have revamped their simulation to advance it even beyond the great original.

Some new features: rain delays, arguments with the umpire, injuries, pitcher stamina and tiring factors, and stadium dimensions and other factors which affect play.

Retail is $59.95, which includes the box score stack compiler previously sold separately. Micro League Sports Association, 2201 Drummond Plaza, Newark, DE 19711-5711 — (302) 368-9990. Reader Service #125.

## Atari ST Desktop Publishing.

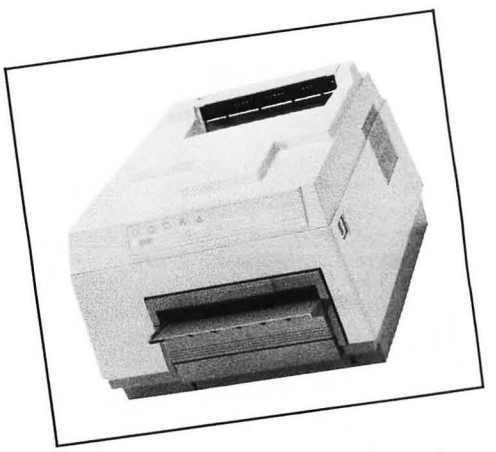
This future release will consist of the Atari **SLM804 Laser Printer**, as well as an Atari-designed "Desktop Publishing Software" package.

The **SLM804** is made up of two parts: the Laser Printer Controller and the Laser Printer Engine. The former resides on the hard disk DMA bus and uses an ACSI device host software interface. The latter is comprised of an electrophotographic mechanism with scanning semiconductor laser. Speed is said to be 8 pages per minute, with a resolution of 300 by 300 dots per inch. Letter, legal-sized and A4 paper may be used, plus envelopes and transparencies.

Manual feed or automatic, and up to 250-page input capacity with face-down output are also provided. Separate toner and drum units, with 3,000 pages per toner cartridge and 10,000 pages per drum. Status indicators show ready, wait, add paper, check paper path, and replace toner signals. Mode selectors on the Controller change horizontal dots/inch resolution and top, bottom, right and left margins. Mode Sense checks image width and height, horizontal and vertical dots/inch resolutions, plus top, bottom, left and right margins, and fat or skinny pixels.

The software package requires a Mega ST or 2-megabyte upgrade. A variety of font styles are available, including: Oldstyle Serif, Square Serif, Sans Serif and Non-Connecting Script. The actual desktop publishing software is still in the works. However, at this time, **Easy-Draw** from Migraph works "out of the box," and the soon-to-be released **Microsoft Write** will, also.

The price is tentatively set at $1500.00. Future products are to consist of an Image Scanner, Fax Modem, External Frame Buffer, Raster Image Processor, CD ROM, Page Description Language and OCR Software.



Watch the pages of **ST-Log** for more news, as it becomes available, about these upcoming products from Atari Corp., 1196 Borregas Ave., Sunnyvale, CA 94086.

## The real, true BASIC.

**Version 2.0** of **True BASIC** has just been released for the ST, developed by John G. Kemeny and Thomas E. Kurtz, two of the original authors of BASIC at Dartmouth College back in 1964. This ST version of the language is said to offer an unprecedented level of graphics support, plus modules (which can be compiled and stored separately as libraries for use in other programs) and compilability.

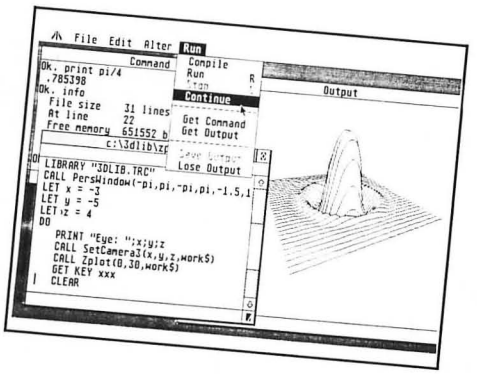
Examples of this newly designed language's advantages can be found in: graphics—which can be plotted to your own coordinates rather than in pixels; speed—**True BASIC** is much faster than interpretive BASIC (before execution your program is compiled into an intermediate code); external programs—space is saved by storing frequently used functions and subroutines into user-defined libraries.

**True BASIC** also runs on the IBM PC, Macintosh and Amiga, giving users portability between systems.

Additional support for **True BASIC** can be found in a number of libraries, also available from True BASIC, Inc., consisting of a Sorting and Searching Library, Advanced String Library and a Mathematician's Toolkit.

**True BASIC**'s special introductory price is $79.95, with a fully documented Reference Guide and Atari User Manual. The separate library packages are $49.95 each.

Releases to come: Runtime Package, 3D Graphics Library, Math Series Libary and an Atari ST Developer's Toolkit. True BASIC Inc., 39 South Main, Hanover, NH 03755 — (603) 643-3882. Reader Service #131.




## Continued ST support from Abacus.

Abacus Software has finished the two latest books in their ST series.

***ST for Beginners*** (first in the series) is tailored for the person who's just purchased an ST and doesn't know where to start. Some of the many topics covered are: an overview of the ST system, GEM, window managing, operating disks, ST BASIC, ST Logo and overall ST usage. The 202 pages sell for $16.95.

***ST Disk Drives: Inside and Out*** (thirteenth in the series) is a more technical look at the ST's drives. The file handling, data structures and file programming techniques are closely examined, as well as the SH204 Atari Hard Disk. Other chapters explain the RAMdisk, programming a disk monitor, and machine language utilities for BASIC. The 403-page book is $24.95.

These books make fourteen volumes now comprising the Abacus library, with still more planned. Abacus Software, P.O. Box 7219, Grand Rapids, MI 49510 — (616) 241-5510. Reader Service #132.

## Easy-Draw update.

**Easy-Draw** version 2.0 has been available since January. The new version will load text or ASCII files and lets you edit polylines. You can also flip/slash mirror objects and break text blocks. Version 2.0 provides a choice of metric/inch page sizes and ruler measurements, and much more. Also included in the update is a new manual.

For an update, owners of earlier versions should send their two original disks (master and drawing), with $26.00 plus $3.00 for shipping/handling (not $3.00, as reported in the May **Reader comment** of **ANALOG Computing**).

Readers can contact Migraph, Inc., 720 South 333rd Street, Ste. 201, Federal Way, WA 98003 — (206) 838-4677. Reader Service #108.

## RS232 programmable interface.

**The Spider** is a new piece of hardware that's a completely programmable RS232 interface, capable of supporting as many as eight simultaneous RS232 devices.

Utilizing recently introduced technology, the interface can individually drive each of the eight ports at an their own baud rate—with sixteen available, from 50 to 19,200 baud. The standard 256K of RAM is expandable up to 4 megabytes, and can be used as a buffer for serial peripherals.

**The Spider** can also be used as a stand-alone system to run multiline BBSs, etc. **The Spider** comes complete with documentation covering use and technical aspects of the unit.

Retails for $349.95. Nite Lite Systems, P.O. Box R, Billerica, MA 01821 — (617) 663-4463, or modem hookup at (617) 663-4221 (300/1200-baud, 24 hours). Reader Service #130.

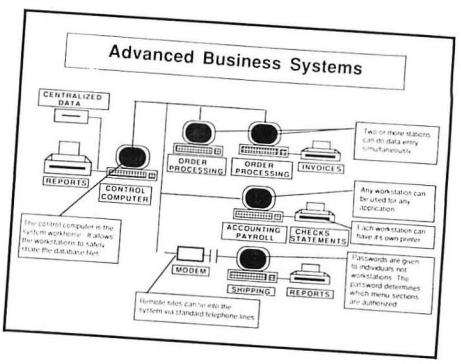
## Advanced Business System.

The nice people at Beckemeyer Development Tools, well known for their excellent line of "software tools" in the ST world, have announced a line of business products. Multiuser/multitasking systems for the business world provide up-to-date sales charts and graphs, accurate financial statements, cash flow management, inventory control, order entry and customer files.

A cash-register and point-of-sale system allows you to enter the quantity and item number or description of a sale, which permits the computer to record the sale, print the invoice and adjust the inventory. An additional feature even tells you which items are moving the fastest and which are slow. Mailing labels and customer tracking are also available.

This software package sells for $2,495.00. Additional products available: **General Ledger Accounting** software for $395.00; a steel, auto-open cash drawer at $295.00; a bar code reader to speed up checkout time, $635.00; and a color touch screen workstation which eliminates the need for a keyboard, $2995.00.

Contact Beckemeyer Development Tools, 478 Santa Clara Ave., Ste. 300, Oakland, CA 94610 — (415) 452-1129. Reader Service #110.



# Introducing

# LabelMaster™ Elite

Here's Migraph LabelMaster Elite™, a fun easy-to-use GEM based label maker and mailing list manager. With LabelMaster Elite, you can create distinctive custom labels that can be used for business mailings, gift tags, disk labels, and much more!

Labels may be printed with designs included with the program or created with the feature-packed Graphics Editor. Print labels with or without designs in a variety of formats including mailing labels, 3.5″ disk labels, and 3x5 and 4x6 index cards.

The mailing list manager has all the expected functions plus the ability to flag a record for personal or business use as well as a 48 character comment field.

Once you use LabelMaster Elite you'll never use plain labels again!

---

**Graphics Editor**

*100 designs on disk *Line box, circle & mirror drawing commands *Clipboard to cut & paste between designs *Compatible with Printmaster designs

**Mail List Manager**

*Search, sort, delete & modify records *48 character comment field *Ability to merge databases *Store thousands of names *Freestyle label with 4 lines of text in varying styles

**Additional Features**

*Works with most 9 & 24 pin printers *Print up to 11 lines of text on disk labels and index cards *Mix print pitch and style on the same line by using special printer codes *Print disk directories

Current LabelMaster owners may update to Elite by returning their disk and $5 to Migraph.

---

MIGRAPH Inc.

Migraph, Inc.
720 S. 333 St., (201)
Federal Way, WA 98003

To order call:
206 838-4677

LabelMaster Elite is a trademark of Migraph, Inc. Printmaster is a trademark of Unison World

CIRCLE #108 ON READER SERVICE CARD

# ATARI 8-BIT POWER

These products are custom built to work with Atari 400, 800, 1200XL, 800XL, 65XE, 130XE

**NEW BOOKS FOR ATARI OWNERS**

**BASIC TURBOCHARGER** NOW for the first time a BASIC programmer can get the power, flexibility and incredible speed of machine language BASIC TURBOCHARGER is a **book and disk package** that contains over 150 ready to use machine language routines. Complete instructions show how to add them to your own BASIC programs to get these features and more; • Smooth Scrolling • Load & Save Picture files • Special Effects Graphics • Much, Much More • Player/Missile control • Sorting and Searching • Incredible Speed • Over 150 programs Only **$24.95**

**YOUR ATARI COMES ALIVE**

SAVE MONEY. Finally an alternative to buying expensive computer add-ons. Your Atari Comes Alive shows you how to **build them yourself**. This 'How-To' **book and disk package** gives you complete step by step instructions and programs needed to build and control these exciting devices and MORE; • Light Pen • Alarm Systems • Environmental Sensors • More than 150 pages • Light & Motor Controllers • Voice Recognition • Data Decoders **Your Atari Comes Alive $24.95**

**Atari Software Protection Techniques Vol I & II**

These Book and Disk packages detail the most advanced copy protection methods in use today. They guide you through the methods used to create the protection as well as the copying techniques to get around them. They include information on Phreaking • Hacking • On-line security • Black boxes • Self-destructing programs • Pirate bulletin board systems • Logic bombs • New piracy laws • Hardware data keys • Weak sectoring (Phantom, Fuzzy and unstable sectors) • Overfilled tracks • CRC errors • Bank Select cartridges and MUCH, MUCH MORE. The disks include automatic program protectors, Protection Scanners, directory hiding and more.

**BOOK I and DISK I** **$24.95**
**BOOK II (Advanced protection) and DISK II** **$24.95**
**Special Offer, Order both sets for Only** **$39.95**

**NEW from Alpha Systems Chipmunk**

Automatic Disk Back-Up System. Make perfectly running unprotected back-up copies of hundreds of the most popular Atari programs. Chipmunk's sophisticated programming Automatically finds and **REMOVES** copy protection from most Atari programs. Back-up even heavily protected programs with ease. Finally, a back-up system that needs no special hardware or skills. **$34.95**

(If you need a full list of what Chipmunk copies, call or write for our free catalog)

**COMPUTEREYES™ & MAGNIPRINT II+**

Turn your computer into a digital portrait studio.

This complete package lets you **capture, save & print** digital images from your **Video Camera, VCR or TV.** COMPUTEREYES hardware plugs directly into your joystick ports for easy use. Print your picture on a 6 foot poster. **$119.95**

**ComputerEyes camera system**

Comes complete with everything above, plus a black and white video camera and connecting cable. **$329.95**

**Graphics 9 Software** Add a new dimension to your COMPUTEREYES pictures – captures images in 16 shades of grey. **$12.00**

**Magniprint II+** Easily the most powerful print program available today. Print graphics from almost any format in hundreds of shapes, sizes, and shades. Supports **color printing** and lets you create **giant posters.** Magniprint II+ lets you stretch and squeeze, invert, add text, adjust shading and much more.

Works with EPSON, NEC, Citoh, Panasonic, Gemini, Star, XMM801, and compatible printers. (850 interface or equivalent required). **$24.95**

**Graphics Transformer**

Now you can combine the most powerful features of all your graphics programs. Create print shop icons from a Koala pad picture, from a photo digitized with Computereyes, or any picture file. Graphics Transformer lets you **Shrink, Enlarge and Merge** pictures for unequaled flexibility. **$22.95**

**SUPERSONATOR** Cartridge to Disk back up system. Create running back-up copies of any cartridge (up to 16k) **$29.95**

**PARROT**

The hardware & software you need to digitize real world sounds – play them on your computer. Make your Atari speak in your own voice or record and play sounds from a complete orchestra. Play the digitized instruments and sounds from your keyboard, or put them in your own BASIC programs.

PARROT is now being used for special effects by professional musicians. Turn your Atari into a digital sound sampler and keyboard. This package is better than those costing more than twice as much.

Everything you need to record and play digital sounds **$39.95**

Additional pre-recorded sound tracks **$4.95**/with purchase of PARROT

**THE Scanalyzer**

Extract the programming secrets for any Atari program. Scanalyzer easily breaks through protection to let you list and study programs. Works with programs on Disk or Cartridge, finds hidden directories, detects copy protection methods and more. ONLY **$29.95**

Call or write for our **FREE CATALOG** explaining our full line of ST and 8-bit products.

**Customer Service line**
**(216) 467-5665.**
9am-3pm EST M-F

ALPHA SYSTEMS

– VISA & MasterCard

**Order by phone or send check or money order to:**

**24 HOUR HOTLINE 216-374-7469**

**ALPHA SYSTEMS 1012 Skyland, Macedonia, OH 44056**

Include $3.00 shp. & hdlg. (US & Canada). Ohio residents add 5½% sales tax. Foreign orders add. $8.00 shp. & hdlg.

CIRCLE #109 ON READER SERVICE CARD

# C-manship

## The ins and outs of menu bars.

by Clayton Walnum

Now that we know everything there is to know about dialog boxes (well, maybe not *everything*), it's time to move on to menu bars. Second only to windows, menu bars are one of the most characteristic features of GEM. Because they're an excellent way to organize the large number of options complex programs offer the user, virtually every GEM program uses them.

You'll be surprised to hear that menu bars are actually much easier to program than dialog boxes. In fact, they're so easy that we'll be able to cover them in just one installment of **C-manship**, rather than the two it took for dialog boxes.

### Another RCP tutorial.

Before we can go any further, you're going to have to load up your Resource Construction Program and create the object tree for the sample menu bar. The following steps will guide you through the entire construction process. It's not as detailed as the instructions I gave in the last RCP tutorial; you should be familiar with using the RCP by now. I can't hold your hands forever—they get too sweaty.

So get to work, and I'll meet you after Step 24.

### Steppin' through the menu bar.

**Step 1:** Click on the "New" selection from the file menu. A window titled NONAME will be opened. Just like when we constructed our dialog box a couple of months ago, this is where we'll work on our menu bar.

**Step 2:** Drag the menu icon from the left of the screen into the newly created window. A dialog box will appear, prompting you for the name of the menu tree. Press RETURN to select the default name of TREE00. The menu tree icon will appear in the work window.

**Step 3:** Double-click the menu tree icon. The beginnings of your menu bar will appear in the work window.

**Step 4:** Give the desk menu selection (on *your* menu bar, not the RCP's) a single click, then press CONTROL-N. The dialog box for naming objects will appear. Name this object DESK.

**Step 5:** Repeat Step 4 for the file menu selection, naming this object FILE.

**Step 6:** Drag the word *TITLE* from the parts list and place it to the right of the file title. Double-click this new menu bar title. A dialog box will appear. Change the text to two spaces followed by the word *Options*, followed by another two spaces.

**Step 7:** Place the mouse pointer on the lower right-hand corner of the title's shaded area and, holding down the left button, expand the shading to the right, centering the title within it. Click once on the options title to select it, then press CONTROL-N and name the object OPTIONS.

**Step 8:** Set up another menu title in the same way, entering the text as two spaces followed by the word *Selections*, followed by two more spaces. Name the object SELECTS.

**Step 9:** Give the desk menu selection a single click, then double-click the "Your message here" entry. Change the text to two spaces followed by *C-manship info. . .* and press RETURN. Press CONTROL-N, and name the entry INFO.

**Step 10:** Give the file menu selection a single click, then place the mouse pointer on the lower right corner of the QUIT object. Holding down the left button, reduce the length of the object by dragging the corner to the left. You have to do this to uncover the menu box beneath.

**Step 11:** Place the mouse pointer on the lower left corner of the menu box and, holding down the left mouse button, drag the box downward, enlarging it so that it can accommodate three more entries.

**Step 12:** Place the mouse pointer on the QUIT object and, holding the left button down, move the object to the bottommost position of the menu box.

**Step 13:** Drag the word *ENTRY* from the parts list and

place it in the top position of the file menu box, making sure you place it as far to the left as it'll go. Double-click it and change the text to two spaces followed by *Load. . .*, followed by two more spaces. Name the object LOAD.

**Step 14:** Create another menu entry below LOAD. The text should be two spaces followed by *Save. . .*, followed by two more spaces. Name this object SAVE.

**Step 15:** Drag the ---------- icon from the parts list and place it below the SAVE entry, all the way to the left. Then move the QUIT object just below it and name it QUIT.

**Step 16:** Reduce the menu box to its smallest size using the same method as when you enlarged it (Step 11). Add enough dashes to the ---------- object (by double-clicking on it and changing the text) to extend it to the right-hand margin of the menu box.

**Step 17:** Single-click the options menu title and enlarge the menu box to accommodate three entries.

**Step 18:** Drag an entry icon to the top of the options menu box. Set the text to two spaces followed by *Option 1*, followed by two more spaces. Before closing the dialog, set the CHECKED option in the attributes list. Name the object OPTION1.

**Step 19:** Create two more entries in the options menu box, named OPTION2 and OPTION3, and place in order below OPTION1. The spacing of the text will be the same as in Step 18. Do not set the CHECKED attribute for these two objects. Reduce the options menu box to its smallest possible size.

**Step 20:** Single-click the selections title, then stretch the menu box to accommodate five entries.

**Step 21:** Create an entry in the selections menu named ONOFF, and enter into the text field five spaces followed by On followed by five more spaces.

**Step 22:** Drag the ---------- icon to a position below the ONOFF entry. Add two dashes to the already existing ten in the text field.

**Step 23:** Create three entries below the dashed line. The entries should be named SELECT1, SELECT2 and SELECT3. Their text fields should contain two spaces followed by *Select n* (where *n* is the entry's number as indicated by the names above), followed by two additional spaces.

**Step 24:** Reduce the selections menu box to its smallest possible size.

### The program.

Now that you've got your version of the menu bar saved in a resource file, type in Listing 1 and compile it. If you use a compiler other than Megamax, you may have to make some slight changes to the source code.

When you run the program, the menu bar you created will come up on the screen (make sure you have the MENU.RSC file on the same disk as your .PRG file). First pull down the desk menu and click on "C-manship info. . ." An alert box should appear, giving you a little information about **C-manship** (very little, actually).

There are two GEM menu conventions used here. First, you should always place the "Info" selection of your menu bar as the first choice of the desk menu. Second, any menu entry that will lead to a dialog box of some sort should be followed by three dots.

If you had any accessories on your program disk when you booted it, they should also be available on the desk menu. Go ahead; check them out.

Now pull down the file menu. There are a few more conventions to take note of here. If you plan to allow the loading and saving of files from your application, this is where the appropriate menu entries should go. Ditto for QUIT commands. When you follow these conventions, users will always know where in a menu bar to find these basic functions. Any other disk handling activities, such as DELETE, should also be located here.

In our sample menu, clicking on LOAD or SAVE just prints a message to the screen. QUIT, of course, returns you to the desktop. Note that the load and save entries are followed by three dots, even though, in this case, they don't lead to a dialog box. Why? Well, when you use them in a *real* program, they'll almost certainly lead to a file selector box, right?

Now we get to the options menu. The three entries found here may be turned on and off by clicking them with the mouse. Any options that are active will have a checkmark next to them. You can have as many of them active as you wish (especially considering, due to the stripped-down nature of the demo program, they don't really do anything, anyway!)

Moving right along, we get to the selections menu. The top entry will toggle between the words *ON* and *OFF* each time it's clicked. When the entry is "on," the selections below will be selectable, and will print a message to the screen when they're clicked on.

When the entry is "off," the other selections will be "grayed out." This means they have been disabled. No amount of clicking on a disabled menu entry will give you the slightest response, except removing the drop-down menu from the screen. This is why we can get away with those dashes separating different sections of the menus. Since they're disabled, the user isn't going to be able to do anything with them.

### Menu bars in your program.

Now that you've had a chance to fiddle with your creation, let's see how the program works. All the action is in the function do_menu(), so let's skip over all the other stuff. You should be familiar enough by now with how to initialize GEM.

First, we declare menu_adr, a longword that'll contain the address of our menu tree. The actual code begins with the initialization of some flags we'll use to keep track of the status of the various menu options. Then we set the mouse pointer to the arrow form.

Next, we load the resource file from disk, using rsrc_load(), which I described last month. If the MENU.RSC file is missing, we warn the user with an alert box, then return to the desktop.

If the resource file loads okay, we find the address of the menu tree, using the rsrc_gaddr() call—which we also discussed last month—storing the returned address in menu_adr.

To display our menu bar, we call:

```
menu_bar (menu_adr, flag);
```

Here, menu_adr is the address of the menu's object tree and flag is a Boolean value indicating whether the menu should be displayed (a nonzero value) or removed (a zero value).

### A nifty message system.

Up until now, we've been able to get our user's input with either specific calls to the mouse or keyboard, or by using dialog and alert boxes. We've now gotten to the point in our GEM programming careers where the simple calls just won't do the job. Why? Well, first of all, what if we want to be alerted to more than one form of input? What if we want to know about the mouse *and* the keyboard at the same time? More to the point for this month, how do we know what the user is doing with the menu bar?

GEM supplies us with a single function call that will monitor the entire system for us and tell us everything we need to know about the user's actions. Are you ready? Take a deep breath, because you're going to need it. The call (as shown in the Megamax manual) is:

```
evnt_multi(mmflags,mbclicks,mbmask,
     mbstate,mm1flags,mm1x,mm1y,
     mm1width,mm1height,mm2flags,
     mm2x,mm2y,mm2width,mm2height,
     mmgpbuff,mtlocount,mthicount,
     mmox,mmoy,mmobutton,mmokstate,
     mkreturn,mbreturn);
```

Sheesh! Didn't I warn you? But before you burn your compilers in frustration and open your wrists to end it all, you should know that, at this point, there are only a few of the above parameters we're interested in. We'll be discussing this function a lot in future installments of **C-manship**, and we'll cover the parts we need bit by bit.

For this month, let's just say that you don't need to use all the parameters. We can send *0*s for any outgoing parameters we're not interested in, and supply dummy locations for any unneeded information being sent back. In our menu bar program, the evnt_multi() call looks like this:

```
evnt_multi(MU_MESAG,0,0,0,0,0,0,
   0,0,0,0,0,0,0,msg_buf,0,0,
   &dum,&dum,&dum,&dum,&dum,&dum);
```

MU_MESAG is the event we want to watch for (in this case, a message event), and msg_buf is the address of a 16-byte message buffer where our messages will be stored.

There are six event types we can wait for. To wait for more than one type of event at a time, we just OR the appropriate flags together. For example, to wait for a message event, a keyboard event and a mouse button event, our first parameter in the evnt_multi() call would be:

```
MU_MESAG | MU_KEYBD | MU_BUTTON
```

Here, MU_MESAG has been defined as *0x0010*, MU_KEYBD has been defined as *0x0001*, and MU_BUTTON has been defined as *0x0002*. This sets the proper bits, as shown in the following table.

| Set bit # | Event |
|---|---|
| 0 | Keyboard |
| 1 | Mouse button |
| 2 | Mouse event 1 |
| 3 | Mouse event 2 |
| 4 | Message event |
| 5 | Timer event |

Note that, even though we're not using the call this way in the sample program, since we're only waiting for one type of message, evnt_multi() returns a word value that will have bits set based on the events that occurred. These bits follow the same format as the table above. It's possible to have more than one event detected by a single call to evnt_multi(), so it's up to the programmer to check each bit in the returned integer, in order not to miss events. If you're checking for more than one type of event, your evnt_multi() call should look something like this:

```
event = evnt_multi(...);
```

Here, event is the integer that will contain bits set based on the events detected by the call. Of course, the . . . in the parentheses will be replaced by that horrendous parameter list.

When we make the evnt_multi() call, everything comes to a stop until the event we're waiting for occurs. If the event is a message event, information about the message is stored in the message buffer. In our program, we've set up the array msg_buf[] to do this duty. The type of message will be returned into msg_buf[0]. There are thirteen possible messages, but the only one we're interested in right now is the *menu selected* message, which has a value of *10*. When the user makes a selection from the menu, therefore, msg_buf[0] will contain *10*.

Since this is the only type of message we're waiting for, we don't even have to check msg_buf[0]. We can just assume that that's the message type we've received.

If this is all getting confusing, you might not be sure of the differences between an *event* and a *message*. They are *not* the same thing. A message is only received when a message event—one of six different events—is detected.

### Enough of this event junk.

Now that we have an idea of how evnt_multi() works, the program's workings are fairly simple. The object number of the chosen menu title is stored in msg_buf[3], and the object number of the chosen menu entry is stored in msg_buf[4]. Once we have that information, we set up nested switch statements to perform the appropriate actions.

The outer switch tests msg_buf[3] to see which menu title was selected. In the sample listing, you'll see that there's a case for each menu title. Within each menu title's case is a switch to test msg_buf[4], the object number of the selected menu entry. This is the best way to handle menu messages—the code looks very much like the menu bar it represents, and so is clear and easy to follow.

Now, let's look at this code in a little more detail, starting with DESK. Since we have only one action we have to take care of, we really didn't need the inner switch statement; we could have used an if, but I wanted to keep that "menu-looking" structure I mentioned before. Here, if the user clicks on the C-MANSHIP INFO. . . entry, we just call up an alert box. Simple. We don't have to handle any desk accessories the user might run; GEM does that for us—almost. I say almost, because, if a desk accessory is called up, it's up to your program to redraw the screen after the accessory has done its thing. Since we don't have anything special happening in the sample program, we can ignore this bothersome detail.

The file menu selections don't do anything fancy, just print a message to the screen. But look at the case statements. Where's QUIT? Wasn't that part of the file menu? Yes, indeed. And you'll find it at the very bottom of the sample listing. We're using it to break out of the do. . .while loop.

The options selections must handle the checkmarks that indicate active options. We use the call:

```
menu_icheck (menu_adr, OBJECT, flag);
```

Here, menu_adr is the address of the menu's object tree, OBJECT is the number of the entry you wish to check or uncheck, and flag is a Boolean value that indicates if a checkmark should be drawn (a nonzero value) or removed (a zero value).

In our menu_icheck() calls, we're not only taking care of the checkmarks, we're also handling the flags—reversing them within the function call with the statement below as the flag for the call:

```
op1=!op1
```

How does this work? First, the flag is changed to its opposite state with the NOT operation, then this new value is passed as the flag to the function. This is one of the advantages of C, being able to nest assignments and expressions.



In the selects menu section, if the ONOFF entry is selected, we must first change the text in ONOFF, then either enable or disable the other selections. To change the text of the entry, we use the call:

```
menu_text (menu_adr, OBJECT, s);
```

Here, menu_adr is the address of the menu bar's object tree, OBJECT is the entry's object number, and s is the address of the string you wish placed in the object. You should make sure that you've left room in the menu for the largest string you'll be using. Otherwise, you'll mess up the desktop. Also, you must make sure that the string is statically allocated; that is, it's global, not declared within a function. You never know when the user is going to activate that particular menu selection, and if your string isn't available globally, it may not be there when the menu is displayed. What a surprise that'll be.

Finally, we use the flag on to determine if we should enable or disable the rest of the menu entries. To perform this function, we use the call:

```
menu_ienable (menu_adr, OBJECT, flag);
```

Here, menu_adr is (you guessed it) the address of the menu tree, OBJECT is the object of the entry you wish to modify, and flag is a Boolean value that indicates if the entry is to be enabled (a nonzero value) or disabled (a zero value). In the sample program, we're using the same trick we used with menu_icheck() to handle the flag, but since we have three calls to menu_ienable(), we only reverse the flag in the first one.

When the user moves the mouse pointer over a menu selection, the title of that selection is highlighted, and the associated menu drops down. Once the user has clicked on an entry, the drop-down menu is removed, but the title remains highlighted. The highlighting reminds the user which menu selection is currently being processed. By leaving the title highlighted even after the user has made his selection, we can perform the actions required by the user's choice, then turn the highlighting off when we're ready. We turn off the highlighting with the call:

```
menu_tnormal (menu_adr, title, flag);
```

Here, menu_adr is the address of the menu tree, title is the object number of the menu title we wish to unhighlight, and flag is a Boolean value that indicates if the title is to be highlighted (a nonzero value) or unhighlighted (a zero value). The value for title will be the value found in msg_buf[3].

And, last but not least, once we're through with the menu bar, we must remove it from memory. We just use the menu_bar() call discussed previously to do this. Just change the flag to false.

**Another lesson learned.**

Your repertoire of GEM programming tricks is building fast. It won't be too long before you'll be ready to put together some professional looking pieces of software. Learning to handle menu bars is an important step in that direction. Without them, you can't really consider yourself a GEM programmer.

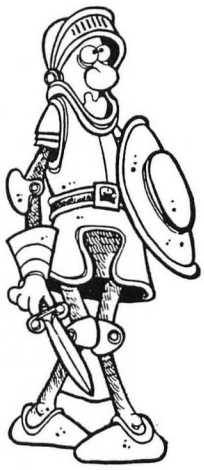
Of course, there's still a lot more we have to cover before we can consider GEM a challenge met. But we're getting there. //

Listing 1.
C listing.

```
/*************************************************/
/*            C-manship, Listing 1               */
/*                  ST-Log #15                   */
/*          Developed with Megamax C             */
/*************************************************/

#include "MENU.H"

#define MU_MESAG 0x0010
#define ARROW    0
#define R_TREE   0
#define TRUE     1
#define FALSE    0

/* The usual required GEM global arrays */
int work_in[11],
    work_out[57],
    pxyarray[10],
    contrl[12],
    intin[128],
    ptsin[128],
    intout[128],
    ptsout[128];

/* Global variables */
int handle, dum;
int msg_buf[8], op1, op2, op3, on;

char *alrt = "[1][C-manship, Issue 15|by Clayton Walnum][Okay]";
char *on_str = "     On     ";
char *off_str = "    Off     ";

main ()
{
   appl_init ();        /* Initialize application.        */
   open_vwork ();       /* Set up workstation.            */
   do_menu();           /* Go do the MENU.                */
   v_clsvwk (handle);   /* Close virtual workstation.     */
   appl_exit ();        /* Back to the desktop.           */
}

open_vwork ()
{
   int i;

   /* Get graphics handle, initialize the GEM arrays and open  */
   /* a virtual workstation.                                   */

   handle = graf_handle ( &dum, &dum, &dum, &dum);
   for ( i=0; i<10; work_in[i++] = 1 );
   work_in[10] = 2;
   v_opnvwk ( work_in, &handle, work_out );
}

do_menu ()
{
   long menu_adr; /* Address of the tree containing our menu.   */

   /* First, we initialize our option flags, so we can keep     */
   /* track of which ones are active.  Also, we change the      */
   /* mouse pointer to an arrow.                                */

   op1 = TRUE;
   op2 = FALSE;
   op3 = FALSE;
   on  = TRUE;

   graf_mouse ( ARROW, &dum );

   /* Here we load the resource file.  If the file is missing,  */
   /* we warn the user with an alert box then terminate the     */
   /* program by skipping the code following the ELSE.          */

   if ( ! rsrc_load ("\MENU.RSC") )
      form_alert ( 1, "[1][MENU.RSC missing!][Okay]" );
```

```
/* If the resource file loads OK, we get the address of the  */
/* tree, then handle menu messages from evnt_multi().        */

else {
   rsrc_gaddr ( R_TREE, TREE00, &menu_adr );
   menu_bar ( menu_adr, TRUE );
   do {
      evnt_multi ( MU_MESAG,0,0,0,0,0,0,0,0,0,0,0,0,0,0,msg_buf,
                   0,0,&dum,&dum,&dum,&dum,&dum,&dum );

      switch ( msg_buf[3] ) {

         case DESK:
            switch ( msg_buf[4] ) {
               case INFO:
                  form_alert ( 1, alrt );
                  break;
            }

         case FILE:
            switch ( msg_buf[4] ) {
               case LOAD:
                  v_gtext ( handle, 20, 120, "Load file " );
                  break;
               case SAVE:
                  v_gtext ( handle, 20, 120, "Save file " );
                  break;
            }

         case OPTIONS:
            switch ( msg_buf[4] ) {
               case OPTION1:
                  menu_icheck ( menu_adr, OPTION1, op1=!op1 );
                  break;
               case OPTION2:
                  menu_icheck ( menu_adr, OPTION2, op2=!op2 );
                  break;
               case OPTION3:
                  menu_icheck ( menu_adr, OPTION3, op3=!op3 );
                  break;
            }

         case SELECTS:
            switch ( msg_buf[4] ) {
               case ONOFF:
                  if ( on )
                     menu_text ( menu_adr, ONOFF, off_str );
                  else
                     menu_text ( menu_adr, ONOFF, on_str );
                  menu_ienable ( menu_adr, SELECT1, on=!on );
                  menu_ienable ( menu_adr, SELECT2, on );
                  menu_ienable ( menu_adr, SELECT3, on );
                  break;
               case SELECT1:
                  v_gtext ( handle, 20, 120, "Select 1 " );
                  break;
               case SELECT2:
                  v_gtext ( handle, 20, 120, "Select 2 " );
                  break;
               case SELECT3:
                  v_gtext ( handle, 20, 120, "Select 3 " );
                  break;
            }
         menu_tnormal ( menu_adr, msg_buf[3], TRUE );
      }
   }
   while ( msg_buf[4] != QUIT );
   menu_bar ( menu_adr, FALSE );
}
}
```

# Ian's Quest



**ST news, information and opinion.**

**by Ian Chadwick**

I just spent three days at the ECOO computer show—that's the Educational Computing Organization of Ontario. The show is intended to inform teachers (who call themselves educators) about computers (which they call learning tools) in the schools (educational institutions). Got that?

It's a lot like any other computer show except on a smaller scale, with a more definite focus than most. Atari was there, along with IBM, Apple and Commodore—the "name" companies. (Atari showed only STs, not XEs, which strikes me as rather limiting, but they seldom ask my advice in these matters anyway.)

There were also several book publishers present, who have a vested interest in the market—McGraw-Hill and Addison-Wesley, among others. However, the majority of the floor space was occupied by software distributors, such as Schoolhouse Computing Ltd. For the most part, these were distributors who exclusively target educational institutions (schools, see above. . .).

One particular afternoon, being nosey, I overheard Atari Canada's educational rep, Bruce Corbett, speaking with several prospective hardware buyers from local school boards. The teachers were usually quite keen on the ST—it shows well and the price is right. But somehow it was being passed over in favor of other systems. I listened more closely when Bruce started in with the real power behind the throne—the buyers—pointing out the advantages of the system.

He had a good line, one that should have connected well, but consumer response was frequently, "What sort of educational software do you have?"—and he could only point out a handful of "educational" programs on a wall. The buyers would then turn and look at the display of eight zillion such programs for Commodore and Apple across the aisle—many of which are tailored to the Canadian curricula. Then they'd mutter thanks and wander off. Some feigned polite interest in the hardware; some got genuinely excited when they saw a program like **Publishing Partner** pumping out high-quality newsletters on a laser printer. But then they'd consider the software availability and mosey on to the eight-zillion display.

Seeing this, I corralled a couple of reps of the larger software publishers one afternoon and asked them why they weren't producing for the ST.

"What's the installed base in the schools?" one asked me in return.

I didn't know. I asked him if that mattered particularly. After all, their products would help develop a base.

"We only produce software for an installed base," he replied. I pointed out that schools were more likely to buy a computer if there was software; thus, availability would *create* the installed base. He was unmoved: "We need a market for our product before we translate any of our existing products."

I shook my head and tried again. His stand made it unlikely for a new machine to get any support—until it was at a point where it didn't need their product, anyway. . .Catch 22.

"We only produce software for an installed base," he reiterated and went off to demonstrate a product to a passing educator. This is crazy, I thought. Schools won't buy until there's software; software publishers won't publish until the computers are in the schools. How can anyone win in this kind of situation? Boy, if the Wright brothers had had that point of view, they would have waited until they could fly over to the beach before test flying their plane!

Is there really such a dearth of ST educational software? Or was my perception based on just the temporary lack of display copies in the show? I walked back to one of the larger distributors who was also selling wares on the floor. Their display wall was littered with products. A sign announced products for Apple IIs of all sorts, Commodore 64s and 128s, Amigas, Macintoshes, IBM PCs and PCjrs, and Atari STs. A salesman approached me and asked if he could help.

"Yes, I'd like to buy several of your products," I answered and pointed out at random a dozen programs in the display—several of them by large publishers, and many well recognized titles. His eyes lit up and he began writing up a bill, listing every program in his neat, legible script. He tallied the total on his adding machine, figured in the tax and quoted me a price.

"You're lucky. We have all of these in stock. What computer system do you want them for?"

"The Atari ST, please."

His eyes narrowed and his brow furrowed. He turned to someone in the booth—probably his boss—and conferred in muted grunts. Then he came back to me, obviously distressed.

"We don't have any of these for the Atari ST, only for the Commodore, Apple or IBM. Don't you have one of those machines?"

"No. Your sign says Atari ST. I assumed that's what it meant. What do you have for the ST, then?" That seemed to catch him. He thought a moment and came up blank. He thrust a catalog into my hands and suggested that I look for myself, then wandered off, looking for more profitable customers. I walked away, but I looked back to see him shredding the invoice he had so carefully prepared.

That afternoon, I looked through the catalog—some 47 pages of products from, perhaps, 150 publishers. In that entire list, only 25 products were for the ST. Among these were **ChessMaster**, **Balance of Power**, **Word Writer**, **Personal Finance Pak**, **King's Quest**, some graphics, printing and games. None of these would I even casually classify as "educational." Only 4, from Coursemaker International, were readily recognizable as educational in the stricter sense of the word. Some, like **Winnie the Pooh**, are games with lip service paid to educational goals, but not much else.

Needless to say, I was disappointed. It seems we could and *should* do better. After all, isn't education a prime purpose for computers in the home, let alone in the schools? My 12-year-old daughter was coming to visit on the weekend, so I decided to try my local software outlet on Saturday and see what they could offer. Retail stores are often up on such things, and I assumed the clerk was over the trauma from my last visit.

"Hi," I said as I walked in, hoping to disarm him quickly. "I'm looking for some educational software for a bright 12-year-old. Any suggestions?"

He pointed me over to a wall of software and finished serving a customer. He had programs like **First Shapes** and **Speller Bee** from First Byte—both in the right vein, but too young for Jessica by about eight years. There were two programs from Unicorn called **Fraction Action** and **Decimal Dungeon** beside them. I shuddered. This sort of thing I'd encountered before—my bet was that she'd quickly learn to either associate decimals with monsters and mazes, or only be able to solve arithmetic in later years if she had a sword in one hand and could kill something later. On the other hand, these might be perfectly good programs, but I'm as prone to being affected by the title and the box art as anyone—and these suggested something less than what I'd had in mind.

This is a problem in a lot of "educational" software—trying to make a game out of a learning process, as if we're so dull and unimaginative that we need rewards and lots of laughs in order to master schoolwork. Not only does that insult the intelligence of the user, but it's not didactic—these programs entertain but don't teach in any sound, pedagogical manner. I recall a spelling program for another computer, a few years back, in which you had to send a diver down to the ocean floor to find the right letters to make up the word—fighting sharks, avoiding jellyfish, hunting through weeds and shipwrecks—one letter at a time. Fifteen minutes of scuba work and you found out you'd misspelled *cat*. Real educational—if you're Flipper.

**Math Word Problems** from Spinnaker seemed a more sensible approach to the topic, at least from the box description, but encompassed grades 7 to 12, and she's only in the 6th. Maybe that's equivalent to a higher grade in U.S. terms, but I wasn't sure enough to buy it. Besides, what a wonderful way to spend an afternoon: hi kid, here are some math problems for you!

But, on the other hand, if it was too dry, she wouldn't use it without supervision, and I wasn't prepared to do that, what with our once-in-a-blue-moon visit status. No, it had to be something light but useful, something she could amuse herself with for an hour or two and still gain some sort of knowledge from. I was musing on the questionable introduction of the game ethic into education when the clerk came over. I said I had already ruled out these few games and asked what else he had.

"Well, we have **Donald Duck's Playground** and **Winnie the Pooh**—they're both adventures. Wonderful fun, good graphics." I looked at the packages a bit dubiously. He ran **Winnie** on an ST and guided me through a few moves. Cute, but I failed to appreciate its purpose—beyond enriching A.A. Milne's and Walt Disney's already considerable estates.

"No, I think she's a bit too old for these. She's gone beyond Winnie the Pooh and is up to Phil Collins and Bon Jovi already. Besides, I want something educational, not just a game."

He looked miffed. "Adventures *are* educational. They help children build sentences and solve problems." He pushed a handful of Infocom games at me. "How about one of these?"

I tried to imagine my daughter typing *Take the knife and stab the dwarf. Get the money from under the table.* Education? That's a lot like using "Miami Vice" as a role model for social studies. "No thanks, I'm thinking of something a little less gamelike and more on the side of learning. Besides, I think these may be a tad too old for her." I could just see her, when she got home, telling her mother we had played **Leather Goddesses of Phobos** all afternoon.

"What about **Typing Tutor**? That's educational." He was right, and it certain-

ly teaches something, but I was pretty sure she got enough of that exercise in school, where they have other computers. It was the best bet, however, so I stored it away until later and asked for more. Obviously it was his last real offer, because he started roaming the stock and making silly suggestions.

"Is she old enough for **Balance of Power**? That's very educational." I had just reviewed the game and personally didn't feel she was ready for Crawford's somewhat lopsided view of politics and history. I was afraid that she'd want to go see *Rambo* afterward. I said no. "Is she good in history? We have a copy of **Great Battles** here."

I had spent several evenings that week playing the games in the Royal Software battles package, and I really liked the program—being a wargamer, it fits about all of my criteria for good wargaming. It's also a surprisingly good opponent, and, overall, the program is well designed and seems thoroughly researched—not an easy thing to accomplish. I like the game a lot (I'll review it real soon now. . .), but aside from having a copy, I don't think a 12-year-old girl would appreciate the intricacies of 19th-century warfare. I told him I'd pass on it.

"There's always **Time Bandits**."

"**Time Bandits**?" I asked incredulously. "What educational value does that have?"

"Well, it encourages problem solving, and it teaches hand-eye coordination." I sighed. Killing bizarre aliens and spiders, roaming about in a dungeon and playing **Pac-Man** is lots of fun, but hardly my idea of education. Maybe the clerk watched a lot of TV, so, for him, it *was* education. Maybe it was even reality for him. No, I said, I already have one, thanks. He rolled his eyes. "What does she take in school? Maybe that will help me."

What does one take in grade six? I can't recall. I went to grade six around the time of the last Ice Age. She'd only mentioned two things recently. "French," I said, "and clarinet." He closed his eyes and seemed to be uttering a silent prayer to whatever gods watch over clerks in computer stores. I noticed a new nervous tic in his facial muscles.

Well, after another half hour, I bought **Typing Tutor** and left the store less than fully satisfied. My daughter had to change her plans at the last moment, so I can't tell you now how it turned out—or whether or not it was worth the money.

I'm still upset by the lack of software in this field after almost two years, and I can't figure out who's at fault. Is it the publishers because they ignore Atari? Some are only now starting to recognize the machine. Will they get involved in educational software? No one is saying.

Or is it Atari's fault for either not encouraging them, or for a weak effort to break into the lucrative school market? I know that Atari Canada has targeted the schools strongly, and they're pushing in that realm (Bruce even has a nice, glossy newsletter now), but I don't know what the U.S. has been up to.

I expected to see a lot more software—language programs, for example, are pretty common. What about geography? Science? History? Surely it doesn't take a lot of programming to create a Trivial Pursuit type of game, one which focuses on educational aspects and has its questions ranked by grade levels.

The disappearance of the 8-bit line from the display suggests that it's no longer in the running—maybe they're targeting it as a game machine now. It has more educational software available than the ST. Both Commodore and Apple 8-bit systems are still selling in schools, so I don't know why Atari isn't continuing to push in that market. I'd sure like some answers, though.

Two final notes: I did get a letter back from subLOGIC about my request for information so users can build their own scenery disks for **FSII**. They said "thank you for your letter" and little else. They didn't answer any of my questions or my requests, nor make any comment on my suggestions. I was only surprised that they didn't send it addressed to "occupant."

Thanks where it's deserved: Marty Herzog, formerly of BI, is now at Atari Canada. He is kind enough to let me borrow or look at his new software, so I get to see it while it's still current. He keeps me informed on a lot of new products and developments. Without Marty and his coworker, Julius Oclamak, I'd see far fewer products than I do now. Thanks to both of you. //

---

*Ian Chadwick lives in Canada with his wife Susan and their numerous pets. He is currently working on a book about using GFA BASIC on the ST, a short story about explorer Henry Hudson and a BASIC game on the battle of Midway. In between hobbies, he works for a living.*

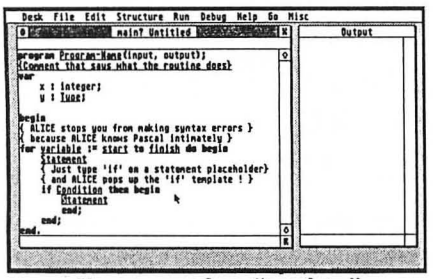
---

# We were going to tell you all about

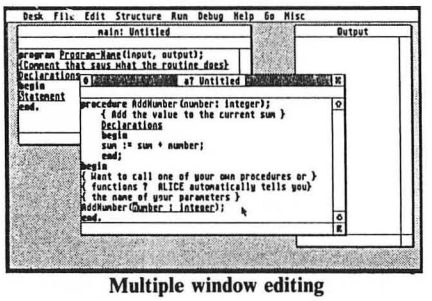
# ALICE

The Personal Pascal™

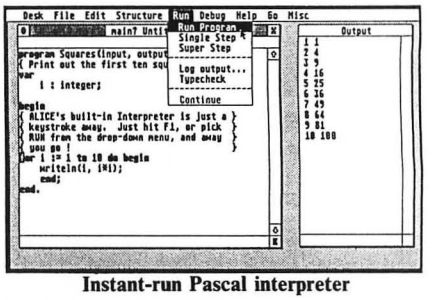
# but we'd rather show you . . .



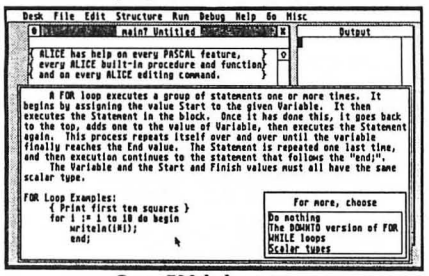
You program from "templates"



Multiple window editing



Instant-run Pascal interpreter



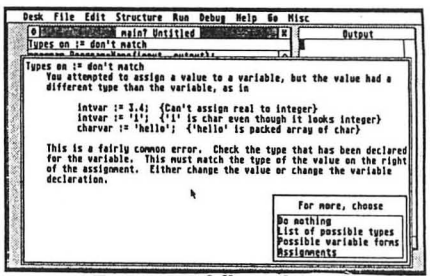
Over 700 help screens

## Programming Made Easy

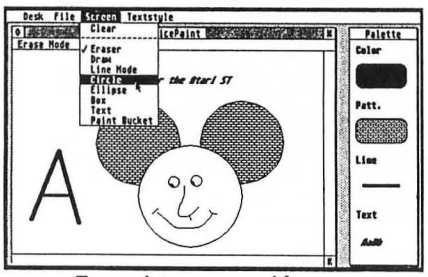
**If you're into programming, or if you *want* to be, you won't find a better system for your Atari ST than ALICE: The Personal Pascal. Already popular on the IBM-PC, ALICE now brings easy, interactive programming to the Atari ST.**

**ALICE is, without question, the best way around to learn about computers and programming. Beginners can even order our ALICE based textbook for only $19.95 with ALICE.**

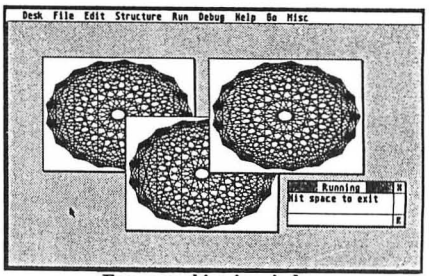
**Even if you're an expert, ALICE makes it easier to write, test and especially *debug* programs. ALICE's extensive Pascal improvements include a GEM interface even beginners can use and most of the extensions of Turbo Pascal. The surrounding examples just give a glimpse at how easy it is to put programs together using ALICE.**



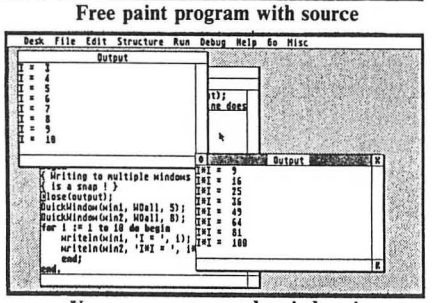
Every error fully explained



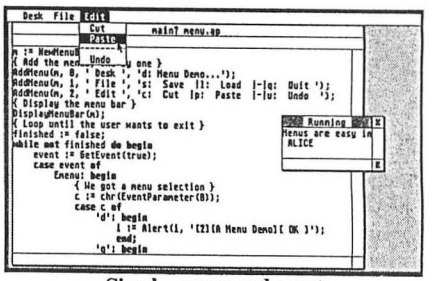
Free paint program with source



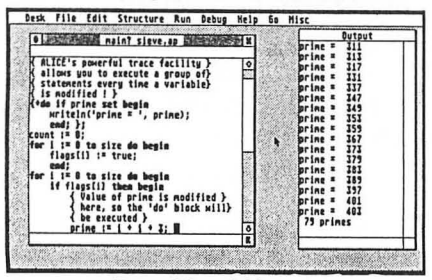
Easy graphics in windows



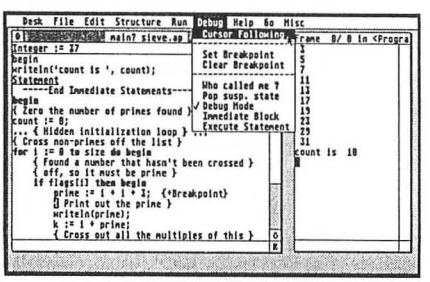
Your programs can do windows!



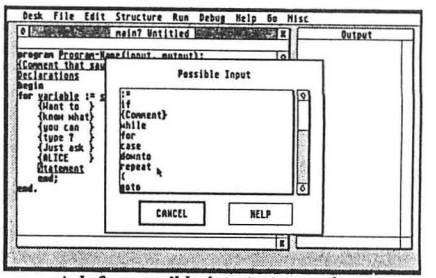
Simple menus and events



Powerful variable trace



Breakpoints, Single Step



Ask for possible input at any time

*"If I needed to learn Pascal all over again, or were going to teach a course in the language, I can't imagine using any program other than ALICE."*

— Adam Green, **Infoworld**

*"ALICE may be the most advanced programming environment currently available for the PC."*

— Michael Covington, **PC World**

*"If you enjoy programming languages, this comes pretty darn close to being as much fun as a video game."*

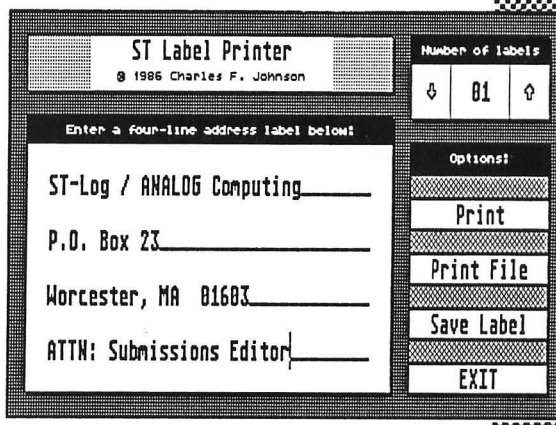
— Gene Wilburn, **Computing Canada**

Visit your dealer or send us $79.95 (USD) plus $5 for shipping and handling. Major credit card orders may phone collect. Dealer inquiries invited. IBM-PC version $95.

**Looking Glass Software**

**Looking Glass Software Limited**
124 King St. N. Waterloo, Ontario
N2J 2X8
519/884-7473

# ST Labelmaker




## A mailing-list/label-maker for your ST.

by Charles F. Johnson

**ST Labelmaker** is a GEM-based mailing-list/label-maker program that works on any ST with TOS in ROM (maybe TOS in RAM, too—never tried it!) and any printer. The program is designed to use 3½-by-$1\frac{5}{16}$-inch tractor-feed mailing labels, which should be available in most places. You may enter up to four lines of thirty-two characters for each address, and print multiple copies (up to ninety-nine) of a single label. In addition, the program lets you create disk-based mailing lists, and print them quickly and easily.

### How to use the program.

Type in the ST BASIC listings accompanying this article, and check your typing with **ST-Check** (see page 8). Running these BASIC programs creates two disk files called LABELMAK.PRG and LABELMAK.RSC, which are the program and resource file for **ST Labelmaker**. Be sure you're in medium or high resolution (the program doesn't work in low), and double-click on the file called LABELMAK.PRG.

A dialog box will appear on the screen with four lines in which to enter label data, a counter box to specify how many labels will be printed, and a box labeled *Options*. When you begin, the typing cursor is located on the first line of the label form; simply type in your label, using the cursor control keys to move from line to line (the RETURN key has no function in this program). If you make a mistake, use the BACKSPACE key to erase it, or ESCAPE to clear an entire line.

The Options box contains four buttons:

**Print.**

This button prints the label currently being displayed. You can print multiple copies of a single label, by using the counter box (marked *Number of labels*) in the upper right. Just click on the up or down arrows to raise or lower the value—single clicks advance by one; clicking and holding the button changes the value quickly. You can print up to ninety-nine copies of a single label at a time. Once the printout has started, you can abort it by pressing the right mouse button.

**Print File.**

This option prints a mailing list disk file, displaying each label on-screen as it is read from disk. A GEM "Item Selector" box will appear to let you choose the mailing list you want to print, displaying all files with an extension of *.LBL*. **ST Labelmaker** checks to make sure you're trying to print a valid label file, and won't let you print anything else. The "Number" setting has no effect when printing a file; each label is printed only once. To abort a printout, press the right mouse button just as above.

**Save Label.**

This saves the current label to a mailing list file. The GEM Item Selector box appears again; if you enter a filename that doesn't exist in the specified directory, a new label file will be created (valid label files have a header consisting of the word *LABELS* on a line by itself, followed by a blank line). If the file does exist (and it's a valid label file), the current label is appended to the end of it. Label files are ordinary ASCII text files, and can be loaded into a word processor for editing if desired. If you use **1st Word** to edit one of these mailing lists, remember to save it with word processor mode turned off.

**Exit.**

This, of course, exits back to the desktop.

### The wrap-up.

**ST Labelmaker** was written in assembly language, using the AS68 assembler from the Atari Developer's Kit. The source code is available on the disk version, or on the Atari

Users' Group on Delphi. It's a fairly good demonstration of how to use GEM to load a resource file, draw a dialog box, and get user input with assembly language.

This is a "bare-bones" type of program, without editing or database-style capabilities, but I've found it to be very useful. Its main advantages are its small size (less than 6K) and its GEM-based interface. In fact, I use it to maintain both my Christmas list and the mailing list for my band. Enjoy! //

---

*Charles F. Johnson is a professional musician and, now, a semi-professional computer programmer/reviewer/author. He lives in Los Angeles with his wife Patty and Spike, the world's most intelligent cat. Charles is a SYSOP on the ANALOG Publishing Atari SIG on Delphi; his user name is CFJ.*

---

**Listing 1.**
**ST BASIC listing.**

```
100 filename$="a:\labelmak.prg"
110 fullw 2:clearw 2:gotoxy 0,0:print
"creating file..."
120 option base 0
125 dim a%(16000):def seg=1:v$=""
130 p=varptr(a%(0)):bptr=p+1
140 for i%=1 to 4062
150 read v$:code%=val("&H"+v$)
160 poke p, code%:print ".";
170 p=p+1
180 next
190 bsave filename$,bptr,4062
200 print "file written":end
1000 data 60,1A,00,00,0B,D8,00,00,02,C
8,00,00,0A,6E,00,00
1010 data 00,00,00,00,00,00,00,00,00,0
0,00,00,2A,4F,2E,7C
1020 data 00,00,18,F6,2A,6D,00,04,20,2
D,00,0C,D0,AD,00,14
1030 data D0,AD,00,1C,D0,BC,00,00,01,0
0,2F,00,2F,0D,42,67
1040 data 3F,3C,00,4A,4E,41,DF,FC,00,0
0,00,0C,3F,3C,00,04
1050 data 4E,4E,54,8F,33,C0,00,00,0E,C
0,4A,40,66,00,00,12
1060 data 2A,7C,00,00,0B,E5,7A,01,61,0
0,08,72,60,00,06,70
1070 data 23,FC,00,00,0E,28,00,00,0E,1
0,23,FC,00,00,0B,D8
1080 data 00,00,11,60,61,00,0B,5E,4A,7
9,00,00,10,42,66,00
1090 data 00,12,2A,7C,00,00,0C,4D,7A,0
1,61,00,08,40,60,00
1100 data 06,3E,23,FC,00,00,0E,32,00,0
0,0E,10,42,B9,00,00
1110 data 0F,42,61,00,0B,30,23,F9,00,0
0,12,60,00,00,0E,A0
1120 data 28,79,00,00,0E,A0,7A,04,61,0
0,07,DE,1A,BC,00,BD
1130 data 28,79,00,00,0E,A0,7A,0C,61,0
0,07,30,7A,0D,61,00
1140 data 07,2A,7A,0E,61,00,07,24,7A,0
F,61,00,07,1E,2A,7C
1150 data 00,00,A0,00,28,7C,00,00,0D,7
4,7A,0C,10,15,B0,14
1160 data 67,00,00,06,52,8D,60,F4,20,4
D,B9,0D,66,E6,51,CD
1170 data FF,FA,20,08,23,C8,00,00,0E,B
4,22,08,08,81,00,00
1180 data 20,41,55,88,22,10,B0,81,66,F
8,23,C8,00,00,0E,B0
1190 data 00,68,00,01,FF,FE,33,E8,00,0
8,00,00,0E,C6,3F,3C
1200 data 00,19,4E,41,54,8F,D0,3C,00,4
1,13,C0,00,00,0D,C8
1210 data 42,67,2F,3C,00,00,13,DA,3F,3
C,00,47,4E,41,50,8F
1220 data 2A,7C,00,00,0D,CA,28,7C,00,0
0,13,DA,7A,3F,4A,14
1230 data 67,00,00,08,1A,DC,51,CD,FF,F
6,28,7C,00,00,0E,08
1240 data 7A,06,1A,DC,51,CD,FF,FC,24,7
C,00,00,13,8A,72,0F
1250 data 42,1A,51,C9,FF,FC,61,00,07,3
0,23,F9,00,00,0E,A0
1260 data 00,00,0E,A4,61,00,07,64,33,F
C,00,0C,00,00,0E,D2
1270 data 61,00,07,FC,02,79,7F,FF,00,0
0,0E,E2,0C,79,00,12
1280 data 00,00,0E,E2,67,00,00,9A,0C,7
9,00,13,00,00,0E,E2
1290 data 67,00,01,5E,0C,79,00,14,00,0
0,0E,E2,67,00,03,8A
1300 data 0C,79,00,15,00,00,0E,E2,67,0
0,04,BC,7A,08,28,79
1310 data 00,00,0E,A0,61,00,06,B2,7A,0
1,DB,C5,0C,79,00,09
1320 data 00,00,0E,E2,66,00,00,3E,0C,1
5,00,39,66,00,00,0A
1330 data 1A,BC,00,30,60,00,00,08,52,1
5,60,00,00,08,53,8D
1340 data 51,CD,FF,E6,33,FC,00,08,00,0
0,0E,E6,33,FC,00,01
1350 data 00,00,0E,DE,33,FC,00,02,00,0
0,0E,E0,61,00,06,B8
1360 data 60,00,FF,66,0C,15,00,30,66,0
0,00,0A,1A,BC,00,39
1370 data 60,00,00,06,53,15,60,CC,53,8
D,51,CD,FF,E8,60,C4
1380 data 28,79,00,00,0E,A0,7A,08,61,0
0,06,3E,61,00,05,74
1390 data 4A,80,67,00,00,92,33,C0,00,0
0,0E,C8,61,00,06,3E
1400 data 61,00,04,88,66,00,00,34,61,0
0,07,52,61,00,06,3A
1410 data 61,00,04,84,67,00,00,10,42,7
9,00,00,0E,E4,61,00
1420 data 08,66,60,00,FE,F6,23,F9,00,0
0,0E,A0,00,00,0E,A4
1430 data 61,00,06,58,61,00,06,06,60,C
6,33,FC,00,03,00,00
1440 data 0E,C4,33,FC,00,0C,00,00,0E,C
A,61,00,04,16,66,00
1450 data 00,36,42,85,3A,39,00,00,0E,C
A,61,00,04,BC,6B,00
1460 data 00,26,52,79,00,00,0E,CA,3A,3
9,00,00,0E,CA,BA,7C
1470 data 00,10,66,D6,61,00,05,1C,53,7
9,00,00,0E,C8,4A,79
1480 data 00,00,0E,C8,66,B4,42,79,00,0
0,0E,E4,61,00,07,F8
1490 data 33,FC,00,12,00,00,0E,E6,42,7
9,00,00,0E,DE,42,79
1500 data 00,00,0E,E0,61,00,05,D0,61,0
0,05,9E,60,00,FE,7A
1510 data 61,00,06,AA,28,79,00,00,0E,B
0,28,BC,00,00,0D,82
1520 data 39,7C,00,F8,00,08,20,7C,00,0
0,0D,C8,61,00,08,18
1530 data 4A,39,00,00,13,8A,66,00,00,1
0,42,79,00,00,0E,E4
1540 data 61,00,07,A4,60,00,FE,34,0C,7
9,00,01,00,00,10,44
1550 data 67,00,00,04,60,E4,61,00,04,B
8,67,00,00,08,61,00
1560 data 03,AC,60,D6,23,F9,00,00,14,3
4,00,00,0E,B8,2A,7C
1570 data 00,00,13,8A,42,45,61,00,04,B
A,6B,BE,33,C0,00,00
1580 data 0E,C2,2A,7C,00,00,13,80,7A,0
A,61,00,04,B6,6B,00
```

```
1590 data 02,E0,2A,7C,00,00,0D,BE,28,7
C,00,00,13,80,7A,09
1600 data BB,0C,67,00,00,0A,61,00,03,9
A,60,00,02,C4,51,CD
1610 data FF,F0,61,00,03,36,66,00,00,0
C,61,00,03,3A,67,F2
1620 data 60,00,02,AE,23,F9,00,00,0E,A
0,00,00,0E,A4,61,00
1630 data 05,1A,61,00,04,C8,42,79,00,0
0,0E,D0,23,FC,00,00
1640 data 00,0A,00,00,0E,BC,33,FC,00,0
C,00,00,0E,CA,28,79
1650 data 00,00,0E,A0,42,85,3A,39,00,0
0,0E,CA,61,00,04,8A
1660 data 33,FC,00,20,00,00,0E,CC,7A,0
1,61,00,04,36,6B,00
1670 data 01,0A,61,00,02,AE,67,00,00,0
A,33,FC,00,01,00,00
1680 data 0E,D0,52,B9,00,00,0E,BC,1A,1
5,BA,3C,00,0D,67,00
1690 data 00,28,52,8D,53,79,00,00,0E,C
C,66,CC,42,15,2A,7C
1700 data 00,00,13,80,7A,01,61,00,03,F
A,6B,00,00,CE,52,B9
1710 data 00,00,0E,BC,60,00,00,04,42,1
5,2A,7C,00,00,13,80
1720 data 7A,01,61,00,03,DE,6B,00,00,B
2,52,B9,00,00,0E,BC
1730 data 33,F9,00,00,0E,CA,00,00,0E,E
6,42,79,00,00,0E,DE
1740 data 42,79,00,00,0E,E0,61,00,04,4
E,61,00,02,36,67,00
1750 data 00,0A,33,FC,00,01,00,00,0E,D
0,52,79,00,00,0E,CA
1760 data 3A,39,00,00,0E,CA,BA,7C,00,1
0,66,00,FF,42,4A,79
1770 data 00,00,0E,D0,66,00,00,64,2A,7
C,00,00,13,80,7A,02
1780 data 61,00,03,80,6B,00,00,54,54,B
9,00,00,0E,BC,33,FC
1790 data 00,03,00,00,0E,C4,33,FC,00,0
C,00,00,0E,CA,61,00
1800 data 01,E2,66,00,00,36,42,85,3A,3
9,00,00,0E,CA,61,00
1810 data 02,88,52,79,00,00,0E,CA,3A,3
9,00,00,0E,CA,BA,7C
1820 data 00,10,66,DA,61,00,02,EC,20,3
9,00,00,0E,BC,22,39
1830 data 00,00,0E,B8,B0,81,6D,00,FE,C
E,61,00,03,5A,61,00
1840 data 03,88,42,79,00,00,0E,E4,61,0
0,05,BC,33,FC,00,13
1850 data 00,00,0E,E6,42,79,00,00,0E,D
E,42,79,00,00,0E,E0
1860 data 61,00,03,94,60,00,FC,42,61,0
0,04,72,28,79,00,00
1870 data 0E,B0,28,BC,00,00,0D,A2,39,7
C,00,D8,00,08,20,7C
1880 data 00,00,0D,C8,61,00,05,E0,4A,3
9,00,00,13,8A,66,00
1890 data 00,10,42,79,00,00,0E,E4,61,0
0,05,6C,60,00,FB,FC
1900 data 0C,79,00,01,00,00,10,44,67,0
0,00,04,60,E4,61,00
1910 data 02,80,67,00,00,32,42,67,48,7
9,00,00,13,8A,3F,3C
1920 data 00,3C,4E,41,50,8F,4A,40,6B,C
8,33,C0,00,00,0E,C2
1930 data 2A,7C,00,00,0D,BE,2A,3C,00,0
0,00,0A,61,00,02,9E
1940 data 6B,B0,61,00,02,B2,2A,7C,00,0
0,13,8A,7A,02,61,00
1950 data 02,62,6B,00,FB,A6,33,C0,00,0
0,0E,C2,61,00,02,BE
1960 data 2A,7C,00,00,13,80,7A,0A,61,0
0,02,58,6B,00,00,82
1970 data 2A,7C,00,00,0D,BE,28,7C,00,0
0,13,80,7A,09,BB,0C
1980 data 67,00,00,0A,61,00,01,3C,60,0
0,00,66,51,CD,FF,F0
1990 data 3F,3C,00,02,3F,39,00,00,0E,C
2,2F,3C,00,00,00,00
2000 data 3F,3C,00,42,4E,41,DF,FC,00,0
0,00,0A,4A,40,6B,00
2010 data 00,40,33,F9,00,00,0E,C2,00,0
0,0E,C4,33,FC,00,0C
2020 data 00,00,0E,CA,3A,39,00,00,0E,C
A,61,00,01,2C,6B,00
2030 data 00,20,52,79,00,00,0E,CA,3A,3
9,00,00,0E,CA,BA,7C
2040 data 00,10,66,E0,2A,7C,00,00,0D,C
6,7A,02,61,00,01,EE
2050 data 61,00,02,04,42,79,00,00,0E,E
4,61,00,04,6A,61,00
2060 data 02,28,60,00,FA,F6,61,00,03,3
4,33,FC,00,02,00,00
2070 data 0E,D4,61,00,03,34,28,79,00,0
0,0E,B0,28,B9,00,00
2080 data 0E,B4,02,6C,00,FE,FF,FE,39,7
9,00,00,0E,C6,00,08
2090 data 23,FC,00,00,0E,3C,00,00,0E,1
0,61,00,05,08,42,67
2100 data 4E,41,42,79,00,00,0E,CE,08,3
8,00,01,27,DE,67,00
2110 data 00,12,08,38,00,01,27,DE,66,F
8,33,FC,00,01,00,00
2120 data 0E,CE,4A,79,00,00,0E,CE,4E,7
5,3F,3C,00,11,4E,41
2130 data 54,8F,4A,40,4E,75,2A,7C,00,0
0,0C,8E,7A,01,61,00
2140 data 01,BC,0C,79,00,01,00,00,10,4
2,4E,75,2A,7C,00,00
2150 data 0D,10,7A,0B,1A,FC,00,20,51,C
D,FF,FA,2A,7C,00,00
2160 data 0D,10,28,7C,00,00,13,8A,7A,0
B,4A,14,67,00,00,08
2170 data 1A,DC,51,CD,FF,F6,2A,7C,00,0
0,0C,FC,7A,01,60,00
2180 data 01,7C,2A,7C,00,00,0D,34,7A,0
B,1A,FC,00,20,51,CD
2190 data FF,FA,2A,7C,00,00,0D,34,28,7
C,00,00,13,8A,7A,0B
2200 data 4A,14,67,00,00,08,1A,DC,51,C
D,FF,F6,2A,7C,00,00
2210 data 0D,2A,7A,01,60,00,01,46,28,7
9,00,00,0E,A0,61,00
2220 data 01,08,42,84,28,7C,00,00,0E,E
A,7A,28,4A,15,67,00
2230 data 00,0A,18,DD,52,04,57,CD,FF,F
4,18,FC,00,0D,18,BC
2240 data 00,0A,54,04,2F,3C,00,00,0E,E
A,2F,04,3F,39,00,00
2250 data 0E,C4,3F,3C,00,40,4E,41,DF,F
C,00,00,00,0C,4A,80
2260 data 4E,75,42,80,0C,15,00,39,62,0
0,00,1E,0C,15,00,30
2270 data 65,00,00,16,E3,88,22,00,E5,8
8,D0,81,12,1D,02,81
2280 data 00,00,00,0F,D0,81,60,DC,4E,7
5,61,00,00,9C,42,15
2290 data 4E,75,2F,3C,00,00,0D,70,2F,3
C,00,00,00,04,3F,3C
2300 data 00,03,3F,3C,00,40,4E,41,DF,F
C,00,00,00,0C,4E,75
2310 data 48,79,00,00,00,14,1A,3F,3C,00,1
A,4E,41,5C,8F,42,67
2320 data 2F,3C,00,00,13,8A,3F,3C,00,4
E,4E,41,50,8F,4A,40
2330 data 4E,75,3F,05,2F,0D,3F,3C,00,3
D,4E,41,50,8F,4A,40
2340 data 4E,75,2F,0D,2F,05,3F,39,00,0
0,0E,C2,3F,3C,00,3F
2350 data 4E,41,DF,FC,00,00,00,0C,4A,8
0,4E,75,2F,0D,2F,05
2360 data 3F,39,00,00,0E,C2,3F,3C,00,4
0,4E,41,DF,FC,00,00
```

```
2370 data 00,0C,4A,80,4E,75,3F,39,00,0
0,0E,C2,3F,3C,00,3E
2380 data 4E,41,58,8F,4A,40,4E,75,2A,4
C,CA,FC,00,18,DA,BC
2390 data 00,00,00,0C,DB,C5,2A,55,2A,5
5,4E,75,33,FC,00,02
2400 data 00,00,0F,42,60,00,00,08,42,7
9,00,00,0F,42,23,FC
2410 data 00,00,0E,46,00,00,0E,10,60,0
0,03,0A,23,CD,00,00
2420 data 11,60,33,C5,00,00,0F,42,23,F
C,00,00,0E,6E,00,00
2430 data 0E,10,60,00,02,F0,42,79,00,0
0,0E,E8,23,F9,00,00
2440 data 0E,A0,00,00,0E,A8,60,00,01,5
8,23,FC,00,00,0E,78
2450 data 00,00,0E,10,23,F9,00,00,0E,A
4,00,00,11,60,61,00
2460 data 02,C4,33,F9,00,00,10,44,00,0
0,0E,D6,33,F9,00,00
2470 data 10,46,00,00,0E,D8,33,F9,00,0
0,10,48,00,00,0E,DA
2480 data 33,F9,00,00,10,4A,00,00,0E,D
C,42,79,00,00,0E,D4
2490 data 61,00,00,96,33,FC,00,01,00,0
0,0E,D4,61,00,00,8A
2500 data 23,FC,00,00,0E,50,00,00,0E,1
0,42,79,00,00,0F,42
2510 data 33,FC,00,03,00,00,0F,44,33,F
9,00,00,0E,D6,00,00
2520 data 0F,46,33,F9,00,00,0E,D8,00,0
0,0F,48,33,F9,00,00
2530 data 0E,DA,00,00,0F,4A,33,F9,00,0
0,0E,DC,00,00,0F,4C
2540 data 23,F9,00,00,0E,A4,00,00,11,6
0,60,00,02,38,23,FC
2550 data 00,00,0E,82,00,00,0E,10,33,F
9,00,00,0E,D2,00,00
2560 data 0F,42,23,F9,00,00,0E,A4,00,0
0,11,60,61,00,02,16
2570 data 33,F9,00,00,10,42,00,00,0E,E
2,4E,75,33,FC,00,03
2580 data 00,00,0E,D4,60,00,00,02,23,F
C,00,00,0E,8C,00,00
2590 data 0E,10,33,F9,00,00,0E,D4,00,0
0,0F,42,33,FC,01,40
2600 data 00,00,0F,44,0C,79,00,02,00,0
0,0E,C0,66,00,00,0E
2610 data 33,FC,00,C8,00,00,0F,46,60,0
0,00,0A,33,FC,00,64
2620 data 00,00,0F,46,33,FC,00,08,00,0
0,0F,48,33,FC,00,08
2630 data 00,00,0F,4A,33,F9,00,00,0E,D
6,00,00,0F,4C,33,F9
2640 data 00,00,0E,D8,00,00,0F,4E,33,F
9,00,00,0E,DA,00,00
2650 data 0F,50,33,F9,00,00,0E,DC,00,0
0,0F,52,60,00,01,86
2660 data 23,FC,00,00,0E,5A,00,00,0E,1
0,33,F9,00,00,0E,E6
2670 data 00,00,0F,42,23,F9,00,00,0E,A
8,00,00,11,60,61,00
2680 data 01,64,3A,39,00,00,10,44,DA,7
9,00,00,0E,DE,33,C5
2690 data 00,00,0F,46,3A,39,00,00,10,4
6,DA,79,00,00,0E,DE
2700 data 33,C5,00,00,0F,48,2A,79,00,0
0,0E,A8,42,85,3A,39
2710 data 00,00,0E,E6,CA,FC,00,18,DB,C
5,3A,2D,00,14,9A,79
2720 data 00,00,0E,E0,33,C5,00,00,0F,4
A,3A,2D,00,16,9A,79
2730 data 00,00,0E,E0,33,C5,00,00,0F,4
C,23,FC,00,00,0E,50
2740 data 00,00,0E,10,33,F9,00,00,0E,E
6,00,00,0F,42,33,F9
2750 data 00,00,0E,E8,00,00,0F,44,23,F
9,00,00,0E,A8,00,00
2760 data 11,60,60,00,00,E0,23,FC,00,0
0,0E,64,00,00,0E,10
2770 data 33,F9,00,00,0E,E2,00,00,0F,4
2,42,79,00,00,0F,44
2780 data 33,F9,00,00,0E,D6,00,00,0F,4
6,33,F9,00,00,0E,D8
2790 data 00,00,0F,48,33,F9,00,00,0E,D
A,00,00,0F,4A,33,F9
2800 data 00,00,0E,DC,00,00,0F,4C,33,F
9,00,00,0E,E4,00,00
2810 data 0F,4E,42,79,00,00,0F,50,23,F
9,00,00,0E,A4,00,00
2820 data 11,60,60,00,00,80,23,C8,00,0
0,0E,AC,23,FC,00,00
2830 data 0E,96,00,00,0E,10,23,F9,00,0
0,0E,AC,00,00,11,60
2840 data 23,FC,00,00,13,8A,00,00,11,6
4,61,00,00,58,42,80
2850 data 20,79,00,00,0E,AC,10,10,90,3
C,00,41,3F,00,3F,3C
2860 data 00,0E,4E,41,58,8F,20,79,00,0
0,0E,AC,54,88,22,7C
2870 data 00,00,13,9A,7A,3F,4A,10,67,0
0,00,08,12,D8,51,CD
2880 data FF,F6,7A,3F,0C,21,00,5C,67,0
0,00,06,51,CD,FF,F6
2890 data 52,89,42,11,48,79,00,00,13,9
A,3F,3C,00,3B,4E,41
2900 data 5C,8F,4E,75,22,3C,00,00,0E,1
0,20,3C,00,00,00,C8
2910 data 4E,42,4E,75,4C,41,42,45,4C,4
D,41,4B,2E,52,53,43
2920 data 00,5B,33,5D,5B,20,7C,20,54,6
8,69,73,20,70,72,6F
2930 data 67,72,61,6D,20,6D,75,73,74,2
0,62,65,20,73,65,65
2940 data 6E,20,69,6E,20,7C,20,6D,65,6
4,69,75,6D,20,6F,72
2950 data 20,68,69,67,68,20,72,65,73,6
F,6C,75,74,69,6F,6E
2960 data 20,74,6F,20,7C,20,62,65,20,6
6,75,6C,6C,79,20,61
2970 data 70,70,72,65,63,69,61,74,65,6
4,21,20,7C,20,5D,5B
2980 data 20,53,6F,72,72,79,21,20,5D,5
B,33,5D,5B,20,43,61
2990 data 6E,27,74,20,66,69,6E,64,20,7
2,65,73,6F,75,72,63
3000 data 65,20,66,69,6C,65,3A,20,7C,2
0,7C,20,20,20,20,20
3010 data 4C,41,42,45,4C,4D,41,4B,2E,5
2,53,43,20,7C,20,5D
3020 data 5B,20,53,6F,72,72,79,21,20,5
D,5B,33,5D,5B,20,50
3030 data 72,69,6E,74,65,72,20,69,73,2
0,6E,6F,74,20,6F,6E
3040 data 2D,6C,69,6E,65,21,20,7C,20,7
C,20,4D,61,6B,65,20
3050 data 73,75,72,65,20,70,6F,77,65,7
2,20,69,73,20,6F,6E
3060 data 20,61,6E,64,20,7C,20,63,6F,6
E,6E,65,63,74,69,6F
3070 data 6E,73,20,61,72,65,20,73,65,6
3,75,72,65,2E,2E,2E
3080 data 20,7C,20,5D,5B,20,43,6F,6E,7
4,69,6E,75,65,20,7C
3090 data 20,41,62,6F,72,74,20,5D,5B,3
3,5D,5B,20,43,61,6E
3100 data 27,74,20,66,69,6E,64,20,7C,2
0,7C,20,20,20,20,20
3110 data 20,20,20,20,20,20,20,20,20,7
C,20,5D,5B,20,43,61
3120 data 6E,63,65,6C,20,5D,5B,33,5D,5
B,20,20,20,20,20,20
3130 data 20,20,20,20,20,20,20,20,20,2
0,20,20,20,20,20,20
3140 data 7C,20,7C,20,69,73,20,6E,6F,7
4,20,61,20,76,61,6C
```

# YOUR ATARI 520ST SUPPORT CENTER

| | |
|---|---|
| Atari 520 ST Mono System | Call |
| Atari 520 ST Color System | For |
| Atari SF 314 Disk Drive | Best |
| Atari SH 204 Hard Drive | Price |
| Atari Hardware | |

| | |
|---|---|
| **ABACUS** | |
| Paint Pro | 33 |
| Text Pro | 33 |
| Data Trieve | 33 |
| Powerplan | 54 |
| PC Board Designer | 139 |
| Assem. Pro | 43 |
| Paint Pro Data Lib. | 21 |
| Forth M/T | 33 |
| **ACADEMY** | |
| Typing Tutor/ Word Invaders | 23 |
| **ACCESS** | |
| Leaderboard Golf | 26 |
| Tournament Disk | 14 |
| Tenth Frame | 26 |
| **ACCOLADE** | |
| Mean 18 Golf | 29 |
| Sundog | 26 |
| Mean 18 Course Dsk | 14 |
| **ACTION** | |
| Action Pak | 28 |
| Solapak | 28 |
| **ACTIVISION** | |
| Mindshadow | 33 |
| Borrowed Time | 33 |
| Hacker | 29 |
| Music Studio | 39 |
| Paintworks | 47 |
| Little Comp. People | 33 |
| Hacker II | 33 |
| Tass Times/Tonetown | 29 |
| Champ. Basketball | 29 |
| Shanghai | 29 |
| Champ. Baseball | 27 |
| Champ. Football | 29 |
| **ANTIC** | |
| C.O.L.R. Obj. Editor | 23 |
| Maps & Legends | 27 |
| Macro Assembler | 60 |
| Lattice C | 114 |
| Disk Doctor | 23 |
| A-Calc | 45 |
| Meta Pascal | 75 |
| CAD-3D | 37 |
| A-Ram | 15 |
| A-Seka | 27 |
| GST C Compiler | 60 |
| GST-ASM | 45 |
| Expert Opinion | 75 |
| Flash | 28 |
| Star Struck | 19 |
| Red Alert | 19 |
| Kermit & Remote Co. | 19 |
| Crystal | 19 |
| Fonts, Primitives | 19 |
| Cartographer | 27 |
| G.I.S.T. | 27 |
| K-Switch | 27 |
| K-Resource | 38 |
| Rambrandt | 15 |
| A-Chart | 30 |
| **ARTWORX** | |
| Bridge 4.0 | 19 |
| Compubridge | 19 |
| Maillist | 14 |
| Strip Poker | 25 |
| Peggammon | 12 |
| Hole in One Golf | 19 |
| **ATARI** | |
| DB Man | 99 |
| DB Master | 33 |
| Joust | 19 |
| Star Raiders | 19 |
| Crystal Castles | 19 |
| **BECKEMEYER** | |
| Micro C-Shell | 39 |
| Micro C-Tools | 19 |
| Micro Make | 27 |
| RTX | 53 |
| Micro MTC Shell | 99 |
| Hard Disk Accelerate | 30 |
| Ansiterm | 19 |
| Hard Disk Tool Kit | 23 |
| **CENTRAL POINT** | |
| Copy II | 26 |
| **DAC** | |
| Easy Accounting | 47 |
| Easy Payroll | 33 |
| **DATAPACIFIC** | |
| Magic Sac | 119 |
| **DATASOFT SOFTWARE** | |
| Mercenary | 25 |
| Alternate Reality | 23 |
| **EIDERSOFT** | |
| ST Protector | 22 |
| ST Karate | 22 |
| **ELECTRONIC ARTS** | |
| Financial Cookbook | 34 |
| Ultima III | 39 |
| Coloring Book | 15 |
| Golden Oldies | 23 |
| Arctic Fox | 29 |
| Skyfox | 33 |
| Starfleet I | 32 |
| Ogre | 29 |
| Quizam | 25 |
| Chessmaster 2000 | 33 |
| **EPYX** | |
| Winter Games | 26 |
| Rogue | 26 |
| Temple of Apshai | 26 |
| World Games | 26 |
| Champion. Wrestling | 26 |
| Super Cycle | 26 |
| **FIREBIRD** | |
| The Pawn | 29 |
| Starglider | 29 |
| **FIRST BYTE** | |
| Kid Talk | 32 |
| Speller Bee | 32 |
| Math Talk | 32 |
| First Shapes | 32 |
| **HYBRID ARTS** | |
| EZ Track | 39 |
| CZ Android | 59 |
| **LOGICAL DESIGN WORKS** | |
| LDW Basic Compiler | 47 |
| **MARKSMAN TECHNOLOGY** | |
| Phaser | 69 |
| **MASTERTRONICS** | |
| Renegade | 14 |
| **MEGASOFT** | |
| A Copier | 27 |
| X-Rated Library | 17 |
| OTHER: | |
| Certificate Maker | 33 |
| Psion Chess | 39 |
| Microleague Baseball | 39 |
| Aegis Animator | 54 |
| **MICHTRON** | |
| DOS Shell | 26 |
| The Animator | 26 |
| Personal Money Mgr | 33 |
| Utilities | 39 |
| M-Disk | 26 |
| Softspool | 26 |
| Flip Side | 26 |
| Calendar | 19 |
| MI-Term | 33 |
| Time Bandits | 26 |
| Cornerman | 33 |
| Mighty Mail | 33 |
| Cards | 26 |
| Business Tools | 33 |
| Major Motion | 26 |
| MI-Dupe | 20 |
| BBS 2.0 | 54 |
| Pro Football Wizard | 27 |
| Pinball Factory | 27 |
| Your Finance Future | 26 |
| MI-Print | 20 |
| Trivia Challenge | 27 |
| GFA Basic Compiler | 54 |
| Trim Base | 69 |
| Match Point | 27 |
| Raid | 27 |
| Super Directory | 27 |
| GFA Basic. Interpret. | 54 |
| Tech Mate | 26 |
| Perfect Match | 26 |
| Space Shuttle II | 26 |
| **MICRODEAL** | |
| Karate Kid II | 29 |
| **MICROLEAGUE** | |
| Baseball | 39 |
| **MICROPROSE** | |
| Silent Service | 26 |
| **MI-GRAPH** | |
| Easy Draw | 52 |
| Fast | 33 |
| **MILES** | |
| Harrier Strike | 33 |
| **MINDSCAPE** | |
| Brataccus | 33 |
| S.D.I. | 33 |
| Balance of Power | 33 |
| **NAVARONNE** | |
| Astrology | 29 |
| Timekeeper | 34 |
| **OSS** | |
| Personal Pascal | 50 |
| **PROCO** | |
| Pro-Copy | 24 |
| **PROGRESSIVE COMP. APPL.** | |
| Graphic Artist | 132 |
| Font Editor | 54 |
| Font Pak I | 34 |
| Desktop Publ. Lib. | 34 |
| **PROGRESSIVE PERIPHERALS** | |
| Picasso's Revenge | 40 |
| **PSION** | |
| Chess | 39 |
| **PSYGNOSIS** | |
| Deep Space | 34 |
| Arena | 26 |
| **QUANTUM** | |
| ST-Talk | 14 |
| **QUICKVIEW** | |
| Zoomracks II | 99 |
| **REGENT** | |
| Regent Spell | 34 |
| Regent Base | 67 |
| Regent Word II | 67 |
| **SHANNER** | |
| Soundwave SW-1 | 15 |
| ST-Key | 10 |
| Macro-Manager | 15 |
| Macro-Desk | 10 |
| DB Calc II | 15 |
| Colorwriter | 10 |
| **SHELBOURNE** | |
| Pool | 23 |
| Shuffleboard | 20 |
| **SIERRA** | |
| Kings Quest II | 33 |
| Ultima II | 20 |
| Winnie the Pooh | 17 |
| Black Cauldron | 26 |
| K/Q Hint Book | 6 |
| B/C Hint Book | 6 |
| Donald Duck | 17 |
| Cash Disbursements | 67 |
| Space Quest | 33 |
| Kings Quest III | 33 |
| Kings Quest I | 33 |
| **SOFTLOGIK** | |
| Publish. Partner | 99 |
| **SPECTRUM HOLOBYTE** | |
| Gato | 27 |
| **SPINNAKER** | |
| Homework Helper:Math | 34 |
| Treasure Island | 27 |
| **SPRINGBOARD** | |
| Certificate Maker | Call |
| **SSI** | |
| Phantasie | 26 |
| Phantasie II | 26 |
| **SUBLOGIC** | |
| Flight Simulator | 35 |
| **TDI** | |
| Modula-2 | 75 |
| Modula-2 Develp.Kit | 99 |
| UCSD Pascal | 59 |
| **TIMEWORKS** | |
| Datamanager | 53 |
| Swiftcalc | 53 |
| Wordwriter | 53 |
| **UNISON WORLD** | |
| Printmaster | 26 |
| Art Gallery I | 19 |
| Art Gallery II | 19 |
| **VIP** | |
| VIP Professional | 149 |
| VIP Lite | 69 |
| **X-LENT** | |
| Typesetter Elite | 29 |
| Rubber Stamp | 25 |
| Music Box | 32 |
| Megafont | 25 |
| Write 90 Degrees | 19 |
| PM Interface | 19 |
| Typeset Color Icon | 13 |

**PRINTERS**

| | |
|---|---|
| **STAR MICRONICS** | |
| NP-10 | 179 |
| NX-10 | 229 |
| **PANASONIC** | |
| KX-P 1080i | 199 |
| KX-P 1091i | 259 |
| KX-P 3131 | 259 |
| KX-P 1592 | 399 |
| KX-P 1595 | 579 |
| KX-P 1092i | 329 |
| **MODEMS** | |
| Supra 300 ST | 59 |
| Supra 1200 ST | 149 |
| Avatex 1200 | 99 |
| Avatex 1200 HC | 129 |
| Compuserve Start Kt | 24 |
| Microstuffer | 59 |
| Omega Terminal | 20 |
| Atari 301 Modem | 49 |

**ST ACCESSORIES**

| | |
|---|---|
| Disk Cleaning Kit | 14 |
| Dust Covers | 7 |
| Monitor Stand | 12 |
| Mouse Pad | 8 |
| Shanner Planner | 29 |
| The Easel | 14 |
| Disc Directory | 20 |
| Pocket Pak | 10 |
| The Library | 34 |
| Flip N File Micro(25) | 6 |
| Flip N File " (50) | 17 |
| Flip N File II (40) | 20 |
| Flip N File Micro (5) | 2 |
| Flip Sort Micro (50) | 7 |

**ST BOOKS**

Atari ST GEM Reference
Atari ST Internals
Atari ST Machine Language
Atari ST Tips & Tricks
Atari ST from Basic to C
Atari ST Basic Training
Atari ST Graphic & Sound
Atari ST Logo
Atari ST Peeks & Pokes
Atari ST for Beginners

All books are $16 ea.

**CABLES**

| | |
|---|---|
| ST to Printer | 15 |
| ST to Modem | 15 |

**COVERS**

130XE Computer
800XL Computer
1050 Disk Drive
520ST Computer
SF354 Disk Drive
SF1224 Color Monitor
SM124 Mono Monitor
1040ST Computer

CALL

*To order call* **TOLL FREE**

# 1-800-824-7506

**ORDER LINE ONLY**

No extra charge for MasterCard or Visa

## COMPUTER CREATIONS, Inc.

P.O. BOX 493 - DAYTON, OHIO 45459

For information, order inquires, or for Ohio orders (513) 435-6868

VISA® MasterCard

Order lines Open 9:00 a.m. to 8:00 p.m. Mon.-Fri.; 10 a.m. to 4:00 p.m. Sat. (Eastern Standard Time). Minimum $15 per order. C.O.D. (add $3.00). Please specify computer system. Call toll free number to verify prices and availability. Prices and availability are subject to change without notice. We ship C.O.D. to Continental U.S. addresses only! Please include 4% shipping on all Hardware orders (min. $4.00). Software and accessories add $3.00 shipping and handling in continental U.S. Actual freight will be charged outside U.S. to include Canada, Alaska, Hawaii, Puerto Rico and APO. Ohio residents add 6½% sales tax. Canadian orders add 5% shipping, (min. $5.00). All other foreign orders, please add 15% shipping, (min. $10). For immediate delivery send cashier's check, money order or direct bank transfers. Personal and company checks allow 3 weeks to clear. School purchase orders welcome. Due to our low prices, all sales are final. NO CREDITS. All defective returns must have a return authorization number. Please call (513) 435-6868 to obtain an RA# or your return will not be accepted for replacement or repair. FOR YOUR PROTECTION WE CHECK FOR CREDIT CARD FRAUD. We do not bill until we ship.

```
3150 data 69,64,20,6C,61,62,65,6C,20,6
6,69,6C,65,21,20,7C
3160 data 20,5D,5B,20,53,6F,72,72,79,2
1,20,5D,0D,0A,0D,0A
3170 data 49,54,45,4D,20,53,45,4C,45,4
3,54,4F,52,00,20,43
3180 data 68,6F,6F,73,65,20,61,20,4C,6
1,62,65,6C,20,66,69
3190 data 6C,65,20,74,6F,20,70,72,69,6
E,74,3A,20,00,20,53
3200 data 61,76,65,20,4C,61,62,65,6C,2
0,74,6F,20,77,68,69
3210 data 63,68,20,66,69,6C,65,3F,20,0
0,4C,41,42,45,4C,53
3220 data 0D,0A,0D,0A,41,3A,00,00,00,0
0,00,00,00,00,00,00
3230 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3240 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3250 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3260 data 00,00,00,00,5C,2A,2E,4C,42,4
C,00,00,00,00,0F,2A
3270 data 00,00,11,42,00,00,0F,42,00,0
0,10,42,00,00,11,60
3280 data 00,00,12,60,00,6E,00,00,00,0
1,00,01,00,00,00,70
3290 data 00,02,00,01,00,00,00,01,00,6
F,00,00,00,01,00,00
3300 data 00,00,00,4E,00,01,00,01,00,0
1,00,00,00,2A,00,06
3310 data 00,01,00,01,00,00,00,2C,00,0
1,00,03,00,01,00,00
3320 data 00,2F,00,08,00,01,00,01,00,0
0,00,34,00,01,00,01
3330 data 00,01,00,00,00,36,00,00,00,0
5,00,01,00,00,00,32
3340 data 00,01,00,02,00,01,00,00,00,3
3,00,09,00,01,00,01
3350 data 00,00,00,5A,00,00,00,02,00,0
2,00,00,00,00,00,04
3360 data 36,0C,10,04,06,04,0A,0A,10,0
4,06,0A,04,06,10,24
3370 data 20,16,0E,12,08,0E,06,14,0E,1
2,04,0C,0C,08,0C,0C
3380 data 0C,0C,10,28,08,08,2A,16,22,0
E,04,12,08,10,0E,06
3390 data 10,06,08,0C,06,06,16,06,0C,0
A,0A,10,1A,04,06,0E
3400 data 06,10,06,2C,04,0E,0A,08,06,0
8,0C,1A,06,12,0A,10
3410 data 0C,10,06,04,06,06,14,06,06,0
E,0A,10,08,08,10,0A
3420 data 06,10,06,14,0C,06,06,12,06,0
C,0A,0A,10,16,12,06
3430 data 16,10,0A,10,06,1E,1E,04,08,0
6,0E,06,0C,10,18,0A
3440 data 06,0C,08,04,0E,1A,06,14,0E,0
8,10,06,14,0C,10,06
3450 data 14,0C,0C,20,08,46,1E,10,26,1
A,16,28,0A,06,04,0A
3460 data 06,06,04,0A,06,04,0A,04,06,0
4,0A,04,06,04,06,04
3470 data 06,04,06,0C,0A,04,06,08,06,0
4,06,04,06,04,06,04
3480 data 06,04,0A,04,06,04,06,04,0A,0
4,0A,0A,04,06,04,08
3490 data 08,0C,0C,08,08,06,04,06,04,0
6,04,06,04,0A,04,06
3500 data 04,06,04,0A,06,06,06,06,06,0
6,08,10,06,0A,06,06
3510 data 04,06,04,06,04,06,04,0A,04,0
6,04,06,06,04,06,04
3520 data 06,04,06,04,06,04,06,06,04,0
A,06,04,06,04,06,04
3530 data 0C,16,08,26,10,01,01,4A,04,0
4,04,04,04,00
3540 data *
```

ST CHECKSUM DATA.

*(see page 8)*

```
 100 data  791, 948, 117, 614, 503, 21
5, 410, 427, 14, 109, 4148
 190 data  647, 357, 700, 578, 792, 85
8, 840, 806, 742, 793, 7113
 1080 data  656, 677, 719, 718, 890, 7
44, 758, 740, 949, 850, 7701
 1180 data  872, 901, 831, 823, 860, 9
30, 14, 944, 788, 880, 7843
 1280 data  763, 741, 797, 853, 690, 7
06, 9, 839, 786, 6, 6190
 1380 data  736, 788, 651, 696, 908, 7
28, 818, 799, 875, 804, 7803
 1480 data  873, 856, 851, 814, 779, 7
24, 740, 724, 881, 827, 8069
 1580 data  823, 831, 844, 847, 816, 8
06, 841, 794, 790, 729, 8121
 1680 data  906, 885, 842, 706, 884, 8
95, 723, 817, 918, 741, 8317
 1780 data  856, 798, 783, 904, 830, 9
18, 848, 846, 743, 899, 8425
 1880 data  775, 869, 682, 738, 908, 7
78, 820, 929, 740, 867, 8106
 1980 data  867, 735, 835, 833, 772, 8
56, 839, 694, 784, 778, 7993
 2080 data  983, 702, 792, 780, 885, 8
11, 725, 980, 713, 41, 7412
 2180 data  863, 891, 945, 716, 783, 5
0, 743, 904, 689, 846, 7430
 2280 data  819, 745, 829, 786, 809, 8
46, 827, 894, 883, 820, 8258
 2380 data  61, 876, 703, 655, 755, 80
0, 809, 697, 816, 792, 6964
 2480 data  823, 771, 704, 783, 817, 8
62, 787, 716, 724, 826, 7813
 2580 data  741, 821, 685, 830, 767, 8
41, 806, 790, 827, 721, 7829
 2680 data  850, 796, 812, 10, 822, 82
0, 804, 775, 746, 774, 7209
 2780 data  874, 850, 807, 776, 804, 7
57, 730, 775, 847, 836, 8056
 2880 data  60, 788, 780, 847, 835, 83
4, 892, 907, 839, 841, 7623
 2980 data  831, 872, 745, 803, 814, 8
72, 834, 841, 876, 825, 8313
 3080 data  871, 826, 738, 669, 772, 5
59, 831, 826, 802, 751, 7645
 3180 data  839, 797, 826, 789, 578, 4
66, 467, 468, 674, 563, 6467
 3280 data  556, 517, 531, 499, 521, 5
02, 497, 511, 610, 670, 5414
 3380 data  653, 645, 643, 610, 611, 6
53, 637, 677, 590, 586, 6305
 3480 data  605, 607, 585, 564, 567, 1
7, 210, 3155
```

●

Listing 2.
ST BASIC listing.

```
100 filename$="a:\labelmak.rsc"
110 fullw 2:clearw 2:gotoxy 0,0:print
"creating file..."
120 option base 0
125 dim a%(16000):def seg=1:v$=""
130 p=varptr(a%(0)):bptr=p+1
140 for i%=1 to 1382
150 read v$:code%=val("&H"+v$)
160 poke p, code%:print ".";
170 p=p+1
180 next
190 bsave filename$,bptr,1382
200 print "file written":end
1000 data 00,00,03,4E,02,52,02,52,02,5
2,00,00,00,24,02,52
1010 data 00,00,05,5E,00,16,00,01,00,0
9,00,00,00,00,00,00
1020 data 00,00,05,62,20,20,20,20,53,5
4,20,4C,61,62,65,6C
1030 data 20,50,72,69,6E,74,65,72,20,2
0,20,20,00,78,20,31
1040 data 39,38,36,20,43,68,61,72,6C,6
5,73,20,46,2E,20,4A
1050 data 6F,68,6E,73,6F,6E,00,00,00,4
E,75,6D,62,65,72,20
1060 data 6F,66,20,6C,61,62,65,6C,73,0
0,00,00,30,31,00,00
1070 data 00,45,6E,74,65,72,20,61,20,6
6,6F,75,72,2D,6C,69
1080 data 6E,65,20,61,64,64,72,65,73,7
3,20,6C,61,62,65,6C
1090 data 20,62,65,6C,6F,77,3A,00,00,0
0,5F,5F,5F,5F,5F,5F
1100 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1110 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,00,5F,5F,5F,5F,5F
1120 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1130 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,00,58,58,58,58
1140 data 58,58,58,58,58,58,58,58,58,5
8,58,58,58,58,58,58
1150 data 58,58,58,58,58,58,58,58,58,5
8,58,58,00,5F,5F,5F
1160 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1170 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,00,5F,5F
1180 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1190 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,00,58
1200 data 58,58,58,58,58,58,58,58,58,5
8,58,58,58,58,58,58
1210 data 58,58,58,58,58,58,58,58,58,5
8,58,58,58,58,58,00
1220 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1230 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1240 data 00,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1250 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1260 data 5F,00,58,58,58,58,58,58,58,5
8,58,58,58,58,58,58
1270 data 58,58,58,58,58,58,58,58,58,5
8,58,58,58,58,58,58
1280 data 58,58,00,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1290 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1300 data 5F,5F,5F,00,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1310 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,5
F,5F,5F,5F,5F,5F,5F
1320 data 5F,5F,5F,5F,00,58,58,58,58,5
8,58,58,58,58,58,58
1330 data 58,58,58,58,58,58,58,58,58,5
8,58,58,58,58,58,58
1340 data 58,58,58,58,58,00,4F,70,74,6
9,6F,6E,73,3A,00,00
1350 data 00,50,72,69,6E,74,00,50,72,6
9,6E,74,20,46,69,6C
1360 data 65,00,53,61,76,65,20,4C,61,6
2,65,6C,00,45,58,49
1370 data 54,00,00,00,00,3D,00,00,00,5
7,00,00,00,58,00,05
1380 data 00,06,00,02,11,80,00,00,FF,F
F,00,1A,00,01,00,00
1390 data 00,59,00,00,00,6A,00,00,00,6
B,00,05,00,06,00,02
```

```
1400 data 11,82,00,00,FF,FF,00,11,00,0
1,00,00,00,6C,00,00
1410 data 00,6F,00,00,00,70,00,03,00,0
6,00,02,11,80,00,00
1420 data FF,FF,00,03,00,01,00,00,00,7
1,00,00,00,98,00,00
1430 data 00,99,00,05,00,06,00,02,11,8
1,00,00,FF,FF,00,27
1440 data 00,01,00,00,00,9A,00,00,00,B
B,00,00,00,DC,00,03
1450 data 00,06,00,00,11,80,00,00,FF,F
F,00,21,00,21,00,00
1460 data 00,FD,00,00,01,1E,00,00,01,3
F,00,03,00,06,00,00
1470 data 11,80,00,00,FF,FF,00,21,00,2
1,00,00,01,60,00,00
1480 data 01,81,00,00,01,A2,00,03,00,0
6,00,00,11,80,00,00
1490 data FF,FF,00,21,00,21,00,00,01,C
3,00,00,01,E4,00,00
1500 data 02,05,00,03,00,06,00,00,11,8
0,00,00,FF,FF,00,21
1510 data 00,21,00,00,02,26,00,00,02,2
F,00,00,02,30,00,05
1520 data 00,06,00,02,11,81,00,00,FF,F
E,00,09,00,01,FF,FF
1530 data 00,01,00,10,00,14,00,00,00,1
0,00,02,11,62,00,0C
1540 data 06,02,00,38,00,0F,00,05,00,0
2,00,02,00,14,00,00
1550 data 00,20,00,FF,11,21,00,02,00,0
1,00,24,00,02,00,01
1560 data 00,03,00,04,00,14,00,00,00,0
0,00,00,11,00,00,06
1570 data 00,00,00,18,00,02,00,04,FF,F
F,FF,FF,00,1C,00,00
1580 data 00,00,00,00,00,24,00,00,00,0
0,00,18,00,01,00,02
1590 data FF,FF,FF,FF,00,15,00,00,00,0
0,00,00,02,52,00,00
1600 data 00,01,00,18,00,01,00,0A,00,0
6,00,09,00,14,00,00
1610 data 00,20,00,FF,11,00,00,28,00,0
1,00,0E,00,03,00,07
1620 data FF,FF,FF,FF,00,16,00,00,00,0
1,00,00,02,6E,00,00
1630 data 00,00,00,0E,00,01,00,08,FF,F
F,FF,FF,00,1B,00,40
1640 data 00,00,02,FF,11,00,00,00,00,0
1,00,04,00,02,00,09
1650 data FF,FF,FF,FF,00,16,00,00,00,0
0,00,00,02,8A,00,04
1660 data 00,01,00,06,00,02,00,05,FF,F
F,FF,FF,00,1B,00,40
1670 data 00,00,01,FF,11,00,00,0A,00,0
1,00,04,00,02,00,0B
1680 data FF,FF,FF,FF,00,16,00,00,00,2
1,00,00,02,A6,00,02
1690 data 00,04,00,24,00,01,00,10,00,0
C,00,0F,00,14,00,00
1700 data 00,20,00,FF,11,00,00,02,00,0
5,00,24,00,09,00,0D
1710 data FF,FF,FF,FF,00,1D,00,08,00,0
0,00,00,02,C2,00,02
1720 data 00,01,00,20,00,01,00,0E,FF,F
F,FF,FF,00,1D,00,08
1730 data 00,00,00,00,02,DE,00,02,00,0
3,00,20,00,01,00,0F
1740 data FF,FF,FF,FF,00,1D,00,08,00,0
0,00,00,02,FA,00,02
1750 data 00,05,00,20,00,01,00,0B,FF,F
F,FF,FF,00,1D,00,08
1760 data 00,00,00,00,03,16,00,02,00,0
7,00,20,00,01,00,00
1770 data 00,11,00,15,00,14,00,00,00,2
0,00,FF,11,33,00,28
1780 data 00,05,00,0E,00,09,00,12,FF,F
F,FF,FF,00,16,00,00
1790 data 00,01,00,00,03,32,00,00,00,0
0,00,0E,00,01,00,13
1800 data FF,FF,FF,FF,00,1A,00,05,00,0
0,00,00,02,31,00,00
1810 data 00,02,00,0E,00,01,00,14,FF,F
F,FF,FF,00,1A,00,05
1820 data 00,00,00,00,02,37,00,00,00,0
4,00,0E,00,01,00,15
1830 data FF,FF,FF,FF,00,1A,00,05,00,0
0,00,00,02,42,00,00
1840 data 00,06,00,0E,00,01,00,10,FF,F
F,FF,FF,00,1A,00,25
1850 data 00,00,00,00,02,4D,00,00,00,0
8,00,0E,00,01,00,00
1860 data 03,4E,00,00,03,52
1870 data *
```

•

ST CHECKSUM DATA.

*(see page 8)*

```
 100 data  794, 948, 117, 614, 503, 21
3, 410, 427, 14, 109, 4149
 190 data  645, 357, 575, 514, 677, 70
6, 766, 835, 708, 821, 6604
 1080 data  828, 864, 48, 12, 50, 958,
 828, 848, 54, 18, 4508
 1180 data  56, 6, 827, 805, 53, 54, 1
8, 56, 824, 834, 3533
 1280 data  994, 60, 17, 55, 865, 833,
 817, 813, 775, 567, 5796
 1380 data  703, 579, 713, 566, 704, 7
35, 668, 694, 623, 709, 6694
 1480 data  590, 793, 695, 536, 868, 5
30, 535, 594, 498, 866, 6505
 1580 data  500, 856, 523, 616, 876, 8
89, 585, 884, 876, 613, 7218
 1680 data  908, 546, 618, 921, 888, 5
94, 948, 892, 508, 624, 7447
 1780 data  883, 526, 866, 886, 531, 8
73, 895, 546, 739, 218, 6963
```

•

## 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

# Assembly line

## Last month's program — the rest of the story.

by Douglas Weir

If this were a column on 6502 assembly language programming, we would already have enough background to write some useful routines. The 68000 is a much more sophisticated chip, and that means a bigger talk-to-code ratio in these first few installments. But it also means that, as time goes on, it will be easier to write equivalent routines for the 68000 than it would be for the 6502.

Last month's program is a case in point. It doesn't do much, but in order to understand how it works, there are several new instructions and addressing modes we must look at. First, the addressing mode.

### And you thought the 9-digit zip code was confusing . . .

Although the trap instructions probably stand out, it's the addressing mode used with the move instructions that requires the most explanation. This mode—indicated by parentheses around an address register name preceded by a minus sign—is connected with a special use of memory that's vital to the way any microprocessor operates. To make a long story short, a block of memory is necessary as a "scratch" area for important (but often changing) internal data.

To learn how this works, we'll start with a simple example. Using memory between the addresses 2400 and 2800 as temporary storage, we make, say, register a0 point to the "end" of this area:

```
movea.l   #2800,a0
```

(Movea—move address—is the correct form of move to use when the destination operand is an address register.) Now let's save a few numbers.

```
move.l    #520,-(a0)
move.l    #1024,-(a0)
move.l    #260,-(a0)
move.l    #1987,-(a0)
```

In each of these instructions the source operand is familiar. A number prefixed with a # means we're using the number as an immediate value, not as an address. Where are we putting these numbers? The parentheses around a0 mean that the destination is not a0 itself, but rather the memory location specified by a0's contents. So, in the first line of code, if a0 contains the address *2800*, then the number *520* is stored (as a longword) at address *2800*. But . . . a0 no longer contains *2800* by the time the number is stored, and that's where the minus sign comes in.

The value held in a0 is used as an address, but the minus sign means that a certain amount is subtracted from it first. This amount is just enough to make "room" for the value to be stored. Since we specified with the .l modifier that the value is to be handled as a longword (4 bytes), the amount subtracted from a0 is 4. This new addressing mode is called Address Register Indirect with Predecrement.

It's a lot like storing numbers into an array in BASIC or Pascal. We would start with an array subscript 1 greater than the number of elements. We write into the array beginning from the end. To write into the "first" (really the last) element, we subtract 1 from the subscript. To write into the next element, we subtract from the subscript again, and so on. Taking numbers out of the array involves a reverse process: first read the number at the current subscript value, then add 1 to the subscript. At all times the subscript "points" to a written-in element. So we must always subtract from the subscript to access a "free" element. Note that reading from an element also frees it again: it's as if we actually remove the value when we read it.

Writing into the array is a two-step procedure: subtract from the subscript, then write into the current element. Reading from the array is the opposite: read the current element, then add to the subscript. We must be careful not to let the subscript go outside the array bounds, but that's another story.

Here are those four lines of code again, with the current effective value of a0 indicated in the comment field. This value is the address at which the source operand is being stored.

```
move.l   #520,-(a0)  addr=2796
move.l   #1024,-(a0) addr=2792
move.l   #260,-(a0)  addr=2788
move.l   #1987,-(a0) addr=2784
```

Memory location 2796 now contains the number *520*, 2792 contains the number *1024*, 2788 contains *260*, and 2784 contains *1987*. After the last instruction is executed, the value of a0 is *2784*. Every one of the numbers is stored as a longword, which means that we're wasting some space: none of the numbers requires more than 3 bytes.

Now let's remind ourselves why we're going to all this trouble just to save a few numbers. After all, nobody uses arrays in the way described above. Instead, we usually decide which element we want to use, get its subscript, and read from or write into it. The advantage of using an array in the somewhat screwy way we did above is that we never have to bother directly with the subscript at all. As long as we remember always to subtract from it before writing, and add to it after reading, we have a large temporary storage area that requires a minimum of upkeep.

Suppose we want to get these numbers back and put them into various data registers. Here's how we'd do it:

```
move.l   (a0)+,d0   get the last
                    number...
move.l   (a0)+,d1   then the next
                    one...
move.l   (a0)+,d2   and so on...
move.l   (a0)+,d3   to the end
```

Register d0 now contains *1987*, d1 contains *260*, d2 1024, and d3 *520*. The value of register a0 after the last instruction is *2800* again, and for all intents and purposes, our temporary memory area is now empty. The plus sign (+) after the parenthesized register name means that the register's current value is used as an address, and after the instruction is executed, a certain amount is added to it. As you would expect, this amount is equal to the number of bytes read during the operation—here, 4 bytes (a longword). This addressing mode is called Address Register Indirect with Postincrement.

There's no rule that says we must use longword values in these operations. It's perfectly okay to store and retrieve single bytes or words in this way. You can even mix data sizes—storing a byte, then a longword, a couple of words, and so on. There are two things to be careful of. First, make sure that the data size you specify is big enough to hold the number (the assembler will catch this kind of error with some sort of "data overflow" message).

Second, you must make sure, when you're reading a saved value, that you're using the same data size you used when you saved it. The assembler can't catch this kind of "runtime" error. If, for example, you save a longword and then retrieve a word, the value you've read obviously won't be the same as what you originally wrote. Worse: the pointer value of the address register will be "out of sync," and all the rest of your reads will be erroneous. The easiest thing to do is simply stick to longwords, as long as execution speed isn't important.

### Look Ma—it's a stack.

So far we've been looking at this block of memory as an array. Most people imagine an array as a horizontal data structure, with elements running left to right. But if you take this array and stand it on end with its highest-numbered element at the base, you can give it a different name: now it's a "stack." The address register used to keep track of our position in the stack—a0 in the examples—is called the "stack pointer."

Writing data into a stack—decrementing the stack pointer, then writing to the address it contains—is called a "push." When you read data from a stack, you "pop" it. Everything we've been saying about these operations remains the same; we're just using different terms now.

Stacks are very important in assembly language. The 68000 allows you to use any address register as a stack pointer, just as we used a0 above. But, whether or not you choose to use stacks, you must always keep in mind that the 68000—like all microprocessor—is using one of its own. The 68000's "system" stack pointer (which is always register a7) is almost as important as the program counter itself.

Here's why: assembly language has facilities for writing subroutines, similar to those in BASIC or C. You "call" a subroutine in BASIC by means of a GOSUB statement; in C, simply by using the subroutine's name. At the end of the routine, execution "returns" to the spot in the program just after where the call was made. The programmer doesn't have to worry about how the computer remembers where to return to.

In assembly language the process is similar, but it's not nearly so transparent. Jumping or branching to a subroutine is a matter simply of loading the program counter with the address of the first instruction of the subroutine we want to execute—we talked about this a little last time, in connection with labels. What about returning from a subroutine? Obviously, this would involve loading the program counter with the "return address"—the address of the next instruction after the jump or branch we originally executed. This means that the return address must be saved at the time of the jump.

Saving the return address is not the programmer's responsibility. It's an automatic part of the execution of a bsr (branch to subroutine) or jsr (jump to subroutine) instruction. How is it saved? On the stack, of course. It's done this way because the 68000 has to be able to return from nested jumps. If you jump to subroutine A, and in the middle of it jump to subroutine B, then you want to return to A from B, and finally to the original caller when A is completed. If a special register was used explicitly to save the current return address, then each new jump would cause the last return address (if there was one) to be overwritten with the new one. This problem doesn't occur with a stack.

The item of data you pop off a stack is always the last item pushed. As you continue popping off items (I apologize for this turn of phrase), you are retrieving them in reverse order to that in which they were pushed. That's exactly how we want to return from nested subroutine calls—in reverse order to the calls themselves. The 68000 instruc-

tion that pops the return address from the top of the stack and loads it into the program counter is rts—Return from Subroutine.

When you consider that most programs are full of such jumps (assembly language uses subroutines just as much as high-level languages do), it's easy to see how important the 68000's system stack is. Suppose something goes wrong with a7. If an rts instruction is executed and, for some reason, the top value on the stack is *not* the correct return address, then the result will be just like loading a random value into the program counter: at best you'll see any number of bombs on the ST's screen; at worst the system will crash completely.

Of course, programmers do use the system stack for other things—the ST's operating system (TOS) requires you to pass parameters to system subroutines using the stack rather than specific registers or memory locations. The reason for this is much the same as before: it allows nested subroutine calls with different sets of parameters—to a certain depth, at any rate.

Note that our stacks "grow down" in memory. In other words, when an item is pushed, the stack pointer is decremented. Pushing 100 words would result in the stack pointer's having a value 200 less than it had before—400 less if we pushed all longwords. Popping items increments the stack pointer. Since stack operations on the 68000 are implemented with move instructions, we can reverse addressing modes and have stacks that grow the opposite way, if we wish. However, we have no control over the way the system stack works.

There's one other special characteristic of a7: values can be pushed and popped from the system stack only as words or longwords. This is done to make sure that a7 is always set to an even address—the possibilities for havoc otherwise are just too great. If you do specify .b size, the values are handled as words anyway. You can push and pop bytes using the other address registers.

### You pushed those parms, you clean 'em up.

Now, perhaps, last month's program is becoming a bit clearer. A trap instruction is just a very specialized kind of subroutine call: a trap #1 executes a jump to a preset address stored near the bottom of the ST's memory. TOS sets this address to point to a system routine that takes a quick look at the parameters being passed. One of these is used to identify which one of a set of subroutines to jump to, in order to execute the service requested.

The first two instructions in the program push the required parameters on the stack for such a system call (GEMDOS function nr 9—you can find it on page 109 of the *ST Internals* book published by Abacus). Now comes the trap instruction: the return address is pushed on the stack, the jump is executed and the function performed, and, finally, the return address is popped from the stack. None of this is reflected in the source code—it's all a result of the trap. When the return address is successfully loaded into the program counter, we're back where we came from—the next instruction after the trap:

```
addq.l  #6,a7
```

Forget about the q for a moment. This is really just an add instruction—all we're doing is adding the value 6 to the contents of the system stack pointer.

It looks at first as though I'm indulging in just the sort of foolishness I warned against—fiddling with the system stack. But if you look closely, you'll see that this is no more than a bit of necessary cleaning up.

You'll remember the two parameters I pushed on the stack for the system call. The effect of executing the trap and returning at its end is to leave the stack just as it was immediately before the trap. In other words, the two parameters I pushed are still on the stack. I could take them off again with the following sequence:

```
move.w  (a7)+,d0    pop 2nd argument
move.l  (a7)+,d0    pop 1st argument
```

. . . But, since we're not interested in the values anymore, just in removing them, it's quicker to simply add a value to the stack pointer equal to the combined size of the data items we want to remove. Since we originally pushed a word and a longword, this value is 6.

You may be wondering how in the world the parameters were read by the TOS subroutine without popping them from the stack. Well, they could have been popped and repushed; but I don't believe anyone does this sort of thing, even in Sunnyvale. There is a way of peeking at the stack without changing the stack pointer. It requires using an address mode we haven't learned yet, but don't worry—we will soon.

What about that q in addq ? It stands for "quick," as in "add quick." (Adverbs were banned from Motorola documentation in the early 1980s.) In a future installment I'll have more to say about quick instructions. What it amounts to is this: if an immediate value is small enough, the assembler can fit it into the word of memory which contains the machine code, depending on how many free bits there are in the word. That makes the instruction execute faster, since less memory has to be fetched into the program counter at run time.

Now we're finally in a position to go through the program, line by line, and explain what it does.

The first line (with a * as its leftmost character) is a comment, ignored by the assembler. The next line (the word text) is an "assembler directive"—it simply tells the assembler that the following lines will contain instructions. Text itself doesn't generate any code.

The instructions follow. None of them are labelled, so this field is empty, but each of the next nine lines does have an instruction, operand and comment field. First, the address of the string we want to display is pushed onto the system stack. The string itself is declared in the second-to-last line of the program, and it has a label, test_str. As we know, a label is really just a name that the assembler associates with a memory location (address). So if we want to get at the address itself, we use the label as an immediate value. Addresses are always considered to be 4 bytes long in 68000 assembly language, so the value is pushed as a longword.

There are three groups of system subroutines in the ST that can be used by the programmer to perform various tasks. All are accessed via the trap instruction. Trap #1

routes us to the GEMDOS routines, trap #13 to the BIOS (Basic Input/Output System), and trap #14 to the XBIOS (Extended BIOS) routines. The latter two groups handle functions that tend to be peculiar to the ST's hardware configuration: e.g., the mouse, screen resolution, the MIDI interface, etc. GEMDOS functions are more "general purpose" routines: get a disk directory, read the keyboard, etc. The last parameter pushed on the stack (always a word) specifies which function we want performed.

In this case, we want function 9, which displays a string on the ST's screen. Executing the trap instruction accomplishes the GEMDOS call. Afterwards we clean up the stack. Now we want the message to stay on the screen for a while, so we call another GEMDOS routine, nr 1, that waits for a key to be pressed. This time all that's required is to push the function number onto the stack and execute the trap. As soon as a key is pressed, execution will return to our program. The correct keycode will be in register d0, but we're not interested in that here, so we just clean up the stack again and continue.

The last two instructions request GEMDOS function 0, which terminates the current program. Obviously there's no return from such a call, so the program proper ends here.

**If you liked form 1040, you'll love this.**

I've reprinted the program below, but in a different form. This time we're looking at the listing file which AS68 outputs (if you request it). The page headings have been eliminated to save space. The rightmost three columns consist of the original source file, but three more columns of numbers now appear on the left.

The first column contains decimal line numbers for the input source file. Every source line, including comments and blank lines, is numbered here. The next column contains 8-digit hexadecimal numbers which give a sort of running "byte count" of object code generated. In other words, assume that the machine codes specified by the source code instructions are actually being written by the assembler into a "work area" in memory. If we number the bytes in this area from 0 on, then the second column tells us at what byte the current line's worth of machine code will begin.

Take the first instruction as an example. It turns out that the instruction move.l #test_str,-(a7) occupies 6 bytes of memory once it's translated into machine code. Now look at the second line of code in the listing. Sure enough, the second column reads *0000006*, meaning that we're now at byte 6 in our hypothetical program space (bytes 0-5 having been filled). The second instruction fills up bytes 6 through 9, so the byte count afterwards is *0000000A*, which is the hexadecimal equivalent of decimal *10*. As the assembler fills its work area with the assembled machine codes,

it keeps track of the current location with a "location counter." You can think of the second column as showing the current location counter value.

The third column of numbers is also in hex. Here, as you might expect, you can read the actual machine code translations of the source code instructions. Looking at the first line again, and recalling that every two hex digits is equivalent to a byte, you can easily verify that the size of the first assembled instruction is 6 bytes.

The numbers in this third column, taken together, show approximately what the program will look like in memory. Some information is omitted by the assembler, although space is left for it. For example, note the eight *0*s at the end of the first machine code (2F3C). This is where the address of test_str will be inserted, though it's not there yet. The eight hex *0*s occupy 4 bytes of memory, exactly the amount of space required for a valid 68000 address. The address will be inserted by the linker. There's a reason for this, but it's not important at the moment.

There are four more lines in the program. The first contains the assembler directive data, which tells the assembler that from now on it will be looking at data declarations rather than instructions. The directive on the next line, even, tells the assembler to make sure that the location counter has an even value before it fills any more values into its work area. This is a useful directive when you want to declare space for word- or longword-size variables. A string is composed of byte-size codes, however, so there's not much reason for even here. I couldn't resist putting it in, though—it's one of my favorites.

The source code for our string declaration consists of, first, a label, followed by another assembler directive, dc. This stands for "define constant"—in other words, reserve space in the work area for the values indicated and insert the values in the space reserved. The .b modifier means that a series of one or more byte-size values, indicated by the programmer, will follow. Here you can see two of the ways the values can be written. The string Hello world is read by the assembler, separated into a series of ASCII codes, and filled into the program's data area. Following the string is a comma, then three decimal numbers, separated by commas. Each is assembled as a separate byte value. The *10* is the ASCII code for linefeed, and *13* is the code for carriage return. The *0* is required by the TOS routines to mark the end of a string—it's the same thing as the null byte at the end of a C string.

The series of hex digits to the left of test_str in the listing shows exactly how the string will look in memory. The string includes its own carriage return and linefeed—the TOS routine will print everything up to the 0. The hex translation of the string was too long for one line, so the assembler split it into two, both of which have the same line number, 16. Note that there should be no spaces in the string definition (except within quotes). A space tells the assembler that the definition is complete, and that anything else on the line is a comment.

The last line, of course, is a final comment.

When the assembler is finished translating your source code, it creates a disk file and writes the contents of its work area into it, along with some other information required by the linker.

This month we've covered two new addressing modes, the trap instruction, how the stack works, the elements of an assembly listing, and how to declare data in assembly language. Looping and testing conditions are the next items on the agenda. ///

**Listing 1.**
**Assembly listing.**

```
C P / M   6 8 0 0 0   A s s e m b l e r   Revision 04.03    Page   1
Source File: TEST.S

 1
 2                          *** this is a program:
 3 00000000                 text                    code segment
 4 00000000 2F3C00000000    move.l    #test_str,-(a7)  get string address
 5 00000006 3F3C0009        move.w    #9,-(a7)         code=display string
 6 0000000A 4E41            trap      #1               do it
 7 0000000C 5C8F            addq.l    #6,a7            fix stack
 8 0000000E 3F3C0001        move.w    #1,-(a7)         code=wait for keypress
 9 00000012 4E41            trap      #1               do it
10 00000014 548F            addq.l    #2,a7            fix stack
11 00000016 3F3C0000        move.w    #0,-(a7)         code=exit program
12 0000001A 4E41            trap      #1               do it
13
14 00000000                 data                  data segment
15                            even                start with even address
16 00000000 48656C6C6F20776F  test_str  dc.b    'Hello world',10,13,0    a string
16 00000008 726C640A0D00
17                          *** ...and that's all there is!
18

●
```

# Publishing Partner

**SOFTLOGIK CORP.**
**4129 Old Baumgartner Road**
**St. Louis, MO 63129**
**High or medium resolution $150.00**

**by Matthew J.W. Ratcliff**

**Publishing Partner** from SoftLogik is a serious desktop publishing application for the Atari ST. An equal opportunity idea processor, it will allow you to create breathtaking documents on a simple dot-matrix printer. Of course, the Epson and Gemini printers are supported, but so are the Apple LaserWriter and other PostScript (a page-definition language for laser printers developed by Adobe Systems) capable laser printers. More printer drivers are coming out every day, including drivers for the Prowriter and for the newer 24-pin printers, like the NEC P6, which can generate images to rival the quality of low-priced laser printers.

To prove the program's integrity—and to alpha and beta test it—the documentation was created with **Publishing Partner** itself, then printed on the Apple LaserWriter.

The documentation is well done and fairly complete, including an index. The biggest problem is that it refers constantly to Atari's GDOS, although **Publishing Partner** was released without GDOS or any support for that operating system. It was a bold move, considering that Batteries Included is still holding back on the release of **PaperClip Elite** until a reasonably finalized GDOS version is completed by Atari. Due to GDOS's inflexibility with font controls (a separate font file must be loaded for each different point size of text you desire) and lack of printer drivers (only the Epson is currently supported), SoftLogik was forced to create their own font and printer driver code. The end results are quite impressive.

The first release of **Publishing Partner** had numerous minor bugs; I spent a lot of time bombing back to the desktop. SoftLogik had shipped over 2000 copies of version 1.00 by the time 1.01 was complete. Yet they shipped a *free* update to everyone who sent in a registration card! I'm writing this review with 1.02. It seems much improved over predecessors, which I'll detail later.

To begin a document, you need only pull down the file menu and click on "New Document," or press ESCAPE-N. Most **Publishing Partner** commands have keyboard equivalents, letting the power user breeze through the program's operations. You have many standard formats to choose from, or you can create your own. If you select "Business Card," for example, your display will be exactly the same size as your business card. Precise dimensioning and positioning are among the program's better features.

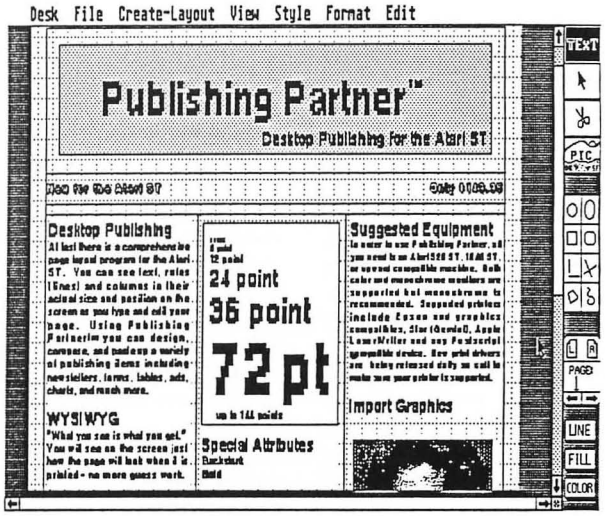
Once created, you have a *page* on your display, with a small dotted grid, at ¼-inch spacing. Now you can assemble columns of spectacular text. You simply drag a box onto the display, switch to text mode (a simple point and click in the "Tool Box" icons window to the right) and begin typing.

When you create columns, you can accept a default or design a custom layout. And by no means are you limited to one configuration. Just switch to the object editor mode (click on the arrow in the toolbox), select any column and resize it.

As you type in a column, **Publishing Partner** continuously formats your text, with microspace justification on the screen. What you see is what you get, with a minor difference: the printouts are much better. You can select from 3- to 72-point on-screen text. The larger your letters, the chunkier they are on the display. But, no matter what size the text is, it always retains a smooth, typeset quality when printed. The screen fonts are rather simplistic forms that give you a good approximation of what you'll get when printing. These screen fonts are only 4K long; the printer fonts (.PNF files) are a full 32K!

You have many special text-effects features available at any time. Under the style menu are Backslant, Bold, Double Underline, Italicize, Light, Mirror, Outline, Reverse Text, Shadow, Strike Through, Tall, Underline, Wide and Normal options. All these have logical keystroke-command equivalents. Most can be mixed—you can specify Tall, Bold, Shadowed Text—if you like. If you realize that part of your text needs to be italicized, you can go back and drag the cursor over that area (move the mouse while holding down the left button) to highlight it, then enable the style you want it converted to.

Also under the style menu, you can access the font selection features (or press




ALT-F) to choose a different font or set a different point size. [A point equals 1/72 of an inch—72-point text is 1 inch tall from the top of an ascender (or capital) to the bottom of a descender (or lowercase g, for example).]

I've found with version 1.02 of **Publishing Partner** that, if you select large point sizes, the characters seem to "blur." The software seems to have trouble blowing them up on the display. The printouts still look fine, and the program does *not* crash. The problem occurs in displaying actual size type, but not in the small screen display formats. It's an annoyance you can work around.

To resize an area of text, you simply highlight it, go to the font menu and change point size. While you're in object editor mode, an entire column can be selected and resized or restyled in much the same way. When these functions are performed, a dialog box comes up to alert you of a "global" operation. When a column is selected, you can drag any of the four corners to resize it, and the program automatically reflows the text in it.

Under the format menu (or appropriate ALT-key command), there's a wide variety of text manipulation features. Super- and subscripts are supported, and easily changed. You can highlight an area of text to convert its case from upper to lower, or the reverse.

Continuous text formatting can be set up in several ways. Text can be blocked left, right or centered. Columns can be linked together with the "set text routing" feature. Then, when your text overflows a column, it automatically flows into the next. If you have many pages routed, this can be quite slow. It's best to break the text routing into small segments (like logically grouped pages), then perform a final link when the document's complete.

Text justification is done by character or by word. Character, or microspace, justification places an equal amount of white space between all characters to fill out a line. Word justification maintains consistent spacing between characters of a word, and expands the spaces between words to fill a line. The fonts provided are proportional. The format menu is rounded out with the ability to change margins, set line and character spacing, and perform manual kerning and hyphenation.

The usual edit features of a good word processor are also available in **Publishing Partner**. Buffer operations let you cut and paste text or graphics, such as **DEGAS** and **NEO-Chrome** pictures (more on the graphics goodies later). Deleting, searching and replacing round out the editor's features.

Clicking on the "Pic" icon in the toolbox switches you to the picture buffer. You can now "import" a **DEGAS** (.PI1 through .PI3), **NEO-Chrome** (.NEO) or Tiny (.TNY) picture. **Publishing Partner** judges from the file extension what format file you plan to load, and adjusts automatically.

It will remap a **DEGAS** picture of low or medium resolution into high resolution if you're in monochrome. (Actually, all pictures are converted to monochrome format for printing, regardless of their original format.) I've found that **Publishing Partner** does a simple bitwise conversion; **DEGAS Elite** does a much better job of converting to monochrome from other formats.

I've also found that, when monochrome pictures created with **ComputerEyes ST** are loaded, they become negatives. **Publishing Partner** assumes that, if a bit is set, it's a black dot on the display. It does not look at the color table associated with the picture. The only way around this (until a utility comes along) is to load the picture into **DEGAS Elite** and have that software remap the colors.

Once the picture is in its buffer, you can drag a box around all or part of it, then copy or move it to the buffer. While in the picture buffer, you can't perform editing functions on the picture. Hardly any menu functions apply in the picture buffer—and the menus are *not* shown in light text to indicate that they're disabled. As a matter of fact, you can select some of them while in the picture buffer and actually change the text document (botching it but good, if you aren't careful).

So do all of your picture editing with **DEGAS** or **NEO-Chrome** before going to **Publishing Partner**. Eventually, some simple editing features may be added.

Once your picture's been moved to the

buffer, you can click back to the text mode and copy from the buffer. A picture frame appears, showing you where the top left corner of your artwork will be placed. Point and click, then select the object editor to position and resize. Note that resizing will adjust proportionality, too. If there's too much fluff, you can click on the scissors icon and crop the edges of the selected artwork.

You can use **Publishing Partner**'s other graphics features—such as boxes, lines, circles or freehand drawing—to overlay graphics on your document or graphics. But if anything moves, your overlay work will go "out of sync." There's no way at present to group a series of objects together (locking their relative positions) or to freeze them in place on a page—so you can't accidentally select and move them later (easy to do with a tightly packed group of objects).

If you do plan to overlay lines on a picture, do so in actual size. **Publishing Partner** cannot resize pictures above 100 percent, and below that your placement won't be much more than an educated guess.

Once your pictures are positioned and sized correctly, you can turn off picture display from the view menu. The picture will be shown simply as an empty box onscreen, allowing for faster screen updates.

The view menu gives you many ways of looking at your document. You can display it as multiple pages, two squished pages at a time. You can look at it full size, half size, full page, full width, doubled, or set your own defined scale. I find myself using full width most often. If you look at your document in actual size, you're scrolling horizontally and vertically all the time. **Publishing Partner**'s slow redraw sequences for a very busy document can easily interrupt your train of thought.

When you have overlapping objects, you can bring the items to front or send them to the back. Other view features let you show (or not show): rulers, text routing (you need this if you forget what columns are linked where), pictures, column outline (but you can't automatically outline created columns) and the ¼-inch background grid.

To create headers and footers, enable page numbering, and more, you can simply edit one of two "master" pages. A single master, or a left and right pair for double-sided documents are available.

Anything you place on a master page will show up on every page following it. You can create a graphics-and-text header, set page numbering (place a text object, then insert a page number from the create-layout menu), and even outline columns.

You cannot create your master column format on the master pages. You can set up columns there, but these can only be edited on the master page. Generally, your master column format will be set up under "Create Columns." If you'd like to outline text columns, for example, simply create your default columns format on the master page, then use the drawing tools to create the outlines. When they're just right, go to the object editor mode, and select and delete each column—leaving the outlines behind. This works well in place of an auto-outline feature for columns.

You can set the measuring system in picas (if you're the typesetter sort), inches or centimeters. The units you select are enabled whenever you view a document with rulers on. You'll see a ruler across the top and down the left side of the display, calibrated in the units you prefer. These come in handy when you're placing things on the page. And, if that isn't accurate enough, you can use one of **Publishing Partner**'s most powerful features, the ALT-E position editor.

When in the object editor mode, point and click on any item (it can be a column of text, a line or a **DEGAS** picture), then press ALT-E. A dialog box will come up with complete size and position information in the current units. You can point and click on any line, as in any dialog box, and press ESCAPE to clear it. You then type in a new value, specifying positioning to the nearest $1/1000$ inch. If you're a stickler for details, this feature has you covered.

When it comes to printing, **Publishing Partner** will put things on your dot-matrix printer that you've never seen there before. One of the neat tricks the program does is retain—at all times—100 percent of the information for a **DEGAS** or **NEO** picture you've placed in a document, regardless of how it's squeezed or cropped. If your printer has a higher resolution (dot density) than the display (and all of them do), the picture will come to life on the printed page.

Printing is slow, but tolerable. Since you can see exactly what you'll get on the display before printing, you won't be wasting time and paper with tests. I found **Publishing Partner** so diligent in its print processing that it doesn't check for a "stop" button press from the mouse very often. This can be maddening, since you have to sit there and peck at the mouse until you finally get its attention. At over 5 minutes a page on dot-matrix printers, this can be annoying when you do need to abort. I'd much prefer to see a simple UNDO keypress abort.

When it comes to printing, **Publishing Partner** handles very well. I created a seven-page "Ratty's Rap" column for my users' group newsletter, *with at least* one **DEGAS** picture per page. Columns were built around pictures and linked together. It printed flawlessly—meanwhile, I had time for a nice little nap. To be fair, laser printing is much faster. And I've been told by some other **Publishing Partner** addicts on Delphi that a printer spooler (like MichTron's **Soft-Spool**) cuts print time in half.

The text-importing feature of **Publishing Partner** is rather difficult to deal with. When you have your columns set up and routed, all pictures cropped and pasted in place, it would be ideal to be able to import a completed text file. But the program cannot directly load word-processor documents like those from **1st Word**.

It expects text files in a word-processor format no one else uses: namely, continuous text with carriage returns only at new paragraphs. Control codes are ignored. You can't even begin to load document files from **1st Word** (or **Word Writer**), because they don't come close to this format. If you turn off the WP mode and save your document as text, there's a carriage return at the end of every line—and each line becomes a paragraph as far as **Publishing Partner** is concerned.

I created the IMPORTER accessory to be used with **Publishing Partner** for just this purpose. SoftLogik now distributes it with the program.

IMPORTER lets you select any text file and convert it to an IMPORT.TXT file, formatted the way **Publishing Partner** likes it. Using **1st Word**, for example, just hit a double carriage return (one blank line) whenever you start a new paragraph. Save it as a text file, with the WP mode off. You can then use the IMPORTER to convert the file before bringing the text into your **Publishing Partner** document. I think you will find this preferable to generating large documents with **Publishing Partner**. It's a bit slow to be used as a general-purpose word processor. (It's fast, considering all it does, but a plain word processor lets me commit my ideas to magnetic storage much faster.) If you're inclined to do all your word processing

with **Publishing Partner**, you'll be pleased to know that it is quite **Thunder!** compatible, provided you have enough RAM for it all.

Earlier versions had a lot of bugs that could crash you completely out of the program back to the desktop—or completely into oblivion. The current revision, 1.02, is much more reliable. I haven't crashed it yet, but there are still several annoying quirks. Once you're aware of them and know how to get around them, the program is just plain fun to use.

As mentioned previously, large characters (or mid-size letters with "tall" or "wide" enabled) in actual size can "bleed" outside their predefined outlines. They'll look bizarre on the display, but will print just fine. SoftLogik has assured me that this is one of the many minor fixes for the next update.

The most annoying **Publishing Partner** feature is the "Save Preference" option. It's convenient in that it saves your preferred drive pathnames for documents, pictures and fonts. But it also saves *everything* related to text style and format. If you inadvertently enable centered text and then save preferences, **Publishing Partner** will default to centered text each time you load up.

The problem is that you want to save your configuration, so you are not constantly searching unnecessary drive paths every time you want to load a file. You just have to make sure that nothing but your most commonly preferred features are set before the save. When restored, style and format features enabled don't show up in the pull-down menus, either.

I feel it would be preferable to have a "Set Path" command. From there, a dialog would pop up prompting you for font, document or picture path settings. As it is now, the path a document is loaded from must also contain the screen font files the document uses.

I've run across many minor glitches difficult to diagnose and repeat. At one point, I went to the file menu to delete a disk file. When that file was deleted, so was my document in memory. The moral of this story: save your documents often. I still get an occasional crash during text routing. Save frequently.

Be careful with the use of folders. If the path or filename is long enough, it will overrun the dialog box that comes up to replace a file during a save, for example. Apparently, no limit checking is done on the text placed in these custom dialogs. This can really confuse your display momentarily, making it difficult to discern just what filename you're looking at, but—fortunately—it doesn't crash. This really clobbers the file select dialog, too.

**Publishing Partner** has one of the best user interfaces of any program I've used on the ST yet. There are scads of features in it; nearly all are easy to find in menus and presented in a self-explanatory way. Its text and graphics processing capabilities on the screen and printer are mind-boggling. As Tom Hudson's **DEGAS** set the standard by which all other ST graphic arts programs are judged, so does Deron Kazmaier's **Publishing Partner** define the goal for other desktop publishing applications.

The bugs in version 1.00 were easy to work around, once you knew about them. They were short lived, and SoftLogik has been very generous about getting updates to everyone who registered. I've found version 1.02 fairly bug free, fast and great fun to use. My biggest problem with the program now is that I'm addicted to it. You too may find yourself burning the midnight electrons frequently, just because **Publishing Partner** is so much fun and easy to use. This is a spectacular program, and it's not copy protected. Please buy it—don't steal it.

**Publishing Partner** has the potential to become a full-blown CAD system, given more editing features for lines and objects. More fonts would be nice, too, and are coming soon. The wish list for improvements and enhancements is a long one—not because the program lacks features, but because it inspires the user's imagination with its sophisticated, free-form approach to text and graphics manipulation, through a simple user interface. Pick up almost any Atari newsletter and you'll see how this program has already begun to revolutionize the Atarian newsletter circuit (and to render obsolete would-be typesetting programs that are merely over ST versions of **The Print Shop**).

I highly recommend **Publishing Partner**. Buy it, but be prepared to spend more time glued to the ST in fascination and a lot less time sleeping! ///

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

## Encrypt

**by Stephen Mehalek**
**STONE AGE SOFTWARE, INC.**
**P.O. Box 1216**
**Amherst, NH 03031**
**(603) 881-7689**
**High or medium resolution $19.95**

**by Charles Bachand**

*The little man came into the room through the rear of the apartment that he'd rented only a week ago. Spotting his partner next to the ST computer they'd acquired by less-than-totally-honest means, he said, "Did you get the disks?"*

*Freddy was startled. Big Al (the name had never really fit him) wasn't due back for at least two hours. Freddy was expecting anyone who came in before then to say something more in keeping with the spirit of the place—something like "This is the police," or "You're under* arrest!"

*Freddy composed himself quickly, spun his chair around and said with an unplaceable but definite accent, "Ya, I got 'em alright—but you're not going to like what's goin' on."*

*Big Al, totally in the dark as to what Freddy's harsh-sounding words implied (and being just plain ignorant to start with) replied, "What's there not to like? Ya swiped that guy's software stuff, and we're gonna pocket a bundle when we sell it to his competition. So what's the problem?"*

*Freddy knew he couldn't hold anything back from Big Al. This had something to do with "markers"—and not the magic ones that come in different colors, either. Freddy told Big Al straight out, "All the guy's files are full of junk, and none of his programs will run either, and," Freddy added, "I found this in his office." He held*

*out a disk to Big Al—a disk that he'd "acquired" along with the others.*

*Al slowly read the label on the disk out loud—he was never any good at what he considered the "social graces" of reading and writing. "It says, '**Encrypt** by Stone Age Software'. So what's dis?" (Neither was spoken English one of Big Al's strong points.)*

*"Boss, that disk is going to lose us our commission. Here, you'd better read this." Freddy handed Big Al a copy of **ST-Log** magazine—it too sort of 'came' with the disks—and Big Al slowly started reading it . . .*

**Encrypt** from Stone Age Software is not what some would describe as "exciting" stuff. It contains no fancy graphics or neat sound effects—it doesn't need them. Besides, bells and whistles of this type would only tempt you to show off the program to everyone you come in contact with, and you *really* don't want to do that. **Encrypt** is a program you'll want to use late at night when you're all alone, and *no one* is peeping over your shoulder.

**Encrypt** takes any file—ASCII as well as binary—and converts it into a completely unreadable or unrunnable mess. Without access to the password used to encrypt an **Encrypt**ed file, it's *impossible* to recover the information.

The author of the program is so confident in the encryption algorithm used that he openly invites people to try to break it. He considers such attempts as additional product testing, but with no financial compensation going to the tester. This certainly doesn't mean that there's no money to be had in attempting to crack **Encrypt**. Quite the opposite. Stone Age Software is offering a reward of $500 to the first person to break their software. They even include an encrypted test file on the program disk that you get. Just be the first person to tell them what's contained in the file called TEST_YOU.DOC and collect the money.

**Encrypt** uses "passwords" made up of up to six alphanumeric characters that act as code keys during the encryption process. If you must use **Encrypt** around a busy office, for example, you'll appreciate how the program hides the password as you type it in. This is similar to the way in which some telecommunication services shut off the "echo" mode when you enter your user password. You may select an option to display these normally hidden passwords while you type them in, but this is *not* recommended. The hidden mode is the default.

To remove the possibility of typing errors when entering passwords, **Encrypt** asks you to type a password twice. If these two entries don't match up exactly, then the program will abort the conversion process.

The actual encryption is quite fast, with the program processing about 1000 bytes per second when used with floppy disks. Users of hard disks and/or RAMdisks will find that **Encrypt** operates considerably faster in these environments.

So what do you do if, while in the process of converting a file—having answered all the questions about password and filename, and having gotten the program started—you suddenly realize that you've made a terrible mistake? You didn't *really* want to encrypt your DESKTOP. INF file, and you wish you could take it all back? Well, just press and hold down the mouse button. The whole process will reverse itself, leaving you with the original file intact. Wow, all this from a program that takes up only about 20K of RAM space!

Usually, I'm not too thrilled when a company puts their documentation out only on disk. You can obtain a printed manual from Stone Age Software, but I think charging $10.00 for 15 pages of paper is a little steep. I'd prefer to see something included, even if only some photocopies stapled together. At least I'd have documentation to read as soon as I opened the package, instead of having to wait for my printer to spit it out.

Well, anyway, that's how I *usually* feel—but **Encrypt**'s documentation resides on a disk that's separate from the one containing the program files. And, since the DOC file isn't large enough to fill that entire disk, the people at Stone Age Software have been good enough to include cut-down demos—what they call "Partialware"—of their other products. This lets you test drive software like **Lewis123** (a chemistry symbol diagramming program) and **UltraCalc** (a scientific and programming calculator) before you buy.

And, for those like Freddy and Big Al who didn't buy **Encrypt** from Stone Age Software, I have just one thing to say: "Put your hands up. You're under arrest!"

*Big Al looked up quizzically from the review he'd just read toward the corner where Freddy sat. "How?. . . Who?. . ." He didn't know what questions to ask—not to mention whom to ask them of. The sound of police breaking through the front door added to the confusion in Big Al's mind, until a question finally crystallized for him. "How could he have known where we are?" he shouted—even as the cops drew closer, guns drawn.*

*"Who do you mean?" Freddy yelled back as they were handcuffed, "The guy we ripped off? No, he couldn't possibly have known. Besides, he ain't the one who called the cops." Freddy's mouth broke into a knowing smile, "I do know who did do it, though, and . . ."*

*"Who was it?" Big Al demanded after a long wait in which he thought Freddy would volunteer the information. As soon as he'd said it, he noticed the blank badges worn by the arresting officers and heard Freddy's déjà vu reply, ". . .you're not going to like it."*

*They were slowly escorted down to the waiting cruiser—the officers, with no further dialog written for them, were silent. In the car, Freddy—his eyes widening—finally turned to Big Al and said, "We've been written into somebody's stupid software review!" The cruiser pulled out into the nighttime traffic.*

*You could tell by the look on Big Al's face that he couldn't believe what Freddy was saying, but Freddy continued anyway. "I figure this bozo thinks he's being pretty cute by writing us into this review, but we'll show him, Al. We'll show him real good!"*

*"Yeah!" chimed in Big Al, "He can't push us around. We'll . . . we'll . . ." His voice trailed off into silence, as he and Freddy saw the large sign they were fast approaching at the far end of the street. It read: THE END.* //

---

*Stephen Mehalek must have a sixth sense. Just before this issue went to press, we received a new version of **Encrypt** with—drum roll, please—printed documentation. This, of course, means that you can ignore the paragraph about DOC files.*

*Oh, I'd better describe the differences to you. If the packaging that you come across for **Encrypt** is about the size of a 5¼-inch floppy, then you've got the old version with the docs on the disk. Versions 2.0 and higher are distributed in a larger-format package, to accompany the legal-size pages.*

---

# Dollars and $ense

**STAR MONOGRAM**
**367 Van Ness Way**
**Torrance, CA 90501-1420**
**(213) 533-1190**
**All resolutions $99.95**

**by Rusel DeMaria**

For most of us, knowledge of our financial position and good record keeping are essential to prevent the monetary malignancy known as debt.

It so happens that computers are admirably constituted to perform the record keeping and reporting necessary.

**Dollars and $ense** is a venerable software package, having been around on other systems for years. The ST version inherits the basic idea from its predecessors and, to a degree, expands on it.

**Dollars and $ense** is designed to be flexible. It can be a simple checkbook manager, or it can keep complete records of your stock portfolio, credit cards, household inventory, business billing and expenses, or any combination of these.

It lets you define and maintain up to twelve separate checking accounts, and you can set up as many separate account disks as you wish. You can also extend floppy disks as many times as necessary if you run out of room for transactions. Information changed in earlier disks can be updated on extension disks.

**Dollars and $ense** can automate monthly bills, print checks, produce many kinds of reports and graphs, and play "What If?" to determine the effect of certain transactions on yearly income figures.

**Dollars and $ense** begins with booting from a special start-up disk, then using a program disk. If you don't have a second disk drive or a hard disk, you'll have to swap disks when you open your account. The program requires a formatted disk when you create a new account disk, and will default to any external or additional drive that you have active.

There's an omission in the "First Time Tutorial," which leaves out a crucial step in the development of a new account. It neglects to mention the need to create a name for the account file, so new users may get a cryptic error message. Here's the missing step:

> Before proceeding to Step 8 in the First Time Tutorial, be sure to enter a name in appropriate GEM format, e.g., *MYACCTS.DAS*. The .DAS extension is useful, though not mandatory. In future sessions, **Dollars and $ense** will automatically display all files with that extension when opening the dialog box to select an account file.

This omission and the resultant error message bring up two of my major complaints with **Dollars and $ense**. The first is that the manual is sometimes sketchy. For instance, I couldn't get the "Void check" command to work properly, and it's poorly covered in the manual.

Serious system error messages were almost never clear and descriptive. I could make mistakes at various early points in the program that either bombed it or produced unreadable error messages (though I never lost data). Even the start-up booklet with the program has warnings about potential system errors if you type too fast or open a command before a window has closed, etc. There shouldn't really *be* situations in which you can bomb a commercial program without going to excessive lengths to do so.

Calling Monogram's technical support line is less than satisfactory for two reasons: (1) it's a toll call (unless you live in the Los Angeles area); and (2) they're often busy—you may spend considerable time redialing and, when connected, even more time on hold.

Support was courteous and helpful, however. It's too bad a company with such a product hasn't installed a toll-free line, or more operators. *[With the recent news of Monogram's intent to merge with Star, we may hope service will change for the better.]*

Once past initial problems, the program ran smoothly. You can use one of three preset account lists (Household, Business and Tax), or you can define all accounts from scratch.

You can create five budget types—Asset, Checking, Expense, Income and Liability—and can have up to 120 accounts per file.

For each account, you can define budgets (fixed and variable) and the starting balance. You would set a starting balance for a checking account, loan account or credit card. You would set a fixed budget for regular payments—like your mortgage—or a variable budget for payments that change from month to month—like heating bills. You can leave out budgets and starting balances if you wish (although you'll need one for any checking accounts and credit cards).

One of the most interesting features is the "Net Annual Budget" display. As you enter each income, expense and liability account, the Net Annual Budget at the top of the screen changes to reflect the information. Once you've filled in all the data, you can see if you're showing a profit or a loss. You can also test the effect of new purchases by creating a budget for, say, a house you want to buy, and testing the effect of the payments on your Net Annual Budget. If it goes below zero, the house isn't such a good idea just now.

After defining accounts (you can go back and add or modify account information if you need to), fill in pertinent information about each checking account (name, address, account number, etc.) Then it's on to transaction entry. . .

**Dollars and $ense** has shortcuts to help you enter transactions (such as automatically filling in sequential check numbers, allowing you to use letter keys for your auto-teller or non-check withdrawals, and more).

It's simple to assign a distribution account to a transaction. If the check was for groceries, you want it in the "Groceries" account. To distribute it properly, type the first letter or so and press TAB. **Dollars and $ense** will automatically find the closest match. If you don't see the account you want, press the right or left arrow keys to scroll the list.

An alternative way to enter a distribution account is to select "Show Accounts" (ALT-N) from the maintenance menu and select from the special account window that appears.

When you enter a transaction amount **Dollars and $ense** automatically defaults to the appropriate column (check or deposit). You can split accounts or transactions, too. If you don't balance the total check against the total of the distribution accounts, you'll get an (intelligible) error message, telling you how much you're out of balance. When you're finished entering the transaction, tab to begin a new transaction line.

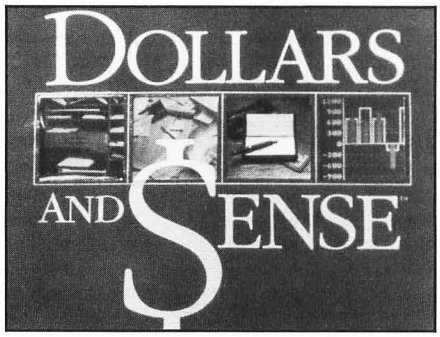
At almost every step of the entry pro-

cess, you're presented with a default value, or a method of speeding up the entry process and saving keystrokes. My only complaint about the entry process—and this recurs throughout this and other ST programs—is that you have to press ESCAPE to clear fields before entering data. I consider this an unnecessary step.

On the plus side, you can modify any transaction, moving to any field in any transaction at any time. The total account balance changes immediately to reflect any changes you make.

**Dollars and $ense** processes all new data in batches. This means entry is fast—there's no disk access. It also means your new data is vulnerable until saved. Choose "End Edit" from the edit menu to complete processing and save all new and modified transactions.

Another timesaving device is the "Automatic Transaction." You can define up to twenty-five automated transaction sets, with up to one hundred transactions in each one. Use this for your regular monthly bills. All you have to enter is the current date; the program takes care of the rest. (The results can be modified, if necessary.)



Dollars and $ense

When you've entered transactions, you can print checks (using preprinted forms you've ordered). Name and address information on each check is printed to display in a window envelope.

You can also reconcile your bank statements with a procedure that's fast and helpful. It shows at a glance outstanding amounts, and any discrepancy.

**Dollars and $ense** lets you review and modify past transactions—up to ninety at a time—using a combination of parameters (date range, check number range, partial string search in the description field, account name, transaction type, cleared or not, tax or non-tax). The same parameters are used to select information for a "Transaction Report."

You can send any report to screen or printer. In fact, you can have several reports and/or graphs on-screen at once, sizing and moving windows with standard GEM techniques.

Other reports are available: Account Year-To-Date Summary, Monthly Budget Totals, Monthly Actual Totals, Income Statement, Balance Sheet and Cash Flow Analysis. Most can be produced for annual, YTD, monthly or quarterly data.

Graphs are produced for composites—special groupings of accounts. The program provides four default composites, or you can create your own composites. Account Contribution to Totals is a pie chart; the rest are bar charts.

Although reports are versatile and excellent, their basic form can't be altered. You can't create a custom report showing only the data you want.

You'll really appreciate **Dollars and $ense** at tax time. The reports it prints out can cut preparation time tremendously.

I've used **Dollars and $ense** to keep my accounts current, but sometimes fall behind. My greatest appreciation of the program comes from the fact that you can enter so many transactions so quickly. With other programs I've tried, it took so long to enter each transaction that it was hard to catch up.

I was disappointed with a few aspects of the program. Having previously used it on an IBM PC, I missed full color usage on the Atari version. Except for brief flashes of the default desktop color, the program is in black and white. Considering the colorful nature of many ST programs, I found this a bit disappointing.

Overall, I still found **Dollars and $ense** a fine financial program. Some features are excellent—the Net Annual Budget, the multiple reports and graphs, and the keystroke-saving features.

Given its strengths and its features, I would recommend **Dollars and $ense** as one of the best home finance management programs if Monogram would stabilize it. Also, given the potential for problems, I think Monogram should beef up user support, as suggested earlier.

I'm a little more reserved about recommending **Dollars and $ense** for business situations. If your business is small, **Dollars and $ense** may be more than adequate. As a writer, I can easily use it to keep all income and expense figures. Other businesses may need more specialized or more complete packages.

**Dollars and $ense** is great—if you're careful. None of the crashes caused any loss of data, just of patience—and they usually required doing something that could be easily avoided. So **Dollars and $ense** *is* a good buy, but it *could* be a better one. ///

---

*Rusel DeMaria is a computer consultant and free-lance writer. He writes a computer column for the* ***Mauii News****, and has worked with computers for over ten years. He is also Vice President of Andromeda Micro Corporation and co-SYSOP of the Royal Hawaiian Software RBBS—(808) 244-9789.*

---


# Follow the bouncing blitter.

by D.F. Scott

*It is the missing piece of hardware which could very well make the Atari ST the fastest microcomputer on the open market. It might also become the biggest embarrassment for Atari Corp. since the 1450XLD. For a year and a half, we've been promised the blitter chip, the hardware that makes it possible to transfer several successive bits of data as one raster, from one area of memory to another all at once. During that period of time, the projected release date has been continually pushed forward; as of this writing, it is projected for September 1st.*

*Our primary source for this* ***Status report*** *is Greg Pratt, Atari's Chief Financial Officer. Although he's had little or nothing to do with design of the blitter component itself, two decisions made through him will affect not only the blitter's future, but the entire Atari production system. Time restrictions prevented us from speaking, at the last minute, to Atari Service Division Chief Randy Hain; therefore, information from that department and from Research and Engineering was relayed to us by Marketing Communications Director Neil Harris.*

**How the blitter works—this week.**

From the latest Atari corporate documentation, here's how the blitter enhancement will work . . .

The main graphics routines in the ST are called "Line-A" routines. These are used by all of TOS, including GEM, to produce lines or filled objects, control the mouse pointer and other graphic "sprite" objects, and copy rasters from one region of memory to another—normally referred to as a "software blit," since it's achieved by means of code, not hardware.

An assembly language program, for instance, will call these Line-A routines through a cleverly designed trap—not a *trap* instruction *per se*, but a 68000 "unimplemented instruction exception" that passes control temporarily to a subroutine. This exception is triggered by declaring in program memory, using the opcode *DC.W*, a word-length value (2 bytes) between hex values $A000 and $A00E, inclusive. It is from the high-order byte in these cases that the Line-A routines get their name.

It is the Line-A routine set, according to Atari documents, that will be carved in silicon, for an operating system speed increase approaching 50 percent and a subsequent speed increase in all software using GEM or Line-A directly, especially those programs which rely on Line-A for blitting.

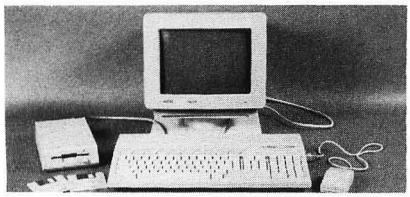
There have been numerous complaints from developers who protest that the blitter is ineffective until accessed directly. Direct access of the blitter using the new TOS, however, will be identical in nature to the relatively slow internal software blit, through a word declaration of $A00E and manipulation of the proper input/output arrays—so the degree of protest may soon diminish.

The actual degree of performance enhancement can be measured, says Harris, by placing a call to the GEM Virtual Device Interface—namely opcode number 102, vq_extnd. The system output variable intout[6] should then contain the number of 16-word (32-byte) raster blits per second. Without the blitter, this value should be only *1000*. Harris maintains that the reason developers haven't seen—or foreseen—much speed enhancement with the blitter is because they're using the old TOS.

The new six-ROM-chip TOS set will be sold with the blitter component, and perhaps also separately; however, the form of the blitter package has not yet been decided. Some (few) 1040s were equipped with a free slot for the blitter, but most existing STs are not "blitter-ready." The packages under consideration are: (1) piggybacking the bare chip onto the 68000 CPU, then soldering in place—which can result in a sporadic overall system failure; (2) offering the chip on a "daughter board," although some STs may not have room for one; and (3) as Neil Harris reports, actually *swapping mother boards* for a completely new board.

Whatever the decision, Harris claims,

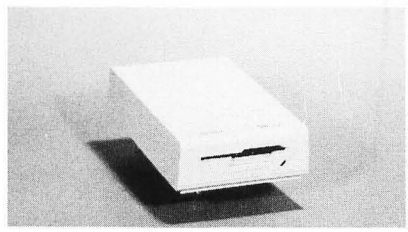
# WHEN YOU WANT TO TALK ATARI...YOU WANT TO TALK TO US!



## ATARI 520 System

**520 Monochrome**
No. AA520S..... **$499.00**

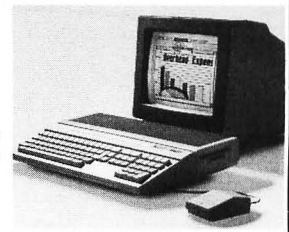
**520 Color/RGB**
No. AA520SC..... **$759.00**



## ATARI 354 Disk Drive

No. AA354..... **$109.00**

## ATARI 1040 System



**1040 Monochrome**
No. AA51040M..... **$719.00**

**1040 RGB/Color**
No. AA51040C..... **$869.00**

## CALL FOR PRICING ON THE NEW ATARI COMPONENTS

**Access**
Leaderboard ..... 24.99
**Activision**
Hacker II..... 31.99
Music Studio..... 34.99
**Antic**
Cad 3-D..... 33.99
**Atari**
ST Project..... call
Star Raiders..... 19.99
Microsoft Word ST..... call
**Batteries Included**
Degas Elite..... 48.99
Paperclip Elite..... 79.99
Thunder..... 25.99
**Cygnus**
Starfleet I..... 35.99
**Ditek**
STA Accounts..... 149.00
**Electronic Arts**
Chessmaster 2000..... 32.99

**Epyx**
Championship Wrestling..... 24.99
Winter Games..... 24.99
**First Byte**
Math Talk (Educational)..... 33.99
**Haba**
Habawriter ..... 31.99
**Hippotamus**
Hippobackgammon..... 27.99
**Infocom**
Hitchhiker's Guide..... 26.99
Zork Trilogy..... 41.99
**Michtron**
Major Motion..... 25.99
Karate Kid..... 26.99
**Microprose**
Silent Service..... 25.99
F-15 Strike Eagle..... 25.99
**OSS**
Personal Pascal..... 48.99

**Paradox**
Wanderer 3-D..... 28.99
**Polarware**
Crimson Crown..... 26.99
**Psygnosis**
Arena ..... 25.99
Deep Space..... 32.99
**Sierra-on-Line**
Winnie the Pooh (Educational)... 17.99
**Strategic Simulations**
Phantasie..... 25.99
**Sublogic**
Flight Simulator II..... 36.99
Jet..... 36.99
**Timeworks**
Swiftcalc Spreadsheet..... 49.99
Wordwriter ..... 49.99
**VIP**
Professional (Gem)..... 144.00
**X-Lent**
Typesetter..... 27.99

## PERIPHERALS

**Atari Components**
1040 Computer (No monitor).... 589.00
SM124 Mono Monitor..... 159.00
SM1224 RGB/Color Mono..... 319.00
XM804 Printer..... 189.00
Laser Printer..... call

**Modems**
Atari XM301..... 44.99
Atari SX212 (300/1200)..... 99.99
Anchor AN520..... 129.99
Supra 1200ST..... 139.00

**Drives**
Atari 314 DS/DD..... 209.00
Atari 354 SS/DD..... 109.00
Atari SHD204 20 Meg. ..... 589.00
Supra 20 Meg..... 559.00
Supra 30 Meg..... 779.00

## ACCESSORIES

**Maxell**
3½" SS/DD (10)..... 12.99
3½" DS/DD (10)..... 18.99
**Sony**
3½" SS/DD (10)..... 12.99
3½" DS/DD (10)..... 19.99

**Curtis**
Safety Strip..... 19.99
Diamond ..... 29.99
Emerald ..... 39.99
Ruby..... 54.99
Universal Printer Stand..... 14.99

**Flip 'n File**
FF280 3½" (5)..... 3.99
FF737 3½" (25)..... 9.99
**Kalmar Designs (Teakwood)**
Keyboard Caddy Drawer..... 54.99
CI 452 3½" Disk Holder..... 14.99

**COMPUTER MAIL ORDER**

**In the U.S.A. and Canada**

## Call toll-free 1-800-233-8950

**Outside the U.S.A. & Canada 717 327-9575 Telex 5106017898**
**Educational Institutions call toll-free 1-800-221-4283**
**CMO. 477 East Third Street, Dept. B805, 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 subject to change and all items are subject to availability. Defective software will be replaced with the same item only. Hardware will be repaired or replaced 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 #118 ON READER SERVICE CARD

the total suggested retail cost of the upgrade should not exceed $120. Official ST service establishments, however, have been told by Atari representatives to expect a cost exceeding $300.

**The third operating system.**

Here's a complete list of changes that have been made to TOS for the new edition:

—The previous VDI was unable to plot a graphic arc below a certain number of degrees; low-degree arcs now plot correctly.

—The RS-232 handler has been completely rewritten, so that Request-To-Send/Clear-To-Send "handshaking," and 50- and 75-baud modes now work.

—The use of the ESCAPE key to send commands to the VT-52 emulator now works.

—The routine used at boot-up to clear user memory has been speeded up considerably. This was especially necessary for use with the Mega STs, which have an internal RAM capacity of 4 Mb.

—The "Save Desktop" and the "Print Screen" options from the desktop menu now utilize a dialog box for confirmation.

—The ROM cartridge handler has been rewritten to allow the execution of .TOS and .TTP programs from cartridge.

—The arrows along the frames of GEM windows, when selected with the left mouse button *held down*, will scroll the window contents continuously without repetitive clicks.

—The bug which would inadvertently save a character to disk on the DESKTOP.INF file has been fixed.

—The Direct Memory Access device handler has been rewritten so that more than one device may be recognized.

—The use of underscore (_) characters in the Item Selector Box no longer causes inadvertent system crashes.

—The disk read-with-verify routine has been modified so that Cyclic Redundancy Checks are no longer overlooked; i.e., the verify *actually verifies*.

—Blitter engagement/disengagement is now an option available from the desktop options menu.

—Documents output to the screen or printer from the desktop will now display characters, including accented characters, going above ASCII code 127. A larger memory buffer has been apportioned for desktop "Shows"; and Show is now the default button in the dialog box, not Cancel.

—A larger memory buffer has also been apportioned for use with disk copies with single disk drives, so fewer swaps are required.

—The mouse-click handler has been reworked so that single clicks show up *as* single clicks at all times.

—BIOS function 3, bconout, will now write single characters to devices faster.

—The disk formatting routine selected from the desktop has been replaced for a faster formatter, though still using 40/80 tracks at 9 sectors per track.

—Certain system variables that were officially undocumented by Atari (and bore warnings against being used) have been moved or deleted. As a result, some applications that used these "illegal" variables may no longer work.

Research and Engineering Vice President Shiraz Shivji told us three months ago that the blitter shipment was being held up *not* because of an incomplete design, but because the supplies necessary to assemble them had not yet been delivered. Had those supplies been shipped, said Shivji, the blitter would have been off the assembly line in a matter of weeks.

With that revelation, we come rather abruptly to the matter of $75 million in 15-year international bonds, known formally as "convertible subordinate debentures," registered with the U.S. Securities and Exchange Commission, consummated on April 29, and offered to the public by Atari Corp. What does Atari want with $75 million, why does it want the money so soon, and what in the world has this to do with the blitter chip? The answers take us, in a sense, on a world tour.

**Champagne wishes, silicon dreams.**

In 1984, Jay Miner, designer of the Atari 800, was negotiating with the new Atari Corp. for sale of the production rights of his newly designed computer called the Amiga. Just when it seemed Miner and Atari would strike another deal, Commodore legitimately outbid Jack Tramiel.

But Commodore had more in its favor than a bigger name and more money. It had its own semiconductor manufacturing plant, established by Jack Tramiel in his Commodore days. In fact, in the first two months of 1981, Commodore sold $9 million in custom-designed semiconductor parts to its main customer—Atari, then under CEO Raymond Kassar. So at a time when, on paper, the Atari/Commo-

# GREAT SOFTWARE
## from across the pond

Leisure Software — ATARI ST ™ — Professional Software and Utilities.

**ST Karate★ $29.95**

The ultimate martial arts game for the Atari ST. 1 or 2 player options with multiple opponents, digitized sound and unbelievable graphics. Joystick control gives 18 possible super-fast movements. A must for the serious games player.

**New METROPOLIS★ $24.95**

A fast action shoot-em-up with high speed lazer canons, smart bombs and 360 movement. Steer your craft around the amazing and colourful city ridding it of the usual barrage of hi-tech aliens. The one way streets, crossfire and road blocks will ensure you don't get through....well maybe!.

**WANDERER★**

**3D Stereo $39.95**

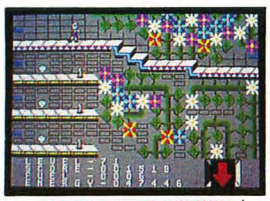
Unbelievable! The first ST arcade in full stereo 3D! Live out a fantastic interplanetary adventure experiencing amazing 3D sensations with the special glasses supplied with the game. Wanderer takes you on a high speed mission through space, with the action punctuated only by rounds of amazing intergalatic poker!. A definite must!.

**FIRE BLASTER★**
**WAR ZONE★**

**midi music 2 Disks**

**$34.95**

Amazing value! Two disk pack with two full colour ST arcade games! Fireblaster is the ultimate shoot-em-up with whole screen movement, multiple ships and over 35 levels. War Zone is a futuristic tank simulation with multiple features. An amazing pack for shoot-em-up fans.

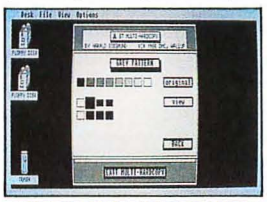


**ST PROTECTOR★**
**SPACE STATION★**

**midi music 2 Disks**

**$34.95**

Another super value two disk, two game pack! Space Station is a colourful and fantastic ST platform game with layer upon layer of hidden depths. Your task of cleaning up the station is helped by your ability to shoot, jet pack and teleport to new and undiscovered areas. Protector is a full colour "defender" style game with multiple screens and fast action.



**FLASH CACHE**
**FLASH BAK**

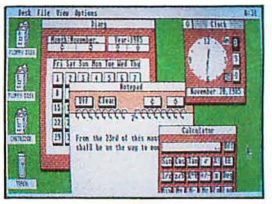
**$79.95 GEM**

This is one of the best hard disk utility packages available. Flash cache offers selective ram disk caching to increase your hard disk performance by up to 50%. Also included is a complete solution to the GEM 40 folder bug and the facility to password protect a disk partition. Flash-Bak is a fully featured hard disk backup utility with compression, encryption, wildcards and many auto disk features. Flash-Bak offers real performance with an amazing 1MB+ per minute backup to floppy disk! Full fact sheet available.

**PRO SOUND DESIGNER $99**

**Includes 8 bit sound digitizer mono version GEM**

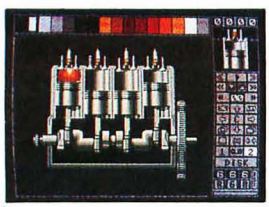
Quite simply the ultimate in sound control on your ST! A colourful VCR style interface ensures that anyone can use this program and be creating fantastic sounds within minutes. Full control over the AY-3-8910 sound chip is provided (with support for your own programs) as well as a superb 8 bit digitizer that allows you to playback samples from within your own programs. The sound sample editor is a dream! Demo disk available for $5.



**BACK Pack**

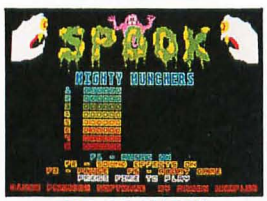
**$79.95 GEM ROM**

Back-Pack is an ST ROM containing nine powerful desk accessories without using up RAM. Back-Pack contains a Printer Buffer, Address Book, Mini Clock, RAM disk, Diary, Notepad, Typewriter, Alarm Clock and a powerful scientific calculator. A full fact sheet is available for this amazing program.



**PRO SPRITE DESIGNER $59.95**

This fully featured sprite and animation designer is the easiest and fastest way to create animations on the ST. Control routines are included for Basic, C and Assembly language. This is quite simply one of the best animation designers available. If you don't believe us send $5 and we will send you an unbelievable demo!



**THE EIDERSOFT EXTRAVAGANZA**

**$19.95**

This two disk set contains two ST arcade games* including a super 3D pac man called Spook, demos, desktops and utility programs. It comes in a super shrink wrap pack with posters, fact sheets, stickers and a host of other goodies. Excellent value!!

**mini comm**

**$39.95 GEM**

Mini Comm is a complete communications package as a GEM desktop accessory! Mini Comm is fully featured and supports most protocols. Mini Comm has a telephone book, macro language, powerful redialer, call cost calculator and a host of other features. The real power of Mini Comm is its ability to work in the background, which includes the facility to do **background downloading!** Full fact sheet available.

**Tempus**

**$49.95**

**The Professional Editor GEM**

Tempus is the ultimate text editor for programmers. Up to four windows of text can be manipulated at high speed. Support is given for structured programming layout and the package comes complete with a fully fledged programmers calculator, among a host of other features. Tempus is one of the fastest text editors available (c1000 search and replace per second) and is extremely comfortable to use. We can provide a non refundable try-before-you-buy version for $7.50.

To order or request product information please telephone

**1-412-947-3739**

VISA MasterCard

or write to the address below.

**Eidersoft USA Inc**
**PO Box 288**
**Burgettstown PA 15021**

PA residents add 6% sales tax.

EIDERSOFT
T·R·I·A·N·G·L·E
SOFT INNOVATION
Paradox

Shipping Info: Continental US free: Add $3 for COD
Paying Info: Credit Card (no surcharge), cashier check or money order.

★ colour only. GEM is the trademark of Digital Research Inc. ST and Atari are the trademarks of the Atari Corporation.

**DEALERS/DISTRIBUTORS WE NEED YOU**
Please telephone for dealer pack and full information.

Note we only sell high quality ST software that is supplied on single sided diskettes and is compatible with all 520/1040 ST computers. Mega ST compatibility please telephone.

dore competition seemed a bitter feud, the two were actually dependent upon each other.

Six years and $75 million in bonds later, Atari finds itself in a position where it can no longer rely on Commodore—or anyone else—for parts and supplies. In a dispute which cost Commodore's CEO Thomas J. Rattigan his job, the chairman and new CEO, Irving Gould, moved his Amiga unit to Europe. Whether or not ST production follows the Amiga to Europe, $75 million is about to send it in that general direction.

Dunn & Bradstreet's *Moody's Industrial*, a twice-weekly investors' publication evaluating individual corporate statistics, reported that Atari intends to use its bond-earned money for "capital expenditures and acquisitions." We asked Chief Financial Officer Greg Pratt just what he intends to acquire.

Pratt responded, "We are looking to integrate vertically, and that could include companies which supply peripheral-type products or semiconductor-type products to us."

He states, $75 million "is not a lot of money for a semiconductor company, or any other kind of company that's going to produce significant peripherals for us. . . What we're trying to find is companies that can complement our manufacturing process, or secure peripheral products. While we don't have anybody specifically identified at this point, there are a lot of companies out there looking to merge with stronger companies."

It's obvious that dependence on other manufacturers for semiconductor parts, including TTL chips, has put Atari in the agonizing position of being behind in its schedule, and falling behind those of its competitors. The bond offering, with its 5¼-percent interest rate per annum, is clearly a move to make Atari more self-sufficient—a move necessitated by the ever-slowing development of that single, gremlinlike little wafer.

We promised a world tour, and now we deliver. We begin in the Far East—namely Japan—which has recently been involved in a U.S. trade dispute over the alleged dumping of electronic goods into U.S. markets at below production costs, thus placing U.S. manufacturers in a noncompetitive position at home.

We asked Greg Pratt if the U.S. imposition of tariffs against Japanese imports in retaliation would affect Atari's pricing structure in any way. "It seems there has been some agreement," answered Pratt, "on the part of the Japanese, to live up to certain commitments they've made." If Japan backs up its words with like actions, says Pratt, "we won't be severely affected. If they continue to do the kinds of things that are a problem for the U.S., then . . . in the face of uncertainty, it's going to push prices up generally all over."

Pratt reports that it is Atari's intention to expand closer to home—maybe in Mexico, maybe in Canada, but possibly just across the street from corporate headquarters. Says Pratt, "We're *not* going to expand in the Far East. We're definitely going to expand closer to where our markets are, basically in the U.S. and Europe. The decision is that we'll build here first and, after that, in Europe. . .We think that investing in the States just makes sense at this time. Of course, the protectionist atmosphere that's starting to develop in Congress and in various segments of industry, and our decision [in tandem] are probably just coincidental.

"In order to compete effectively against the Japanese, you have to think *globally*. You have to buy and sell in the right places. . .to move not only from region to region, but within regions. So, basically, what's going on in Korea and Japan and Taiwan and Hong Kong and Singapore and Malaysia and the Phillipines, you have to know all at the same time from the supply side—as well as what's going on in the other countries on the other side.

"To say that Atari," continues Pratt, "is wrapped up in any particular single country would be somewhat misleading. We are an international company; we try to be good corporate citizens in each country that we do business with, and we do think and look at our computer business on a global basis.

"You're going to find that every company operating in each country will try to wrap itself in that individual country's flag. . .I think if you insulate yourself, and think that you're only American, or only British, or only German, and are not aware of what's going on in other countries, you won't be in business very long. There's so much going on, and we're all connected via modems, and information is flowing so quickly back and forth around the world. It's one world."

Still, the major portion of that world,

for Atari, happens to be Europe. States Pratt, "If you look at any of our financials, you'll see that Europe has been, and continues to be, our strongest contributor, [with] about two-thirds of our sales."

The reason? "It's very simple. We've tried to go after the European market and not yet begun to aggressively go after the United States market. You have to realize that, when we acquired the company, we had a tremendous debt—a tremendous amount of issues to be worked out in the States. We had a comparatively small amount of money and brand new products we were trying to launch—we were trying to do a lot of things at once. It made much more sense to try to approach markets that were smaller in size, markets we could effectively tackle, saving the United States for last.

"Eventually, we'd like to see our sales split fifty-fifty, but America is such a large market; it takes so much money to get anyone's attention from an advertising point of view. The geographic territory is so *huge* compared to [Europe], that we just didn't have the fuel, if you will—the dollars to attack everybody at once. [In Europe], we didn't have entrenched competitors like Apple and IBM; they don't have the kind of dominance there that they have here in our category. The people there are more receptive to new products; they aren't as label oriented as people are here. Atari. . .didn't have a large negative blanket, because the name didn't connote thoughts of losses and things of the magnitude [seen] here in the States."

Has the inability to meet production schedules, we asked Pratt, only managed to prolong the endurance of that bad image back in the U.S.? Is Atari, as some critics say, incurably slow? "I can tell you my sales wouldn't be moving the way they are if that was a widely held view. There may be some squeaky wheels around. . . The bottom line is that a person goes out and votes by spending his dollars on my machine."

With that vote of confidence, we conclude our global tour of this frantic-paced, philosophy-filled industry, courtesy of a sly slab of silicon called the blitter chip, which American ingenuity may yet make it possible for us to own. That's the **Status report** for this month. I'll see you on Delphi. ///

## Reader comment

*continued from page 8*

mat. Having the MC68000 reversing the byte order during disk read/write is ludicrous. Leave the IBM disk format to an IBM emulator. (When will that be released, in the 21st century?)

Finally, ANALOG Publishing should be more critical of Atari Corp. Atari has made it very difficult for ST owners to learn about their machines. Charging $300 for a development package is outrageous. The software that comes with the kit has serious bugs. The documentation is horribly organized and written. If it's Atari's intention to keep people ignorant of their machines, they have succeeded. If the software out in the market is pretty bad, one can blame Atari. Atari enjoys delivering products at very late dates. Granted, Atari is a small company, but still, they should be able to announce the correct release date.

These are just some of the areas **ANALOG Computing** and **ST-Log** only lightly touched. If you wish Atari well, the magazine should inform the Atari community from an unbiased view. I'm sure Jack will respond if a million letters are dumped on his desk.

I would also like to comment on one of the statements in a previous issue of **ANALOG Computing**. The statement claimed it would not be feasible for Atari to release a NS32000 series computer in the near future. I do not believe that is so. A higher-end NS32000 series (20 MHz NS32332 or NS32532) machine—or better yet, a workstation—can be built with an MC68000 co-processor (ST emulation). Such a product will challenge both the MC68020 and the Intel 80386 market. A properly designed NS32000 series machine will, of course, outperform the other machines. John Dvorak, former editor of *Infoworld*, along with several other electronics magazines predicted the NS32000 series would power the next generation of workstations.

Sincerely,
Raymond Chin
Chelsea, MA

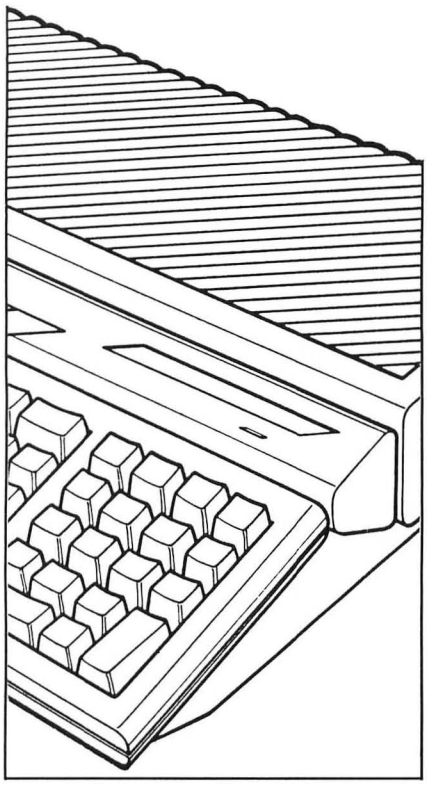
*However one feels about the Atari Developer's Kit, there are now alternatives to it—many of them.*

*GEM does have its problems, but it's here, and it works, for the most part. Recent history shows just how hard it is to write this kind of system—look at IBM's experience with Topview, or how long it took Microsoft to "finish" Windows.*

*On paper, the NS32000 series certainly seems to have many features that the 68000 lacks. But, until recently, that's the only place the 32000 existed—on paper. The 68000 has been around quite a while now, which means that the bugs have been worked out, and manufacturers like Atari can be assured of a steady supply of good chips. It's hard to understand why Jack Tramiel should have felt obligated to beta test an unproven chip in the Atari make-or-break ST series, just for the sake of some fancy microcode. As for virtual memory, this is available with the 68010 chip, which it's reasonable to expect will be introduced into the ST series at some point.*

*The programmer's manual for the 32000 is just becoming commercially available, and it makes interesting reading. What it doesn't make clear, among other things, is that floating-point operations are just as much an option with the NS series as they are with the 68000 or 80286 chips. You have to read very closely to discover that all the floating-point instructions listed in the body of the book can only be used with the floating-point chip installed. At least Motorola's and Intel's documentation is up front about this.*

*The 68000 may not "support" high-level languages as thoroughly as one could imagine, but it does support them, with instructions such as* link, movem, *and others. The compiler-oriented instructions in the NS32000 manual look impressive. Experience teaches one to check the clock speeds of such complex instructions—they can end up being slower than a whole block of code that does the same thing on a different processor. However, we could not find clock times listed in the NS32000 manual. Somehow one doubts that this means the NS32000 is so fast that it just can't be timed.* —Ed.



ST user

# THIS MONTH:
# A little black box and some talk of PCs.

*Arthur Leyenberger is a human factors psychologist and free-lance writer living in New Jersey. He's been writing about computers for four years and continues to be an Atari enthusiast. When not computing, he enjoys playing with robotic toys.*

*CompuServe — 71266,46*
*Delphi — NJANALOG*

**by Arthur Leyenberger**

Chances are good that, if you own an Atari ST—any ST—you probably also own an 8-bit Atari computer. Of course, there are no statistics regarding this, but, based on talking with people in my own user group and users all over the country, I would guess that at least half of all ST users are previous 8-bitters.

No doubt this was especially true during the first year or so of the ST's existence, when all you could do with the machine was open and close GEM desktop windows, and attempt to figure out how to use Logo. If you own an ST and you *don't* own an 8-bit Atari, then you most likely own some other home computer—Commodore 64, VIC-20 or even (egads!) a TI-99.

Ownership of two (or more) computers has its problems. You may want to share certain peripherals (such as printers and modems), but it's a hassle to continually connect and reconnect cables among the various pieces of hardware. One solution to this problem is to use switch boxes (also called T-switches, A-B boxes or black boxes). The Mercedes of switch box companies is Black Box Corporation of Pittsburgh.

Black Box Corp. has dozens and dozens of products that will make your computer life easier. The majority of these are hardware switch boxes, like a switch to let two computers talk to the same printer (ABC-25 Switch). Another useful model allows two computers to communicate through the same modem.

I had a couple of special switch needs that Black Box was able to fulfill. For example, I wanted to use my daisy-wheel printer and dot-matrix printer with both my ST and AT&T 6300. I found just the switch I needed in the Black Box catalog: an X-switch. This device let me hook up both printers and both computers, simultaneously, to the two-position switch box. In one position, the ST was connected to the daisy-wheel, while the PC was connected to the dot-matrix printer. Turning the knob connected the ST to the dot-matrix and the PC to the daisy-wheel printer—a simple and elegant solution to *that* switching problem.

Another example illustrates the helpful nature of these folks. I wanted to use both my computers with one modem, but also wanted to connect the computers, so that I could send files from one machine to the other. For a while, I was using a simple RS-232 A-B switch box and would disconnect both computers from the box when I wanted to hook the machines together. (Of course, I would have to install a null modem between the two computers.)

If you don't know what a null modem is, allow me to digress for a minute. A modem is simply a device that lets a computer communicate via a phone line to another computer, using a modem on the other end. At any given time when the two computers are communicating, one of the machines is the transmitting device

and the other is the receiving device. I like to use the dancing analogy to explain this: when two people are dancing, one of them has to lead and the other, follow. If both partners try to lead at the same time, communication ceases to exist.

This is (loosely) what happens with computers. The modem on each end of the communication path allows one computer to lead, then the other, etc. When two computers are connected directly, they cannot transmit and receive properly unless they're wired correctly. A null modem reverses pins 2 and 3, and 11 and 12 so that the computers can understand each other. By the way, you can make your own null modem by changing the pins to the above-described wiring on one side of a cable. Enough of the digression; onward.

After briefly describing my modem switching need and my kludge solution to a technical representative on the phone, I was told I needed switch ABC-AB-25. This product would let me leave both computers and the modem continually connected to the three-position switch box. I then could connect either of the two computers to the modem or to each other, without using a separate null modem. Once again, a Black Box product satisfied my needs.

Any data-switching product ordered from Black Box has a 30-day return guarantee. If it doesn't meet your needs, you can send it back within 30 days for a full refund. If you use their phone support line before you buy, chances are you'll get the appropriate product for your needs the first time.

Black Box also sells tools, test equipment, books and a variety of other data communication products. These are all explained in their catalog. It contains a wealth of information and includes a glossary of most of the terms used. A separate, 88-page glossary can be ordered for $2, directly from the company.

Although you may be able to find assorted flavors of A-B boxes—and some other products—for a little less money, Black Box is the only company I know of that provides support before *and* after the sale. They have the most complete line of data communication and computer devices in the industry (this is all they do—no chairs, ribbons or floppy disks for sale here), provide telephone support and publish a catalog, complete with periodic updates. Next time you need a black box, make sure it's a Black Box.

From years of experience as a user, I can tell you their products are reliable and function exactly as advertised. For information, contact: Black Box Corporation, P.O. Box 12800, Pittsburgh, PA 15241 — (412) 746-5530.

**IBM begets more IBM.**

IBM recently announced a new line of personal computers, six years after it standardized the computer industry with what is now the most common computer found in the business office. Called the Personal System/2, these four computers were long awaited by some PC users and the general computer press. However, it's largely felt that the actual machines released by IBM fell short of the mark.

You may be wondering why, in an Atari-specific magazine devoted to the powerful 68000-based ST computers, I'm bothering to discuss the mechanical prodigies of Big Blue. You'll recall Atari announced two MS-DOS computers (read: PC clones) at the Winter Consumer Electronics Show in January. Since Atari's entered the MS-DOS market, it's appropriate that **ST-Log** cover these machines. Of course, once the Atari PCs become widely available, we will cover them in greater depth.

In any case, any action by a competitor in the computer industry—which may influence Atari, its products or its users—is proper material for this column.

The other reason for discussing IBM's newest machines is the compatibility (no news breaks here, read on) with ST computers. As you may already know, the ST disk drives share the same ASCII file structure with existing IBM and clone computers. In fact, I have written about, and have been using for six months, a 5¼-inch disk drive with my ST, called the Byte-1. ASCII files written by the ST on a 5¼-inch disk can be read by my MS-DOS computer when the disk is inserted into the PC's drive. Likewise, PC-written ASCII files can be read by the ST on the Byte-1 drive.

All of the new IBM computers use 3½-inch disk drives. One is a 720K byte (double-sided, double-density) disk drive—similar, I suspect, to that used by the ST. The other three computers use 1.44-megabyte drives (double sided, quad density) and should be able to read the 720K disks. I have yet to attempt to read ST disks on the new IBMs or vice versa, although I have no reason to believe that they aren't compatible.

If they are, think of the implications: after using my new IBM PC (or similar new clone) at the office, I can bring just the disk home and continue word processing on my ST, using ASCII files. Also, **VIP Professional** can read Lotus 1-2-3 files, so I can use Lotus at the office and **VIP** at home. Now do you see why it's important to discuss the new IBM PCs?

One of the new IBMs uses an 8086 processor, has a 20-megabyte hard disk, 640K memory and three expansion slots. The 8086 is a faster processor than the 8088, which is used in the original IBM PC. Most PC clones have been using the 8086 for years; they generally offer a 20-megabyte hard disk and come with 640K of memory. Two use the relatively new 80286 processor, and the most powerful—and also the most expensive—model uses the 80386 microprocessor.

One feature that sets these new IBM computers apart is their high-resolution graphics. In addition, they all use Microsoft Windows for managing application programs. In the MS-DOS world, Microsoft Windows is a major competitor to Digital Research's GEM. In fact, Windows has been winning the contest for almost a year—even without the help of having IBM specify it as part of their new computer line.

The top of the new IBM line of computers will require a new operating system that's being written by Microsoft, to be ready in about a year. Although new software will have to await this operating system, the machines will be able to use current MS-DOS software. In addition to the operating system, the new IBM PSs (Personal Systems) use a large percentage of proprietary VLSI (Very Large Scale Integration) chips. This will slow down the clone makers by a couple of years.

The impact of the new IBMs on the Atari ST user community is uncertain. One foreseeable effect may be to lessen the demand for the Atari PC. On the other hand, many users could find that there's no reason to pay more, even for a more powerful IBM computer, or to wait another year (or more) for more powerful software. If these potential users have a need for an MS-DOS computer, it could be that the Atari PC will fulfill their needs. ///

# Step 1
## LOOSE ENDS

**The fifth in our series for first-time computer users.**

by Maurice Molyneaux

None of my outlines for future **Step 1** pieces seemed appropriate at this time. All relate to specific topics that are a bit more advanced than I think many of you are ready to tackle just yet. Since there remains a lot of information about ST programs, utilities and operations that should be covered before moving on, this month and next I'll try to tie up as many "loose ends" as I can. This issue: disks, RAMdisks, etc. Next month, we'll tackle what's left.

One other point: in this column I've tried to be as precise as possible. However, what I say isn't always carved in stone. Many programmers have found ways to make the ST system do things it isn't normally supposed to, using "creative" programming methods. So when I say you can't do such and such, it's a generalization that applies in *most* cases. If you happen upon a program that lets you do something I said the ST couldn't, please don't think I've mislead you. To go over every possible exception would mean we'd only cover a single small topic per issue. By the time we got through the basics, your ST would be as old and quaint as an Altair 8080 is now. (A bit of microcomputer history: the Altair was a build-it-yourself kit computer, starting with 256 bytes (¼K!) of RAM, back in the Neolithic days of computing.)

### Flawed GEM.

In **Step 1 — Soft Wares** (issue 12), the section called "The case of the missing TOS" stated that "The first STs had to load their operating system (GEM) from disk. . ." *What?* I cried on reading this; GEM isn't the OS, TOS is! How could I have been so stu. . .I gritted my teeth and loaded the original text file to see the error for myself—and it wasn't there. "It's not my fault!" I cried, stabbing a finger at the magazine's staff box, "Somebody *there* did it!"

This kind of problem is bound to crop up in readying an article for print, so I blame no one. But the misstatement appeared under my name, and I want to be sure **Step 1** readers know what is *really* happening in the ST.

As in issue 11's **Step 1**, TOS (The Operating System) gives a "generic" computer instructions on how to be an Atari ST. The first thing TOS does is run a program called GEM (Graphics Environment Manager), which handles the windows, menu bars and mouse—through which the user can communicate with the computer. For a beginning user, what matters is that the two are integral to your ST's identity—its function and its style.

### Slipped disk.

Tid-bytes about STs and disks. . . First: people have a tendency to call the 3½-inch disks used by the ST "hard disks," because of their rigid plastic shells. This phraseology is incorrect. If you were to open the casing of one (I said "if"—don't try it!), under its strong shell you'd find a weak, flimsy floppy disk. The shells and neat metal doors make them *look* different from what most people know as a floppy disk, but a floppy by any other name is still a floppy.

Second: one thing that surprises a lot of new users is seeing another user pull a disk out of the drive while a program's still running. They can't understand how the program continues to work without the disk.

This isn't as silly as it may sound. Your cassette deck, turntable or CD player doesn't do diddly without its tape, record or CD. And remember the video game systems which only work when a cartridge is plugged in? Given this kind of background, it's not surprising that many people would think a disk must remain in the drive for a program to work. Of course, this isn't the case.

Tapes, records and compact disks all work on the same principle: a mechanism in the player reads data from the media and plays or displays it. The device is constantly reading (and sometimes writing)—or it's really doing nothing but spinning its figurative wheels.

A computer's disks are different. The computer scans the disk, finds the sections of data it wants, reads them in bulk, stores the data in its memory, then stops accessing the disk. When this happens, you can play air hockey with the disk for all the computer cares—because it has *taken* what it needs. This is not an omnipresent absolute truth, however. Some programs, while they don't constantly read the disk like a stereo does an album, do go back to read/write to/from the disk now and again. Removing the disk in such circumstances may cause a problem. With the ST, it's usually easy enough to tell when you should have left the disk in, because you'll see one of those cute dialog boxes stating something like *Drive A is not responding. . .*

This applies to floppy disks, but never, *never, never!* to hardware devices like cartridges and interface cables. Pulling them out while a program is running—or your ST is just *on*—might not only bomb whatever application you're running, but damage your ST and its peripherals, too!

Third, a common problem: somehow, you keep losing data on your disks. Files disappear, or you accidentally format a master disk. The sensible solution: *before using* almost any software, write-protect the original or master disk. This is done by pushing the little square tab on the bottom of the disk so that the "window" it covers is open. When the window is open, the disk (theoretically) cannot be formatted, nor any of the files on it erased.

Do this to all your master disks before using them. Then format some blank disks and copy the masters onto them. Once you've made copies, store the master disks in a safe place—use the copies to work with. This way, if you somehow manage to kill a disk (or some of its files), the original is safely tucked away, to be recopied over the disk you accidentally messed up.

If you lose a backup, make another. But *never* use the original if you can avoid it. Disks are cheap; there's no excuse for not making backups. Even better, make backups of all disks with programs or data you don't want to risk losing. I have two backups of the disk on which these articles go—just in case!

If you have a disk you can't copy and the system just seems to go nuts or do nothing when you try to copy it, you're probably trying to duplicate copy-protected software. What this means is that the software company has used a method to make the disk generally impossible to copy by normal means. In such a case, find out from the documentation or the manufacturer if you can obtain backup disks from them directly. If not, there are programs available for the express purpose of making copies of protected programs.

If you decide to purchase or already own one of these, remember the following: you are legally allowed to make backup copies of commercial software for your own use, but distributing copies to others is a violation of copyright laws, is a criminal offense, and hurts *you* in the long run. The more free copies given out, the less programs will be sold. The less programs sold, the smaller the chances that a company will make any new programs for your computer. No excuses are acceptable. Don't give or take copies. If your excuse is "But I wouldn't have bought it anyway," don't bother saying it. Stealing is stealing, regardless of whether it's shoplifting or pirating software. Don't do it!

Last: sometimes, if you try to copy a disk with a format that the GEMDOS (GEM Disk Operating System) copy functions don't support (meaning you can't copy it from the desktop normally)—such as a disk with 82 tracks instead of the usual 80—you may also need special duplicating software to back it up (many are in the public domain, by the way). For more information on disk formats, see **Step 1** in issue 12 of **ST-Log**.

### Disk free space.

More on disks. First, if you want to know how much free space remains on a given disk, use "Show Info" under the desktop's file menu and not the "X bytes used in X items" flag at the top of a window. This is because this "Information Line" lists only bytes used in the current directory; it doesn't display the total amount of bytes used if your disk has any folders on it.

If you're in the main directory, the line displays the bytes used there; if in a folder, the number of bytes there; but not *all* the bytes used on the disk. If you take the Information Line's word, then you may suddenly find GEM telling you *This disk doesn't have enough room for this operation.*

It has happened more than once that, when someone takes a look at the total number of bytes used by files on a given disk, they assume they have *X* amount left over. For example, the other day I looked at my boot disk—which is double-sided and formatted for fast-read, with 82 tracks at 10 sectors per track, yielding over 800K of storage space—and saw that my files consumed about 690K. Great, I thought, I've still got about 120K left. Much to my chagrin, using Show Info from the desktop rudely informed me I only had about 50K left. *What?* I lost 70K somewhere! Had I mistakenly formatted the disk for the ST double-sided default (720K disk)? No, I couldn't have done that. Now where did that extra storage space go?

Determined to get to the bottom of this, I began looking through all the subdirectories (folders) on my disk. Sure enough, in one folder I found sixty-four small data files, each listed at 74 bytes apiece. Well, 74x64=4376 bytes—about 4K by that reckoning. But appearances can be deceiving, because the minimum space taken by any file on an ST disk is two physical disk sectors of 512 bytes apiece, better known as 1024 bytes, or 1K. So each file took 1024 bytes instead of 74, and those total 4376 bytes of data were in fact eating some 64K of disk space! The additional missing K were accounted for by other small files on the disk.

The moral of this story? Never assume the list of bytes used on a given disk corresponds verbatim to the actual amount of space used. It's usually a little more—in cases like this, where you have scores of tiny files (printer data, fonts, etc.), usually a lot more. So always use Show Info (file menu) and read the "Bytes Available" when checking to see how much room there really is on your disk.

### RAMdisk zone.

Wait! You can't put those memory chips in your floppy drive's slot! What are you doing? Huh? Making a RAMdisk? No, no, no! Putting RAM into a disk drive isn't what *makes* a RAMdisk!

Silly as it may seem, the preceding is not all that far removed from the concept some new users have of what a ramdisk is. Their ideas on the subject are often just a half-step removed from the Twilight Zone, and in that spirit. . .

"Presented for your consideration: an Atari ST computer with a single floppy drive and no hard disk. The beings who programmed GEM for this machine were obviously sadistic, forcing users to swap disks *ad infinitum* when copying files on a single-drive system. The result: confusion, mistaken disks, deleted files and sudden hair loss. The solution: a South Seas vacation, a second disk drive, or. . .a trip to the RAMdisk Zone."

Okay, Rod Serling I'm not, but at least it let me hint at one of the more common reasons for beginners to use a RAMdisk: file copying on a single-drive system.

If you have a single floppy drive and no hard disk, you've no doubt experienced the frustration of having to swap "disk A" and "disk B" multiple times for each file you attempted to copy from one disk to another. It doesn't matter if the file to be copied is only 1K (1 kilobyte) in length; the system will *still* require multiple disk swaps. This whole affair might seem unimportant if it wasn't a fact that a 520ST (with TOS in ROM) can copy an entire single-sided disk in one pass. Insert the source disk once, insert the destination disk once, and bingo! But copy just files, and disk swap will follow swap after swap. . .

The point I'm getting at: the whole procedure is not only ludicrous but downright dangerous, because it's easy to stick the wrong disk in the drive when prompted for a swap. And that action can lead to disaster. (Disaster can take the form of scrambled files, messed up disk directories, etc.) Thus, if you can't afford—or don't want—a second floppy drive or a hard disk, your salvation may appear in the form of a RAMdisk.

A RAMdisk on the ST is not (usually) a hardware attachment, but rather a mere program. When run, it will convince your ST to treat a block of RAM as a disk drive. Once this is done, you can copy files from a real drive to the RAMdisk, then copy them back from the RAMdisk to the drive, into which you will have inserted another disk. Look ma, one swap!

The biggest problem with a RAMdisk: it's volatile. With a real disk you can save your work, pop the disk out and turn off your computer. With a RAMdisk, the moment you turn off your computer, the RAMdisk ceases to exist. Its contents go to meet their maker in electron heaven. It's *important* to remember to copy any and all needed files from a RAMdisk to a floppy or hard disk before shutting off or resetting your ST. Otherwise, they're history!

On to cases. Most RAMdisks for the ST are created by desk accessories (more on these next time), although a number of them are also set up by .PRG or .TOS programs which can be run from the desktop or the AUTO folder. (See **Step 1** in **ST-Log** 12 for more on the AUTO folder.) All work in basically the same manner, making the ST "pretend" a block of its memory is a disk drive. And all are fast, because moving data around in RAM is much easier and quicker than magnetically coding data on and reading it from real disks. However, RAMdisks differ in key areas. . .

**Memory.**

Most RAMdisk accessories or programs have preset values for the size (kilobyte capacity) of the RAMdisk. You have to be careful in choosing such predefined RAMdisks, because an accessory designed to set up a 720K RAMdisk won't be of much use on a 512K 520ST. Some RAMdisk programs (like **M-Disk** from MichTron) will tell you how much memory you have available for a RAMdisk, then let you select the amount you want.

Warning: many ST RAMdisks will be unusable if made too small. The smallest RAMdisk size you should try, if you have a program which lets you set size, is about 100K. Furthermore, if you have a program which tells you exactly how much free (unused) memory your system has available, *don't* use every bit of it for your RAMdisk! Always leave at least 100K of free RAM for programs to run in. If you use all your memory for the RAMdisk, your system won't have any RAM left for anything else! (Most good RAMdisk programs which let you set their size will tell you the maximum size RAMdisk you can have—and that number is usually lower than the maximum available RAM, to leave some memory in which other programs can run.)

**Stability.**

Most RAMdisks are very fragile. Switching resolutions, pressing RESET or having a program bomb—all are enough to blast most RAMdisks in a vital area, crippling or destroying them and all their contents. There are some RAMdisks designed to survive a reset, but they're few and far between.

Most RAMdisks can't be removed without turning off your computer. And don't think using the "Remove" function under "Install Drive" in the desktop's options menu will do it, because all *that* does is remove the icon for the drive from the desktop. One exception to this is the Micro-Time **RAM-Buffer**. It's a desk accessory that lets the user define the size of and install printer buffers (more on this later) and RAMdisks which can later be removed with the click of a mouse button.

To further confuse things for the beginner, while pressing RESET may kill a RAMdisk in memory, the reset may not necessarily "remove the body." The RAM set aside for the RAMdisk may still be locked off, even though the RAMdisk itself is dead, buried and pushing up daisies.

The only *sure* way to terminate a RAMdisk is to turn off your ST for 5 to 10 seconds. (I recommend 15 seconds on a 1040, because their internal power supply seems to hold a charge that takes longer to bleed.) If you want to keep your current RAMdisk and it's not reset-proof, don't change resolutions or press RESET or turn the ST off. . .or else!

**Unreal drive.**

The problem with a RAMdisk is that it isn't *really* a disk drive. Therefore, you can't carry out true disk copies on a RAMdisk. It usually just doesn't conform, byte for byte, to a real disk. There are a few RAMdisks that emulate actual floppy disks to the point that disk copies are possible, but these are rare. The usual use of a RAMdisk is for file copy—not disk copy—functions.

**Installation.**

Most RAMdisks have to be "installed," particularly the

first time you use one. This involves telling GEM to place a drive icon for the RAMdisk on the desktop. You'll have to specify the RAMdisk's identifying letter (as in drive *C* or *D*) and label (like *400K RAMdisk*). Both of these parameters are entered using the "Install Drive" selection under the options menu on the GEM desktop. (For more on installing drives, consult your ST manual, or see the **Step 1** article in April's **ST-Log**.) Fortunately, most RAMdisks come with simple documentation telling you how to set up and use them. That should get you started.

Now, you owners of systems with two floppy disks and/or hard drives are probably smug in thinking you don't need a RAMdisk. Fine. You don't need one. But that doesn't mean using one couldn't *benefit* you.

If you're involved in telecommunications and spend a lot of time on-line (using a modem, of course), a RAMdisk could save you time and money. Since RAMdisks are extremely fast, they can store and retrieve data more quickly than can a floppy or even a hard disk. If you're paying an on-line fee or long-distance rates (or both) the extra seconds and/or minutes you save by uploading and downloading files via RAMdisk could add up to a decent amount of money in the long run. (For more on this, modems and money, read **A Baudy Tale** in **ST-Log** 13 and **Modem Max** in **ANALOG Computing** 53.)

Just make sure the RAMdisk you use is compatible with your telecommunications software. Little is more annoying than downloading a bunch of *good stuff* to a RAMdisk—then having the system crash when you leave your terminal program, because it and the RAMdisk don't get along!

Furthermore, if you're interested in programming in a language which uses compilers and linkers (like C), you could save a lot of time (and disk wear and tear) by copying all your language's associated compiling and linking files and libraries to a RAMdisk—along with your source code—and compiling them there. The difference: seconds on the RAMdisk versus minutes with a floppy!

This also applies to using any program which does a great deal of disk access. Squeezing, unsqueezing, archiving and de-archiving files is more easily and quickly carried out from a RAMdisk, too. (We'll cover these file compression topics soon.)

Finally, there are a few RAMdisks on the market that are actually hardware attachments for your ST. They contain RAM that acts as a RAMdisk, so you can have one without losing sizable amounts of your ST's internal RAM to set one up. Some such attachments have their own power supplies, which means their contents can survive even if you turn your ST off!

### Buffer zone.

In the previous section I mentioned printer buffers, also known as *print spoolers*. Though there are hardware spool/buffers on the market, most of you will encounter the software variety. Much like a RAMdisk, a spooler program/accessory sets aside a chunk of RAM to act as a "buffer" between your computer and its printer.

When you decide to print something, the system "prints" the data to the buffer's RAM, from which it can then trickle out through a port to your printer. Since the average dot-matrix printer prints at only about 120 characters per second maximum, when your ST is printing it's spending most of its time twiddling its figurative thumbs—tied up with the task of slowly feeding information to the printer.

By "printing" (in essence, sending all the print codes) into a block of RAM, the computer can complete *its* end of the printing job very quickly, to be free for your use again while the printer slowly sucks up the information from the spooler. Thus, using a spooler, your printer could merrily churn out a report while you quit your word processor and play a nice, relaxing game of **Star Raiders**.

A few warnings: turning off or resetting your computer with a spooler in action will cause the buffer's contents to be lost. As in the case of RAMdisks, the RAM consumed by most spoolers can only be freed by turning off your ST.

Finally, make sure what you want to print is correct before letting it go through the spooler. If your formatting or such is wrong, a "pause print" command will be of no use when your printer's on page two and your buffer holds thirty more! Turning your printer off and on will only delay the inevitable power-off or resetting of your ST to abort the print.

(A note: earlier, I mentioned the Micro-Time **RAM Buffer**. I just recently got hold of one and want to warn some of you about a peculiar thing. When I set up a printer buffer, quite often when I print, the buffer sucks up the data, but my SG-10 printer does nothing. However, if I flip the printer off then back on, it takes off and starts printing from the buffer. I don't know if this is a problem unique to Star printers, but if you have the same "no print" problem while using Micro-Time's **RAM Buffer**, try the power off/on trick yourself. One other thing: **RAM Buffer** allows you to abort the printing process from the accessory itself.)

You can buy hardware printer buffers for your ST. These consist of a box with its own RAM that goes between your ST and printer (on the cable). Often, they have their own power supply, so you can finish printing, turn off your ST—and leave the buffer box and your printer to merrily go about their business.

### Farewell, but not good-bye.

Mercy! In order to get this down to a reasonable size, I had to whack a good 10K-worth of text off the end. I didn't realize when I started there was so much to cover on loose ends. Anyway, what was cut this issue will appear next time (with some more goodies as well). Topics will include desk accessories, file compression, controller compatibility, the truth about ST RAM and, of course, even more. Until next time. . .

---

*Allergic to all things Commodore and never bitten by Apples, Maurice Molyneaux first purchased an Atari 800XL for animation work, but upgraded to an ST as soon as they became available. Currently slaving to complete the fifteenth draft of a science-fiction novel, he also masochistically churns out free-lance articles, artwork and animation on his ST, and hopes to dig out of a mountain of pending projects by the year 2000.*

---

# Database Delphi

## ST matters discussed in the Atari Users' Group SIG on Delphi.

by Matthew J.W. Ratcliff

Charles Johnson gives us some more programming tips for the ST this month. I have a summary of the fascinating GDOS conference for you, and a few details on the latest files in the ST Programs Database. Next month, I'll give you a brief look at the April conferences with Brian Moriarty (INFOCOMINC) about Infocom games, and on desktop publishing and the ST with Deron Kazmaier, author of **Publishing Partner**.

### Mailbag.

The following question about memory allocation and the ST's screen memory came to me via EMAIL:

From: MOCKO (Mike Duppong)
To: MATRAT (Matthew J.W. Ratcliff)

Can you give me an easy way to ensure that screen memory starts on a 512-byte boundary? I have a secondary screen in RAM, and I don't know exactly where to load it. I'm using malloc with an argument of 32512 bytes. When I get the address back, I just increment it and divide by *512* until the remainder is *0*, then take that as the address. Is there a better way?

Another question—if I have a screenful of information just *anywhere* in memory, can I use that as the source for a raster operation, with the current screen as the destination? I know you *can* do that, but does the secondary screen (the source) necessarily have to start on a 512-byte boundary, or just anything that has to be displayed as a physical screen?

Not being up to speed with my Megamax C compiler just yet, I let Charles Johnson field this question. Here's his reply:

From: CFJ (Charles F. Johnson)
To: MOCKO,MATRAT

The method you described, of setting your screen to a 512-byte boundary, will work just about as well as any other. Since a program could end up anywhere in memory, you're forced to use a method like this.

Yes, a source raster for a copy-raster operation can be located anywhere in memory. Screen memory actually should start on a 256-byte—not on a 512-byte—boundary, by the way. And the boundary issue only applies to screens that *will be* displayed.

[*Editor's note: a more efficient way to align a screen pointer is:*

```
ptr = (ptr + 0xffL)
      & (0xffL);
```

*. . .where* ptr *is the pointer (here treated as a long integer). These bitwise operations should generate less object code than the increment/divide scheme. The above statement "ands" out the bottom 8 bits of* ptr*—since this is the same as rounding* ptr *down, the* 255 *is added before the operation, to compensate. (You don't want to go below the memory area allocated!)]*

From my EMAIL bag, I've been informed that John Tal (username JOHNTAL), a programmer for MichTron, will not be representing that company on Delphi. We hope to have a GFA BASIC conference with John and others from his company, soon.

### Forum messages—ST Vertical Blank Interrupts.

CFJ continued his tutoring of MOCKO—and of me and anyone else interested in the programming intricacies of the ST—in the Forum, beginning with information on VBI processing.

From: CFJ
To: MOCKO

I've been playing around with the vertical blank for a couple of days, and have some information for you. First, it's a very simple thing to install your own routine into the list of routines executed during the vertical blank. Address $456 (vbl queue) contains a pointer to a table which holds the addresses of up to eight possible VBI routines. (I checked this table with SID, and there's usually only one entry there—the system routine. But the best method would be to search for the first *0* longword in the table.) Just stuff the address of your routine in the first available slot, and that's it. Your routine begins executing on the next VBI. [*For examples, see* ***ST Color Palette*** *in the January 1986* ***ANALOG Computing****, pages 72-78, or* ***Money*** *in the February 1987 issue of* ***ST-Log****, pages 55-56.]*

Your interrupt routine should *not* call any GEMDOS, BIOS or XBIOS functions, because these take too much time. If you have to do those kinds of tasks, the best route is to use hardware registers (if you can). Your VBI routine should end with an RTS.

Also, remember that the vertical blank occurs with different frequency on the color (60Hz) and monochrome (70Hz) monitors.

You can use any registers you want in the routine. The OS saves everything for you, except the user stack pointer. Also, you're in the supervisor mode when your VBI routine is executing; you don't have to enter the super mode to change any protected memory locations.

From: MOCKO
To: CFJ

Don't you have to enter supervisor mode *first*, before you can start messing around with, say, vblqueue? To get into supervisor mode you have to do a trap, right?

From: CFJ
To: MOCKO

You have two ways to get into supervisor mode to access the vblqueue variables. You can use the GEMDOS Super call ($20) or the Xbios Supexec() call (38). The Abacus documentation is fairly clear on the operation of these calls.

**GDOS conference.**

The GDOS conference was a big success. I would like to thank Tom Hudson (**DEGAS, DEGAS Elite, CAD-3D**), Dan Moore (**PaperClip, PaperClip Elite ST**) and Frank Cohen (**Regent Base**) for the time they took from their busy schedules to share their views of GDOS with us. An edited transcript is now in the Reviews and News database. Below is one user's response to the conference and my reply (summation of the conference):

From: STARTWO (Doyle Helms)
To: MATRAT

I just downloaded and read the conference text. I'm a little depressed about what I read. Though quite satisfied with the ST and the GEM interface, I'm surprised at the amount of great software for the ST if it's so difficult to program for. Thanks for being honest in the conference about all the shortcomings of the OS. I'm beginning to program for the ST and didn't realize the problems I was in for.

From: MATRAT
To: STARTWO

Everyone wants a "GDOS standard," which gives you a *standard* way of printing text and graphics, a *standard* way of loading and displaying different fonts, and so on. This means that Batteries Included and Broderbund won't *both* have to invent their own printer drivers, font drivers, etc.

A common program developed by Atari that would do all that dirty work for you would be ideal. The problem is that GDOS eats RAM like you would popcorn at a movie. It can easily consume 1 meg of RAM for fonts and a good printer driver in a word processing application, even though GDOS itself is only an 8K program! A different *font file* must be loaded for each point size of the same font you wish to use. If you want three fonts, and three point sizes each, that's nine font files you must load at boot time. **Publishing Partner** supports up to seventy-two point sizes—but without GDOS's help—from a single 4K screen font file!

GDOS has a lot of bugs. Tom Hudson has confirmed that *most* of the bugs reported on **DEGAS Elite** can be directly attributed to GDOS glitches. No one likes GDOS in its present form, but no one has a good alternative, either.

So what does it mean? Wait for Atari to get a good finalized GDOS out the door (as Batteries Included is doing relative to **PaperClip Elite**), or develop your own custom drivers (as has Deron Kazmaier, author of **Publishing Partner** from SoftLogik).

As for the TOS ROM: yes, it has lots of bugs. No program that takes 192K of ROM is going to be *extremely* bug free. The biggest problems are realizing the mistakes in the documentation and working around them. Glitches in the Atari Developer's Kit documentation have been propagated throughout *all* other publications made since. Once you know the documentation errors and work-arounds, things get a lot easier. It takes time to learn, but, once you have the fix routines, it's pretty easy to deal with—especially if you're programming for yourself or for publications.

But if you have to turn out a copyrighted product, to be sold to the masses, then you must have a bug-free product—even if that means compensating for bugs in the operating system. That *is* the hard part; it's what makes programming for Tom Hudson, Dan Moore, Frank Cohen, and other developers so difficult. If they turn out a product that will crash due to a bug in the ST ROMs and *not* in their software, who do you think will get blamed? One guess.

**New ST database files.**

YELVINGTON (Steve Yelvington) has sent up a new version of MicroEMACS, revision 2.16. This revision was customized by Dale Schumacher and John Stanley of Minneapolis/St. Paul.

It features mouse control of the cursor and a cut-and-paste buffer. Rev. 2.16 has a fast screen update (an assembly language module handles writing to the console), extensive on-line help, an internal command line interpreter for .TOS or .TTP files, and extensive use of the function keys. This ARCed file includes the program source code, also.

CTCHANG (Clarence Chang) has sent up the latest version of Simon Poole's Uniterm, 1.7A. It emulates a VT200 terminal, and supports YMODEM and KERMIT file transfers.

IMPORT2 is an accessory sent up by yours truly. It is a *must* for all **Publishing Partner** users. The original version of my importer accessory is currently being shipped with revision 1.01 of **Publishing Partner**, which will convert any text file into a pseudo-word-processor file format necessary to import *neatly* into your documents.

IMPORT2 will now allow foreign characters (greater than ASCII 127) to pass through the importer's conversion of a text file, if you so desire. (Version 1.02 of **Publishing Partner** supports some foreign characters.) Also, if you ever load a **DEGAS** picture (such as ComputerEyes pictures) and it's a *negative*, IMPORT2 can convert it to a *positive* IMPORT **DEGAS** picture file for you. The AssemPro source code is available for downloading also.

**Delphi user tip.**

A quick way to find someone's real name is to use the ENTRY command at the ANALOG prompt. *ENTRY ANALOG2* will reveal that this is SYSOP Charles Bachand and not ANALOG3 (Publisher Lee Pappas), or ANALOG4 (Technical Editor Clayton Walnum). While in conference, the /ENTRY command will perform the same. //

# GFA Basics

## By popular demand— a how-to column on the new BASIC.

by Ian Chadwick

This is the first of a new programming column about GFA BASIC, an ST language written by GFA Systemtechnik in Germany, and distributed by MichTron in North America. GFA BASIC is a different type of BASIC from that you may be accustomed to; it is a "structured" BASIC. By that, I mean that its design encourages—and, in some cases, forces—programming structures similar to those found in languages like C, Pascal and Modula-2. This column is an overview. In subsequent issues, I'll explore techniques, features, tricks and traps you should know about when writing your own code.

GFA BASIC is the first language I've been really comfortable with on the ST. The entry method is through a straightforward and easily learned text editor, which does a syntax check on your code as you write it—favorably reminiscent of BASIC on the 8-bit Atari. It also has test routines you can use before running your program—if a mistake is encountered during execution, you're returned to the offending area of code, so you don't need to hunt for it. I give the program the "Chadwick Seal of Approval."

Some important differences between GFA and other BASICs are:

(1) It doesn't use line numbers.

(2) Only one command is allowed on a line.

(3) The editor and the program execution screens are separate and can be different resolutions simultaneously.

(4) The editor screen is a TOS—not GEM—screen, so it scrolls and redraws quickly. However, you can write GEM windows into the program screen.

(5) It has a runtime module that runs a program but won't let it be edited. This module may be freely distributed, so you can give your working code away to someone who doesn't yet own the program.

(6) A separate two-pass compiler is available, which turns your BASIC code into fast stand-alone 68000 code. I highly recommend it: a compiler makes programming in BASIC a viable proposition for writing commercial code.

(7) It has many powerful commands and features not available in other BASICs, including direct access to GEMDOS, BIOS, XBIOS, AES and VDI functions.

It's also a lot of fun to work in; for the first time since I received my ST, I'm actually enjoying programming. It's not easy to make GFA crash, or to create a situation where the CTRL-SHIFT-ALT key combination won't bring you back to the editor. On the flip side, the documentation is, to be kind, abominable. MichTron is, however, rewriting it and coming out with a new, corrected and updated manual. I don't recommend waiting to buy the program—you can get the new manual later, through their upgrade policy.

Finally, to clear up some confusion about numbers and versions that occurs in the manual, the U.S. release is version 1 for this continent, but it's actually version 2 for Germany. (Their original version didn't get released here.) A second U.S. version to be released will support the blitter chip when it comes out. I'm told this is the only difference between it and the existing version.

If you're accustomed to other BASICs, you'll probably find nothing wrong with a line like this:

```
100 Z=C:IF Z>B% THEN GOTO 199:
A=A/ABS(D-E):GOSUB 1000:B%=B%+
1:TEMP=F:F=Q:Q=TEMP
```

For this, you'd need code at Line 199 for the GOTO and a subroutine ending with RETURN which began at Line 1000. (Don't worry if your current BASIC code doesn't quite

look like this; it's generic—it doesn't actually do anything—it's just an example!) In GFA BASIC, you'd write the same code like this:

```
Z = C
If Z>B%
    Goto Add_up
    Div A,ABS(D-E)
    Gosub Set_max
    Inc B%
    Swap F,Q
Endif
```

Again, you need two more areas of code—first a segment labeled for the Goto—that is, its first line would be Add_up: (a label ending with a colon). This is the unconditional jump. The Gosub also needs code to jump to, this time, Procedure Set_max. The word *procedure* defines the segment as a subroutine. It also must end with a Return statement.

Notice the similarity in elementary statements like Z=C. Nothing unusual there. But there are three commands evident that make programming considerably simpler and execution faster. Inc, for increment, simply adds 1 to the variable named—the same as B%=B%+1. There is also a corresponding Dec command to subtract 1 from the variable.



Div, for divide, divides the first variable by the second and is the same as A=A/ABS(D–E) above. Similar commands exist for Add, Mul and Sub (add, multiply and subtract). Swap exchanges the two variables named directly, without the convolutions of a third temporary variable some BASICs require.

You can also see that the If statement requires an Endif, which makes it a "compound statement." If routines can be nested, used with logical operators like And and Or, and can include an Else statement. The Then is optional. Also note the indenting—done automatically by GFA, and a good visual guide to loops, nests and routines.

Making the switch to GFA from other BASICs is therefore not difficult—most of the commands and statements are similar. What you need to learn, aside from the differences, is how to create structured code—you can't simply hop into and out of code blocks and routines. You need to structure the program logically into coherent segments—a practice that will help if you decide to move on to other structured languages.

But don't think of GFA BASIC as simply a precursor to another language like C. It's quite adequate for commercial coding efforts, although it lacks some of the powerful features in C or Pascal. These can be simulated well enough, and output from programs like the Resource Construction Set, (aimed at integration with C programs) can be used with GFA without too much trouble. However, for this sort of work, you should have the Atari Developer's Kit or the Abacus GEM and ST internals books, in order to understand what you're doing. The GFA manual, at least in the existing version, is weakest in its discussion of these aspects of the language.

And, as a last word, there are some prime examples of code written in GFA BASIC on CompuServe in the Atari16 SIG. I'm told there are even more on GEnie, and you'll be able to find programs written for **ST-Log** and code examples used in this column on Delphi. If you have access to any of these services, you should download some examples and experiment with them—it's the easiest way to learn the language.

Of course, I have some sample code for you even in this first column—two simple routines you can incorporate into your own programs. One loads an uncompressed **DEGAS**-format picture to the screen; the other saves the screen as a **DEGAS**-format picture. There are a lot of little bits in these routines I won't go into until later, but the code is commented, so it should be easy to follow. The **DEGAS** file structure is documented in the **DEGAS** and **DEGAS Elite** manuals.

Because of its flexibility, GFA also offers more than one way to perform a task—something we'll explore in later columns. You'll see an alternative method of loading a file in one of the routines. Either method works for this example.

Until next time—happy coding! ///

Listing 1.
GFA BASIC listing.

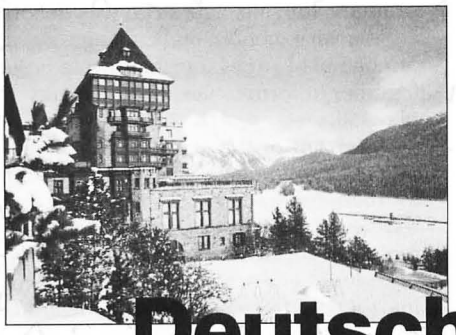
```
' Degas format screen saver:
' by Ian Chadwick for ST Log
' This routine saves a screen as an uncompressed Degas format file
' first, get the screen resolution
Res=Xbios(4)
' next let's create a simple screen we can save
Gosub Test_data
' a little loop - tell the program to save the screen when
' we press the left mouse button
' so we can put a disk in the drive first
Do
  Mouse A,B,C
   If C=1
    Gosub Pic_print
    End
   Endif
Loop
' here's where we do the actual save:
Procedure Pic_print
' first save the screen as a 32,000 byte string
  Sget Screen1$
  ' now create the file on the disk
  ' change the drive and name to suit your system
  ' and extender to suit the res
  Open "O",#1,"A:DEGASPIC.PI2"
  ' the first two bytes are the resolution info
  Print #1;Chr$(0);
  Print #1;Chr$(Res);
  ' next get the palette information and save it
  For X=0 To 15
    C_num1=Xbios(7,X,-1)
    ' clear out the top byte before we do anything else
    Cnum=C_num1 And &H777
    ' break the data into two parts - a word, rather than a byte
    Ch=Int(Cnum/256)
    C_l=Cnum-(Ch*256)
    ' now write the two bytes to disk together
    Print #1;Chr$(Ch);Chr$(C_l);
  Next X
  ' save the screen string to the file
  Print #1;Screen1$
  ' finally save the animation information - all zeroes
  For X=0 To 31
    Print #1;Chr$(0);
  Next X
  Close #1
Return
' here's the screen we want to save - write a character to
' a portion of the display area, but you can easily
' replace these lines with your own routine
Procedure Test_data
  Cls
  For N=25 To 76 Step 2
    For Y=1 To 24
      Print At(N,Y);Chr$(237)
    Next Y
  Next N
Return
```

Listing 2.
GFA BASIC listing.

```
' Degas format screen loader:
' by Ian Chadwick for ST Log
'  This routine loads a Degas format file into your screen area
' First get the screen resolution
Res=Xbios(4)
' Now open a channel to read in the picture
```

```
Open "I",#1,"A:DEGASPIC.PI2"
' change the drive and name as necessary
' change the .PIn extension for the resolution you want - PI1, PI2 or PI3
' now get the first two bytes (DEGAS resolution information
' see the Degas or Degas Elite manual)
Y=Inp(#1)
X=Inp(#1)
' make sure the resolution matches the file
' give an alert box if not
If X<>Res
  Alert 1," Wrong resolution ",1,"okay",Dummy
  Stop
Endif
' if the Res is okay, load in the palette info
' and store it in the ST palette block as defined in Xbios(7)
For N=0 To 15
  X=Inp(#1)
  Y=Inp(#1)
  Z=X*256+Y
  C=Xbios(7,N,Z)
Next N
' we ignore the animation bytes that follow the screen data
' there are two ways to read in the picture. here's the first:
' move the head to the start of the actual picture
Seek #1,34
' now get the screen data - 32,000 bytes sequentially
Bget #1,Xbios(2),32000
' and close the file
Close #1
' the second method is done by replacing the lines from the Seek command
' and on with those below (don't forget to remove the apostrophe!):
' Close #1
' Bload "A:DEGASPIC.PI2",Xbios(2)-34
' this performs a binary load into the screen address at Xbios(2) minus
' 34 bytes - which are used by Degas for resolution and palette info.
' the animation info is loaded, but is beyond the screen area
' make sure you're not using the memory directly adjacent to the
' screen RAM if you use this technique!
' now wait for the left mouse button press to end
Do
  Mouse A,B,C
  If C=1
    End
  Endif
Loop
```

●


PHOTOS BY LEE H. PAPPAS

# Deutschland? Ja! Atari? Ja! Ja!

## The Atari is alive and well in the West German marketplace.

**by Frank Cohen**

Returning home from a trip to Switzerland and West Germany was quite an experience. Spending two weeks skiing the Alps then investigating the personal computer industry in Europe had taken its toll on my health. I was fatigued, sick with a cold—ready to collapse in bed. Flights were delayed and, by the time I got back into Los Angeles, I was *happy* to see traffic, smog, and the beach.

Getting off the airplane, I asked someone for the time. "Well . . . Like it's about two o'clock," he said in what could only be described as Californian. You're probably familiar with Californian; it's the same language Atari uses to discuss firm release dates for their announced products. "Like, I don't know exactly. Well . . . Like, you know!"

When I got home, I slept for what seemed like a week. It was nice to have a few days to rest up and reflect on what I'd seen and done the past three weeks.

The trip started off as a nice vacation in Davos, Switzerland, with Lee Pappas of ANALOG Publishing. (It's remarkable, the people you can meet at a COMDEX. That's how Lee and I originally got acquainted.) We spent a week in Davos watching it snow, during the harshest winter Europe's seen in the past fifty years.

After the vacation, we went to West Germany. I wanted to see if what Sig Hartman of Atari Corp. has been saying is true. You'll probably remember Sig as the man who coined the saying "We are number one in Germany!" (Actually, it's normally pronounced, "Vie are number vun in Germany!" as Sig's accent tends to be on the heavy side.)

Expecting this to be just another statement from Atari, I was ready for anything. After all, I've heard that the West Germans are very sophisticated and expect a lot from their computers. The same rumors said that West German computer users were highly trained and fluent in programming, database languages and very specific applications. I found much of this to be true, but not to the extent of the rumors.

The West German Atari computer market seems to be made up of about 200 retail outlets, spread throughout West Germany. There are a certain number of STs in East Germany, but most computer software and hardware manufacturers aren't willing to attempt that market, because of socio-political problems facing the Communist bloc countries.

Of the 200 retail outlets, approximately half are specialty shops that sell computer products, but mainly retail drafting supplies, accounting products and other noncomputer goods. The other half are either large department stores—on the order of Macy's or Robinson's—or computer shops which also carry other office equipment.

West Germany is not a large country, by comparison to the United States. If I were to travel by car across West Germany, the trip would take less than eight hours. In contrast, traveling by car from San Francisco to San Diego would take me over nine hours. So it was interesting to see just how closely the major West German cities were placed.

We started our trip into West Germany from Zurich, Switzerland. As there was some time to kill, Lee and I wandered through the airport shops, eventually finding a magazine rack. The current issue of **ANALOG Computing** was on sale—a promising sight.

There were a number of other computer magazines with an Atari emphasis: *Atari ST Computer, Aktueller Software Markt, 68000er, Computer Revue, Atari Magazin* and *Markt & Technik Computer Personlich*. Most of these are printed monthly, and feature reviews of new products, programming information and type-in programs.

The only problem I had was that these were all in a language I don't speak. But, after a couple of hours, I started picking up various words here and there. It might have been all the beer I'd been drinking (or maybe saying "Ja! Ja!" all the time) but I really began to comprehend some articles.

And there was a lot to comprehend. The special issue of *Atari ST Computer* magazine included an article describing how to

develop a Pascal compiler for the ST. On reading further, I realized it not only discussed the theory, but also included a full assembly language listing of the finished compiler! I've never seen such in-depth articles in an American magazine.

The content too was extensive. Normally, forty articles or more are published in the German offerings. The content is divid-ed into categories, including: news and rumors, product testing, applications of tested products, tips and tricks, game reviews, and user group information. The magazines are filled with screen shots of interesting products, most of which I'd never seen before.

Some advertisements showed interesting German products. **Jack Paint** (R&P Soft, 8490 Cham, West Germany) is a painting program that seems to have all of the functions **DEGAS Elite**, but is more oriented toward the beginner. A company called C-Soft (C-Soft, Holzfallerstr. 4, 8400 Regensburg, West Germany) has an integrated word processor, mail merge and painting program for the ST that seems to be a low-cost clone of **MacWrite** and **MacPaint**. GFA BASIC is getting a lot of attention. MichTron is marketing the American version. It's biggest competition seems to be Omikron BASIC (Omikron, Erlachstr. 15, 7534 Birkenfeld), which comes with a cartridge and disk.

Another interesting product is Ashton Tate's **dBase II** for the Atari ST. At first, I thought this version of **dBase** was an unauthorized clone. But the advertisements showed Ashton Tate's logo, so it must be a legitimate version. Ashton Tate is the third largest software company in the world, following Lotus and Microsoft. The availability of an Ashton Tate product indicates that the ST *must* be making some strides at getting the attention of the big guys.

Trains are very popular in Europe; Lee and I saw a number of toy stores that sold mostly model train sets. *Atari Magazin* had a fifteen-page article to help you build an interface between a 520ST and a model train set. The article even included a program for switching tracks and for routing trains.

Our next stop was Munich, where we set out to find the biggest retail store in the city. We came upon Ludwig Computer & Burotechnik. There we met Mr. Alois Fisher, the owner and manager, who was very proud to show us around.

Ludwig is typical of the retail outlets in Germany. The Atari ST section of Ludwig made up about half of the overall products for sale. The store was decorated in a very modern style, using a system of enameled pipes and platforms bolted together to form tables, counters and display cabinets.

The front of the store was devoted to personal computers, with the rear set up for high-end CAD/CAM sales. The rear of the store had a large section for calculators and portable computers.

Ludwig sells both the Atari ST and Amiga 1000, with a clear leaning toward the ST. On display are the 520ST, 1040ST F and the 260ST. I had heard rumors that a 260 existed, but wasn't available in the U.S. It's quite popular in West Germany, with 1 megabyte of memory and without a mouse or monitor. The plastic case of the 260ST is the typical 520ST case and comes with a German keyboard.

Typing on this is an interesting feat. The *Y* and *Z* keys have been reversed. The brackets key has been removed and in its place is the special *umlaut* character (") for use over certain other keys. An extra key has been added to the left of the left-hand SHIFT key.

The systems also have the German version of GEM, with titles and icon names in that language for the desktop's drop-down menus. The system appears otherwise to function as does an American ST. I inserted **Regent Base** into a 260ST black-and-white system, and it functioned normally. **Regent Base** even accepted the special German keyboard characters.

The salesman at Ludwig told me that more than 90 percent of the STs sold in Germany are black-and-white systems, and that the 260ST is very popular, as the color monitors are expensive.

Hardware in Germany was also costly. An NEC P6 printer was being sold for 1498 German Marks. At 1.80 Marks to the dollar, that printer would cost $832. In the U.S. the same printer costs $479.

Software prices, too, were high. The **Magic Sac** Macintosh emulator was being sold for 398 Marks, about $221. Even Ashton Tate's **dBase II** costs twice as much as the equivalent product, **H&D Base**, in the U.S.

Ludwig carried a number of products I hadn't seen before. **The Art Director**, (PDS Software of Belgium) is a complete painting program with a huge variety of built-in functions. Several fonts, plus a disk library of drawings were included. **The Atari Director** costs 400 Marks, about $222. It will probably be available in the U.S. soon, as it's a high-quality program with outstanding features.

Omikron BASIC has a price of 229 Marks, about $128. It comes with a disk and a cartridge. The thing that makes this product unusual is that it lets you access most of the AES and VDI routines used to have GEM paint windows, operate drop-down menus, etc. A screen editor and compiler are included.

German software products are packaged in economical boxes, with little or no artwork. The Omikron BASIC cartridge comes in an ugly orange box with a simple hand drawing printed in black ink. Most of Data Becker's software line is sold in cheap white paper boxes with large yellow lettering. The reputation, *not* the packaging of a product, seems to be all it takes to make or break a company producing software in Germany.

We were shown an interesting CAD program, which seemed like a turbo version of **CAD 3D** from The Catalog, with greatly enhanced speed and versatility. The CAD program was copy protected. The protection was in the form of a cartridge, plugged into the ST, containing a special serial number which the program checks before it runs.

Later that day, we drove to the center of Munich to visit a department store called Hertie, where you can buy anything from beds to car parts. Hertie is a national chain, with stores in most of the large cities in West Germany.

In a section of the store, called Comp & Carry, computers and computer-related products are sold. Comp & Carry also had a magazine rack with current issues of **ANALOG Computing**, *Antic*, and *STart*, *Byte* and *COMPUTE!*

We conversed with Heinrich Huneke, a salesman who spoke reasonable English. Mr. Huneke was surprised at our interest in the German market. He told us most Germans follow Atari news from the magazines he carries—Germans seem to take English and two other languages as a normal part of their education.

Comp & Carry sold the 260ST and the 520ST. They had carried the 1040ST, but discontinued it after sales fell off. Mr. Huneke told us that Germans are very apprehensive of the 1040ST, as the initial machines were shipped with defective or poorly constructed internal disk drives. He said the 260ST is popular, as there are a number of low-cost monitors, color and black-and-white, available. The average price of a 520ST monochrome system was $1700.

Looking at Comp & Carry's software selection, I realized something about the

German computer market: software is very limited and not many products are available. I was expecting to see hundreds of software products. Both Ludwig and Comp & Carry sold about thirty products, but that was the full extent of the products available. Even the mail-order companies advertising in the magazines listed small numbers of products.

Lee and I visited other stores and companies in Frankfurt, our next stop. But, since Germany is such a small country by comparison to the U.S., the products offered in various cities showed little variation from what we'd already seen.

Two weeks after we left, the CEBIT trade show was held in Hanover, West Germany. There, Atari was to show some new hardware products, and the software publishers were to have some great new offerings for the ST. I didn't mind missing it—except I can't report on it for you—because I was exhausted by then.

We left Germany from Frankfurt the next day. The flight was uneventful, until I started talking to the man next to me. I told him I wrote software for the Atari ST. He laughed a little and told me he had one of the original 800s, even did some beta testing for Atari. The plane cruised toward America with the two of us talking about our favorite games and telling old Atari stores. ///

---

*Frank Cohen has been developing software for Atari computers since 1983, when he wrote his first game,* ***Clowns and Balloons****. He founded Regent Software in 1985 and wrote* ***Regent Base*** *for that label. He can be contacted on Delphi (as REGENTWARE), or on CompuServe (72457,3171).*

---



# 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 **ST-Log** Group Listing, P.O. Box 23, Worcester, MA 01603. **ST-Log** is not responsible for errors.

 *continued from page 20*

```
        res3 := 0 ;
      END ;

   If dir = 0 THEN
     Copy_Opaque( screen, screen_form, ROP_Replace,
                  x, y, w, h, 0, 0, w, h )
   ELSE
     Copy_Opaque( screen_form, screen, ROP_Replace,
                  0, 0, w, h, x, y, w, h )

   END ;

BEGIN
END.
{ End of blitscrn.pas }
```

Listing 3.
Pascal listing.

```
PROGRAM PaintBrush;
  { Put GemRast.O and Blitscrn.o in additional link files under link options}

  CONST
    {$I gemconst.pas}

  TYPE
    {$I gemtype.pas}

    Form_Format = ( Device_Format, Standard ) ;
    Mem_Form_Def = RECORD
                     mem : long_integer ;
                     pw : integer ;
                     ph : integer ;
                     width : integer ;
                     format : Form_Format ;
                     planes : integer ;
                     res1 : integer ;
                     res2 : integer ;
                     res3 : integer ;
                   END ;
    buffer = Packed ARRAY[1..15000] OF char;

  VAR
    window,xw,yw,ww,hw,mx,my,event,dum : Integer;
    scrn_sav : buffer;
    msg : Message_Buffer;
    wind_name : window_title;
  {$I gemsubs.pas}

PROCEDURE Blit_Screen(x,y,w,h,dir : integer ; VAR scrn_ary : buffer );
    EXTERNAL;

BEGIN  {PaintBrush}
 IF Init_Gem>=0 THEN
  BEGIN  {main program}
    Init_Mouse;
    wind_name:='PAINT BRUSH';
    window:=New_Window(G_Name|G_Close,wind_name,0,0,0,0);
    Open_Window(window,0,0,0,0);
    Work_Rect(window,xw,yw,ww,hw);
    Paint_Color(White);
    Paint_Style(Solid);
    Hide_Mouse;
    Paint_Rect(xw,yw,ww,hw);
    Paint_Color(2);
    Paint_Style(Solid);
    Paint_Rect(10,25,40,25);
    Line_Color(3);
    Line(10,25,50,50);
    Line(50,25,10,50);
    Blit_Screen(10,25,40,25,0,scrn_sav);  {save}
    Paint_Color(white);
    Paint_Rect(10,25,40,25);  {erase}
    Show_Mouse;
    WHILE (msg[0]<>WM_Closed) DO BEGIN  {While}
     event:=Get_Event(E_Message|E_Button,1,1,1,0,false,0,0,0,0,false,0,0,0,0,
                      msg,dum,dum,dum,mx,my,dum);
```

```
      IF event & E_Button <> 0 THEN
       BEGIN {event loop}
        Hide_Mouse;
        Blit_Screen(mx,my,40,25,1,scrn_sav);  {draw}
        Show_Mouse;
       END  {event loop}
     END;  {While}
     Close_Window(window);
     Delete_Window(window);
     Exit_Gem
    END  {PaintBrush}
  END.
```

Listing 4.
Pascal listing.

```
PROGRAM Animate;
  { Put GemRast.O and Blitscrn.o in additional link files under link options}

  CONST
    {$I gemconst.pas}

  TYPE
    {$I gemtype.pas}

    Form_Format = ( Device_Format, Standard ) ;
    Mem_Form_Def = RECORD
                     mem : long_integer ;
                     pw : integer ;
                     ph : integer ;
                     width : integer ;
                     format : Form_Format ;
                     planes : integer ;
                     res1 : integer ;
                     res2 : integer ;
                     res3 : integer ;
                   END ;
    buffer = Packed ARRAY[1..8000] OF char;


  VAR
    key,tmr,flag,window,xw,yw,ww,hw,mx,my,event,dum : Integer;
    scrn0,scrn1,scrn2,scrn3 : buffer;
    msg : Message_Buffer;
    wind_name : window_title;
  {$I gemsubs.pas}


PROCEDURE Blit_Screen(x,y,w,h,dir : integer ; VAR scrn_ary : buffer );
    EXTERNAL;

PROCEDURE Draw_Ghost(x,y,flg : Integer);
 BEGIN  {Draw_Ghost}
  Paint_Rect(x,y,16,14);  {block}
  Plot(x+flg,y+14);  Plot(x+flg+4,y+14);
  Plot(x+flg+8,y+14);  Plot(x+flg+12,y+14);  {feet}
  Line_Color(0);
  Plot(x+flg+4,y+4);  Plot(x+flg+8,y+4);
  Plot(x+flg+4,y+5);  Plot(x+flg+8,y+5);  {eyes}
 END;  {Draw_Ghost}

PROCEDURE  Ghost(flg : Integer);
 BEGIN  {Ghost}
  Paint_Color(2);
  Line_Color(2);
  Draw_Ghost(45,25,flg);
  Paint_Color(1);
  Line_Color(1);
  Draw_Ghost(10,50,flg);
  Paint_Color(3);
  Line_Color(3);
  Draw_Ghost(45,50,flg);
 END;  {Ghost}

PROCEDURE Erase ;
 BEGIN
   Paint_Color(White);
   Paint_Rect(0,25,112,50)  {erase}
 END;
```

```
BEGIN  {Animate}
 IF Init_Gem>=0 THEN
  BEGIN  {main program}
   tmr := 100;
   flag := 0;
   Init_Mouse;
   wind_name:='ANIMATION';
   window:=New_Window(G_Name|G_Close,wind_name,0,0,0,0);
   Open_Window(window,0,0,0,0);
   Work_Rect(window,xw,yw,ww,hw);
   Paint_Color(White);
   Paint_Style(Solid);
   Hide_Mouse;
   Paint_Rect(xw,yw,ww,hw); {fill the window with white backround}
   Erase;  {begin with clean slate}
   Paint_Color(2);
   Paint_Style(Solid);
   Paint_Arc(25,32,15,7,0,3600);   {first shape}
   Ghost(0);
   Blit_Screen(0,25,112,50,0,scrn0);  {save}
   Erase;
   Paint_Color(2);
   Paint_Arc(25,32,15,7,200,3400);  {second shape}
   Ghost(1);
   Blit_Screen(0,25,112,50,0,scrn1);  {save}
   Erase;
   Paint_Color(2);
   Paint_Arc(25,32,15,7,400,3200);  {third shape}
   Ghost(2);
   Blit_Screen(0,25,112,50,0,scrn2);  {save}
   Erase;
   Paint_Color(2);
   Paint_Arc(25,32,15,7,600,3000);  {fourth shape}
   Ghost(3);
   Blit_Screen(0,25,112,50,0,scrn3);  {save}
   Erase;
   Draw_String(xw,yw+10,'Move mouse to move figure, press up/down arrows');
   Draw_String(xw,yw+20,'to decrease/increase animation speed, click close');
   Draw_String(xw,yw+30,'box to end this demo.');
   Show_Mouse;
   WHILE (msg[0]<>WM_Closed) DO BEGIN  {While}
    event:=Get_Event(E_Message|E_Keyboard|E_Timer,1,1,1,tmr,false,0,0,0,0,
                     false,0,0,0,0,msg,key,dum,dum,mx,my,dum);
    IF event & E_Keyboard <> 0 THEN BEGIN  {process key press}
     IF key = $4800 THEN BEGIN {up arrow}
      tmr:=tmr+10;
      IF tmr>1000 THEN tmr:=1000
     END;  {up arrow}
     IF key = $5000 THEN BEGIN {down arrow}
      tmr :=tmr-10;
      IF tmr<1 THEN tmr:=1
     END  {down arrow}
    END;  {process key press}
    IF event & E_Timer <> 0 THEN BEGIN  {timer event}
      flag:=flag+1;
      IF flag=7 THEN flag:=1;
      Hide_Mouse;
      IF flag = 1 THEN
       Blit_Screen(mx,my,112,50,1,scrn0);  {draw}
      IF flag = 2 THEN
       Blit_Screen(mx,my,112,50,1,scrn1);  {draw}
      IF flag = 3 THEN
       Blit_Screen(mx,my,112,50,1,scrn2);  {draw}
      IF flag = 4 THEN
       Blit_Screen(mx,my,112,50,1,scrn3);  {draw}
      IF flag = 5 THEN
       Blit_Screen(mx,my,112,50,1,scrn2);  {draw}
      IF flag = 6 THEN
       Blit_Screen(mx,my,112,50,1,scrn1);  {draw}
      Show_Mouse;
     END  {timer event}
   END;  {While}
   Close_Window(window);
   Delete_Window(window);
   Exit_Gem
  END  {Animate}
END.

{end of Animate.Pas}
```

# Attention Programmers!

**ST-Log** is interested in programs, articles, and software review submissions dealing with the Atari home computers. 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 **ST-Log**, authors acknowledge that such materials, upon acceptance for publication, become the exclusive property of **ST-Log**. 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, **ST-Log**
P.O. Box 23, Worcester, MA 01603.

# INDEX TO ADVERTISERS



*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.*

# Kill ~~Put on~~ a Happy Face!


**New from Xanth F/X! The game that lets up to 16 people play at the same time on 16 different STs. All networked together using MIDI.**

Design your own mazes using First Word™.
Select up to 15 happy face opponents in single player mode.
Requires 1 MIDI cable for each computer in network.

Only $39.95 suggested retail.

For a free product catalog and newsletter call or write us at:

**Hybrid Arts, Inc.™**

11920 W. Olympic Blvd., Los Angeles, CA. 90064
**Computer BBS**: (213) 826-4288 **Main Office**: (213) 826-3777 **Telex**: 4931231 (HYBRDUI)

Hybrid Arts offers a full range of MIDI products for the Atari ST including:
**EZ-Track™**: A 20 track polyphonic MIDI recorder.
**EZ-Score™**: A sheet music printing program.
**CZ-Android™**: A patch librarian-editing system for the Casio CZ keyboard series.
**DX-Android™**: A patch librarian-editing system for the Yamaha DX-7 and TX series.
**GenPatch ST™**: A generic patch librarian system for all MIDI devices.
**MIDITrack ST™**: A professional MIDI recorder availlable with and without SMPTE sync.
**ADAP SoundRack™**: A 16 bit digital audio workstation for music and broadcast professionals.



CIRCLE #124 ON READER SERVICE CARD

WHY SETTLE FOR 2ND PLACE WITH BASEBALL GAMES?

# HIT A GRAND SLAM!

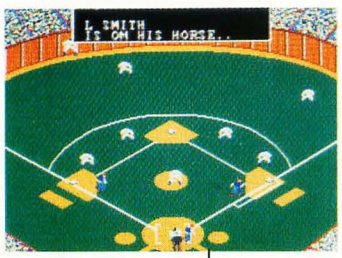
# MICRO LEAGUE Baseball® II

## The "Exclusive" strategy simulation of Major League Baseball.

By popular demand, in its *SECOND* great edition, MicroLeague Baseball is a winner again! And it's so good, it's the MVP (Most Valuable Program) of baseball simulation games.

**New levels of baseball strategy.** Now you can enjoy an all new level of baseball strategy. Because MicroLeague Baseball II features everything the award-winning first edition had, including: dynamic, high-quality graphics ■ real players and stats ■ true-to-life performance ■ powerhouse teams from the past and present ■ and exclusive endorsement by Major League Baseball and the Players Association. Whether you play a game or a season, if you're looking for managerial excitement, you can't lose.


Natural vs. Artificial Turf



**We made it even better!** MicroLeague Baseball II goes beyond the first edition, with phenomenal new visuals and sophisticated features such as: stadium dimensions, grass and artificial turf factors, box score and stat compiler *built in*, so you can keep track of your players and their season ■ expanded statistical factors if you're a baseball purist ■ injuries, arguments with the ump and even rain delays. Plus a *"Quik-Play Option"*, if you like to get to the bottom of the ninth inning *fast*.

**Major League Excitement Year-Round!**

MicroLeague Baseball is *not* a joystick game. Your *mind* — not your reflexes — are in control, and that's what keeps you challenged, game after game. The sights, sounds and stats create a level of involvement no other product can approach. Add to this MLB Season disks and the unique General Manager/Owner disk for drafting & trading players — even creating *your* own fantasy dream teams, and you've got a League Champ!

OFFICIAL LICENSEE MAJOR LEAGUE BASEBALL®

MLB

Official Licensee Major League Baseball, Major League Players Association

MicroLeague Baseball II is available for the Atari ST systems, and will be coming for the Amiga and MAC this summer.

Available at computer and software stores everywhere; or for direct VISA/MC orders simply call **1-800-PLAYBAL**.

**MicroLeague I is still available for Apple II Series; Atari 65/130 XE, 800/XL; Commodore 64/128; and IBM PC, PC Jr. and compatibles.**

**MicroLeague Sports**
**2201 Drummond Plaza**
**Newark, DE 19711**
**(302) 368-9990**




CIRCLE #125 ON READER SERVICE CARD