ATARI® EXPLORER

THE OFFICIAL ATARI JOURNAL

March/April 1989
$2.95 / $3.95 In Canada

Free Inside:
A Calculator Accessory

The High-Performance ST

Programs For Power Users: Revolver, ST Control, TOS 1.4

Reviews:
- DOS XE
- Turbo ST
- Art & Film Director
- Disk Drive Switchers
- Chrono Quest
- All Aboard
- Sinbad
- Star Quake
- Offshore Warrior

C Life: GEM Resource Files

80-Column Word Processing For 8-Bits

Aussie JOKER POKER $200,000 JACKPOT

The latest multi-player multi-format PC game from Australia is different to all forms of Poker.

Aussie JOKER POKER features 90 player capacity, open-ended discard ability, selectable deck size and hands per player, password controlled gambling system with automatic accounts – and **5 free entry forms for the $200,000 Aussie JOKER POKER Contest.**

Each month December 1988 through April 1989 winners of 240 JOKER SOFTWARE games and 4 finalists will be randomly drawn from all entries received that month.

With a guest, the **20** Finalists will be flown to Las Vegas to play **Aussie JOKER POKER** for a **first prize of $100,000 in cash** at the **Golden Nugget.**

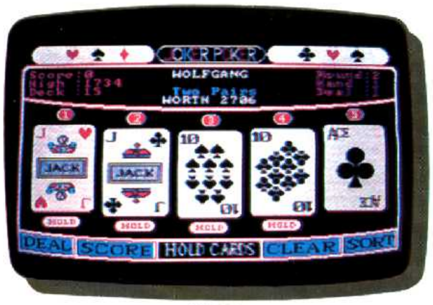
1,220 Prizes Value $200,000

1,200 Joker PC software games at $29.95 to $49.95 dependent on disk format. Game prizes at sole discretion of sponsor.	$60,000

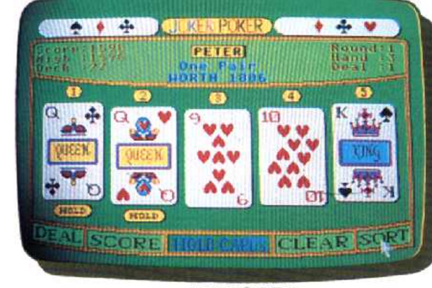
Cash Prizes for Aussie JOKER POKER Contest Grand Final:

Highest Scorer:	$100,000
Second Highest Scorer:	$5,000
Third Highest Scorer:	$2,500
Lowest Scorer:	$1,500
16 Consolation Prizes of $1,000 each to eliminated Finalists	$16,000
Prize includes air travel for Finalists and their guests from the major airport to Las Vegas with two days and two nights accommodation at the Golden Nugget (approx. retail value $750 each subject to departure points).	$15,000

All taxes and other expenses not specified herein are sole responsibility of winners. All winners will be notified in writing



IBM®



ATARI® ST™

Aussie JOKER POKER is available for SIX major PC's

If your PC has a mouse or keyboard, a mono or color monitor and a 512K minimum ram (except Apple II and C64/128 use 64K and keyboard only) you and your family can practise at home for the Las Vegas final of the **Aussie JOKER POKER contest.**

Suggested retail prices:

IBM & compatibles (CGA Board required)	**$39.95**
Amiga & Atari ST	**$49.95**
Macintosh (mono only)	**$49.95**
Apple II	**$39.95**
C64/128	**$29.95**

If ordering by telephone add $3 shipping & handling and check that your PC meets the minimum hardware requirements as no cash refunds apply. Warranty is limited to free replacement of faulty products returned by prepaid post.

Aussie JOKER POKER

A Gambling Game of Skill & Chance

GOLDEN NUGGET

Aussie JOKER POKER Contest Rules

1. No purchase necessary to enter.
2. Void where prohibited by state or federal law.
3. To enter, simply complete and return the the official entry form.
4. Limit five entries per family or household. Five free entry forms and full contest rules are included with "Aussie Joker Poker" or may be obtained by sending a stamped self-addressed envelope larger than 5½" × 7½" with a hand written request to: Aussie Joker Poker Contest Entry Forms, P.O. Box 22381, Gilroy, CA 95021-2381. Mail-in requests limited to one per name, household or family and must be received no later than 3/31/89. WA & VT residents need not include return postage. Full rules also available from participating Mindscape retailers.
5. Monthly entries must be received no later than the last day of the month in which a drawing will take place in order to participate in the month's drawing. Drawings will be held from December, 1988 through April 1989, inclusive. Final entries must be received by 4/30/89.
6. Contest open to legal residents of the U.S.A. and Canada (other than Quebec).
7. Odds of winning depend on number of eligible entries received.
8. Contest subject to complete official rules.

SEE YOUR NEAREST MINDSCAPE SOFTWARE RETAILER

EXCLUSIVE DISTRIBUTOR

MINDSCAPE INC

or if not available order direct on

1-800-24-JOKER

IN CANADA: 1-800-54-JOKER.

24 hour order service

JOKER SOFTWARE INTERNATIONAL.
PO BOX 22380, GILROY CA 95021-2380.

RETAILERS CALL: 1-800-221-9884

B

IBM, Apple & Macintosh, Amiga, Atari ST and C64/128 are trademarks or registered trademarks of International Business Machines, Apple Computer, Inc., Commodore Amiga, Inc., Atari, Inc., and Commodore Electronics Ltd. respectively. © 1988 Joker Software

JP13A

COMPUTER GAMES +

FIVE STAR 34.95

RAMPAGE – CRAZY CARS
ENDURO RACER – WIZBALL
BARBARIAN (Ultimate Warrior)

Arcade Force 4 49.95

GAUNTLET
ROAD RUNNER – METROCROSS
INDIANA JONES

MEGA PACK 34.95

FROST BYTE–MOUSE TRAP
WINTER OLYMPIAD 88
BLOOD FEVER–PLUTOS

ACTION ST 29.95

NORTH STAR – TRAIL BLAZER
3D GALEX – DEFLECTOR
MASTERS OF THE UNIVERSE

HIT DISK #1 34.95

GOLDRUNNER
KARATE KID II – SLAYGON
JUPITER PROBE

TRIAD 39.95

DEFENDER OF THE CROWN
BARBARIAN
STAR GLIDER

ARCADE CONVERSIONS–ST

GAUNTLET II	29.95
OPERATION WOLF	29.95
THUNDER BLADE	29.95
DOUBLE DRAGON	29.95
R-TYPE	29.95
OUTRUN	29.95
BIONIC COMMANDO	29.95
STREET FIGHTER	29.95
VICTORY ROAD	29.95
WANTED (GUNSMOKE)	29.95
FLYING SHARK	29.95
AFTER BURNER	29.95
SUPER HANG ON	29.95
1943	29.95
ALIEN SYNDROME	29.95
SIDE ARMS	29.95
ARKANOID	29.95
ARKANOID II	29.95
BUBBLE BOBBLE	29.95
STAR WARS	29.95
EMPIRE STRIKES BACK	29.95
RETURN OF THE JEDI	29.95
SHACKLED	29.95
SOLOMON'S KEY	29.95
SPACE HARRIER	29.95
IKARI WARRIORS	29.95

EUROPEAN ST SOFTWARE FROM THE #1 IMPORT

Dealer Inquiries Welcome
Over 300 titles imported

ACTION SERVICE	29.95
ADDICTABALL	29.95
ANNALS OF ROME	34.95
BAD CAT	29.95
BATMAN	29.95
BERMUDA PROJECT	34.95
BETTER DEAD THAN ALIEN	29.95
BLUE WAR	29.95
BOMBUZAL	34.95
BOMB JACK	29.95
BRIDGE PLAYER 2000	29.95
CAPTAIN AMERICA	29.95
CAPTAIN BLOOD	34.95
CHUCKIE EGG	29.95
CIRCUS GAMES	34.95
CYBERNOID	29.95
ELIMINATOR	29.95
ESPIONAGE	29.95
EXOLON	29.95
FERNANDEZ MUST DIE	34.95
GALACTIC CONQUEROR	29.95
GARFIELD	29.95
HELTER SKELTER	22.95
HOSTAGES	34.95
INTERNATIONAL KARATE +	29.95
JOAN OF ARC	29.95
LANCELOT	29.95
LEGEND OF THE SWORD	34.95
LIVE & LET DIE	29.95
LOMBARD RALLY	34.95
MACH 3	29.95
MICKEY MOUSE	29.95
MINI GOLF	29.95
MORTVILLE MANOR	19.95
MUNSTERS	29.95
NEBULUS (Tower Topler)	29.95
NETHERWORLD	29.95
PACMANIA	29.95
PANDORA	29.95
PURPLE SATURN DAY	34.95
RAMBO III	29.95
SDI (new not Cinemaware)	29.95
SHACKLED	29.95
SKRULL	34.95
SPEED BALL	34.95
SPITTING IMAGE	29.95
STAC	49.95
STAR GOOSE	29.95
STOS	44.95
SUPERMAN	29.95
TETRIS	29.95
THUNDERCATS	29.95
TIME & MAGIC	29.95
TRACKER	34.95
TRANTOR	29.95
TRIP-A-TRON	49.95
TRIVIAL PURSUIT II	29.95
TURBO CUP	29.95
VIXEN	29.95
VECTORBALL	29.95

Many more imports available, send self addressed stamped envelope for complete list

DOMESTIC ST SOFTWARE

AWESOME ARCADE PACK	37.95
BREACH	27.95
BREACH SCENARIO	19.95
BUGGY BOY (Speed Buggy)	24.95
CARRIER COMMAND	29.95
CHRONO QUEST	34.95
CORRUPTION	29.95
DUNGEON MASTER	24.95
DUNGEON MAPS	4.95
DUNGEON MASTERY	12.95
FALCON	34.95
FLIGHT SIMULATOR II	34.95
GOLD OF THE REALM	27.95
GOLDRUNNER II	27.95
SCENERY DISK I	11.95
SCENERY DISK II	11.95
HEROES OF THE LANCE	29.95
INTERNATIONAL SOCCER	27.95
JET + JAPAN SCENERY	34.95
KENNEDYAPPROACH	34.95
LEATHERNECK	27.95
MENACE	27.95
NIGHT RAIDER(DiveBombr)	24.95
OBLITERATOR	27.95
OIDS	24.95
PALADIN	27.95
PALADIN QUEST	19.95
POLICE QUEST	29.95
PRESIDENT IS MISSING	34.95
ROAD RAIDER	27.95
ROCKFORD	24.95
SCRABBLE DELUXE	24.95
SHADOWGATE	29.95
SHANGHAI	24.95
SKYCHASE	27.95
SPITFIRE 40	24.95
STARGLIDER II	32.95
STAR RAY	24.95
STAR TREK	27.95
STELLAR CRUSADE	34.95
TECHNO COP	27.95
TIMES OF LORE	34.95
TYPHOON THOMPSON	24.95
ULTIMA IV	34.95
U.M.S	29.95
ZANY GOLF	29.95

PRODUCTIVITY SOFTWARE

TIMEWORKS PUBLISHER	89.95
PUBLISHINGPARTNERPro.	129.95
PRINTMASTER PLUS	27.95
TYPING TUTOR	24.95
MAVIS BEACON	37.95
WORD UP	59.95
WORD WRITER	47.95
DATA MANAGER	47.95
PARTNER ST	34.95
BASE TWO	44.95
VIP PROFESSIONAL (GEM)	89.95
DEGAS ELITE	39.95
ART & FILM DIRECTOR	59.95
PC DITTO	69.95
ST TALK PROFESSIONAL	22.95
DESK CART	69.95
MULTI DESK	22.95
G+	27.95
OMNIRES	24.95
GFA BASIC 3.0	69.95
REBOOT CAMP	14.95
Programmers Ref. Guide	22.95
Concepts In Programming	19.95

Many more titles available, call for those not listed

DISK DRIVES

INDUS GTS-100X	179.95
INDUS GTS-100	189.95
INDUS GTS-1000	199.95
30MEG HARD DISK	619.95

MONITORS

COLOR	319.95
MONO	179.95

IMPORTED MAGAZINES

ST ACTION	6.95
ST USER	6.95
COMPUTER & VIDEO GAMES	6.95
GAMES MACHINE	6.95
ST/AMIGA (DISK)	8.95
THE ONE	6.95

XE/XL OWNERS

send SASE for list of specials

SPECTRUM 128	**129.95**
COMPUTEREYES	**179.95**
IMAGE SCAN	**79.95**
ASTRA MONITOR BOX	**49.95**
SUPRA 2400 MODEM	**139.95**

come visit our walk-in store at
1839 E. Chapman
Orange CA
Store Hours Noon-6 Mon-Fri (714)538-1234
MailOrder Hours 9-6Mon-Sat(714)639-8189

714-639-8189

ORDERS ONLY PLEASE
1-800-443-8189

MasterCard

SHIPPING: Software - free shipping on U.S. orders over $100, otherwise $2.50 U.S., $6.50 outside U.S. Hardware - depends on weight, call for quote. Charge cards + 3%. C.O.D. orders are welcome, add $2.20 for UPS + 3%.

VISA

COMPUTER GAMES + • Box 6144 • ORANGE CA 92667 • (714) 639-8189

ATARI EXPLORER
THE OFFICIAL ATARI JOURNAL

MARCH/APRIL 1989

PROGRAMMING

50 **Total 100 . . . Solved!**/*Ahl*
The solution to an old friend . . . and a new challenge.

72 **C Life**/*Jainschigg*
An introduction to the resource files.

64 **Hacker's Progress**/*Jainschigg*
A calculator desk accessory for the ST and how it came to be.

REVIEWS

16 **TOS 1.4**/*Ramsey*
Updates to the ST OS fix some old problems and add some new capabilities.

22 **Art and Film Director**/*Kane*
Epyx releases a multi-featured art and animation program for the ST.

26 **Revolver**/*Jainschigg*
The program that will change the way you use your ST.

34 **ST Control**/*Jainschigg*
If you don't mind a little programming, this universal control program can help you turbocharge your ST.

39 **Atari SFP004**/*Ramsey*
Atari releases a floating point coprocessor and developer kit for the Mega ST.

40 **Turboword**/*Noyes*
A capable, full-featured, 80-column word processor for 8-bit Atari computers.

44 **DOS XE**/*Bocchino*
A powerful, user-friendly operating system for 8-bit Atari computers.

48 **Astra MIDI Distribution Box**/*Livingston*
Increasing the number of ports on the ST makes MIDI easier to manage.

53 **Disk Drive Switches**/*Ahl*
Two convenient ways to eliminate the annoyance of drive swapping.

54 **Software Survey**/*Andrews, Cove, Eddy, and Manor*
Lots of new software packages to fill the shelves of your library.

Cover photograph by Jeff MacWright

Atari Explorer(ISSN: 0882-3340) is published bi-monthly for $14.95 per year by Atari Explorer Publications Corp. All rights reserved.

POSTMASTER:
Send address corrections to Atari Explorer, CN961, Netcong, NJ 07857-0691.

FEATURES

14 **Trivia Quiz**/*Ahl*
How much do you know about computing past and present?

18 **Graphics Gallery**/*Ahl*
Ten readers win free subscriptions with top graphics entries.

DEPARTMENTS

4 **Letters**/*Readers*
LDW Power, 8-bits forever, and a wish-come-true.

6 **New Products**/*Staples*
The latest releases in hardware and software for ST and 8-bit computers.

10 **Puzzles & Problems**/*Ahl*
Use your Atari computer to solve these brainteasers.

12 **User Friendly**/*Noyes*
Hints and out-takes from Atari user group newsletters.

78 **Dealer Directory**/*Edwards*
Look here to find the name of an Atari dealer in your area.

74 **Index to Advertisers**

Call For Authors

The editors of *Atari Explorer* are always looking for competent writers to review products and do feature articles on assignment. The requirements are few; you must:

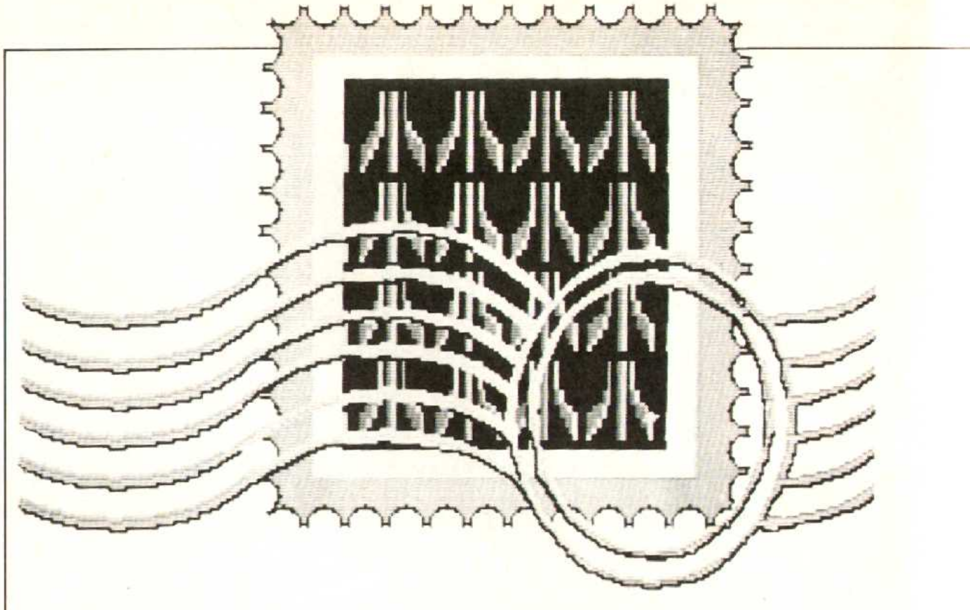
- Have a good command of the English language and a pleasing writing style.
- Have access to an Atari 8-bit or ST computer and be familiar with its operation.
- Be able to meet deadlines and follow directions.
- Be able to provide a manuscript in *1st Word* or ASCII format on an ST disk or via upload to CompuServe or Genie.

If you think you qualify and want to be considered for writing assignments, please send us an *unedited*, double-spaced manuscript (a disk is not necessary) of a review or feature of the type you would like to write. Include a cover letter describing your system(s), your qualifications, and the type of product you feel most competent to review—graphics, MIDI, utilities, desktop publishing, etc. (competition is stiffest for reviewers of arcade games). Be sure to include your address (street address as well as P.O. box) and day-time phone number.

We promise to review every submission carefully, but we have a small staff and simply cannot acknowledge or critique individual submissions. If we like your work and have a product we feel is appropriate for you to evaluate, we will call you. If you want us to acknowledge receipt of your ms., please include a stamped, self-addressed postcard.

Send your manuscript to Prospective Author, Atari Explorer, 7 Hilltop Rd., Mendham, NJ 07945.

Hint: The only way to be considered for a writing assignment with *Atari Explorer* is to follow these instructions exactly. Don't disqualify yourself by single-spacing your manuscript or calling our office to find out the status of your submission. ■



Letters To The Editor

Power-flaw

Dear Editor:

I have been in the securities business for the past 11 years, so it was with more than passing interest that I read your recent review of *LDW Power*. What was attractive to me was the ostensible increase in processing speed over *VIP Professional* coupled with additional features found in *Lotus 1-2-3* v. 2.0.

After reading yours and several other reviews, I purchased the program. Unfortunately, upon using the spreadsheet I have found what is, at least for my type of work, a fatal flaw in an otherwise outstanding effort.

Specifically, in comparison with *VIP Professional*, the graphing capability of *LDW Power* is severely limited. This limitation results from the fact that only 125-130 individual data points can be graphed at one time, and this occurs in using a simple line graph designation without any symbol configurations that might require additional space. *VIP* has no such limitation that I have been able to discern.

For those uninitiated in the analysis of securities, I should explain that charting is a fundamental aspect in determining such factors as supply and demand, support, resistance, etc. Typically, price charts for stocks minimally include the past 12 months of daily price quotations. With the average trading year consisting of 220+ days, the graphing capability of *LDW Power* presents an unacceptable handicap.

On a positive note, I should add that Mr. Stephan Zielinski, the technical support person at LDW was genuinely helpful, sympathetic, and a generally positive aspect of the product. It is important, however, for readers such as myself, who have extensive data needs as regards spreadsheets, to be aware that the power of this program does not extend to its graphing capabilities.

Peter von Nessi
Managing Director
The Normandy Group
P.O. Box 2296
Westfield, NJ 07091

8-Bits Forever . . .

Dear Editor:

I am 16 years old and have been a die-hard 8-bitter for the past three years. I have argued for the XE over 64 and IIe and written hundreds of letters to software companies, urging them to release software for the XE/XL line. But every once in a while the Lone Ranger needs some consolation.

Am I the *only* one? For two years I have tried to get my user group fired up about the 8-bits, but they don't seem to have the same kind of enthusiasm I have for my computer. Everywhere I see fellow 8-bitters selling out for the ST world.

What really kills me is that I never any XE/XL TV commercials, while I see dozens for the 2600, 7800, and XEGS.

In conclusion, let me just make one

thing clear to you and your readers: I *love* Atari. I will always fight tooth and nail for Atari and the XE/XL line. I am working on opening up a computer store and have already sold five Atari computers to eager families.

Is anybody else out there feeling as I am? Write to me; it would really be great to hear from somebody who shares my enthusiasm.

Nathan Block
2656 Georgia Ave. S
St. Louis Park, MN 55426

We're sure that there are lots of loyal 8-bitters who feel as you do, and we do our best to cover what remains of the 8-bit market. In this issue, for example, you will find in-depth reviews of two significant new products for that line—Turboword and DOS XE. The bottom line, however, is simply that software publishers will develop any type of software (games or productivity, XE or IIe) on which they can make a profit—and you, the consumers, are ultimately the ones who determine with your hardware and software dollars whether a profit can be made.

. . . and Ever

Dear Editor:

Atari Explorer is a great magazine. For its size, it is packed with interesting articles and is a bargain.

There are, however, a few things that would improve *Explorer*, such as making it a monthly magazine, making it a bit larger (with a slightly higher price), and including a Reader Service card.

Also, I would like to see regular interviews with old and new Atarians, such as Nolan Bushnell, Al Alcorn, John Feagans, Shiraz Shivji, Michael Katz, and of course, Jack Tramiel.

And finally, let's have some write-in campaigns. To start, I urge all of you *Explorer* readers who own Atari game machines—2600, 7800, and XE—to write to Jack Tramiel and get him to bring back the Atari Club and *Atari Age* magazine for gamer support.

Keep up the good work; I'm looking forward to the improvements and the return of the Atari Club.

Jeremy Wilburne
Loyal Atarian
7755 El Modena Ave.
Elverta, CA 95626

Funny you should mention it . . . Jack (actually Michael Katz of Atari's Game Division) has done just that. The Explorer staff (actually Publisher Dave Ahl) is gearing up right now for the first issue of The Atarian, a magazine dedicated exclusively to Atari game systems. The new publication—which will be filled with game reviews, high scores, new product previews, interviews with old and new Atarians, and articles on playing strategy—will be available to members of the new Atarian Club, membership in which also entitles you to a T-shirt, discounts on software, and lots of other neat stuff. For more information look for the Atarian ad in this issue.

Unfortunately, your other wishes are not as easy to grant. The size of a magazine depends entirely on the number of advertisements it carries, so encourage the manufacturers of the Atari-related products you buy to advertise in Explorer. If enough of them respond, we will be able to expand—in frequency as well as volume.

ATARI®
EXPLORER
THE OFFICIAL ATARI JOURNAL

Before You Write or Call: Please do not call or write to us about problems with individual Atari products or orders placed with Atari Corp. We will simply forward your query or complaint to Atari Customer Service (P.O. Box 61657, Sunnyvale, CA 94088), where it will arrive days, if not weeks, later than it would have if you had mailed it there yourself. Barbara Monahan and Diana Goralczyk, Atari's customer service experts, are much better equipped to solve individual problems than we are.

Editorial: Editorial material, including article submissions, press releases, and products for evaluation, should be sent to Atari Explorer, 7 Hilltop Road, Mendham, NJ 07945 (201) 543-6007. Atari Explorer is not responsible for the return of unsolicited manuscripts, tapes, disks, program listings, etc. not submitted with a self-addressed, stamped envelope.

Subscriptions: Domestic: 6 issues, $14.95; 18 issues, $39.95. Canadian: Add $5.00 per 6 issues. Foreign: Add $10.00 per 6 issues. Foreign subscription orders must be accompanied by a check in U.S. funds drawn on a U.S. bank. Send orders to Atari Explorer, CN961, Netcong, NJ 07857-0961.

Customer Service: If you have a problem with your subscription, please write to Atari Explorer, CN961, Netcong, NJ 07857-0961, or call (201) 347-8383.

Publisher
David H. Ahl

Editor
Elizabeth B. Staples

Senior Technical Editor
John B. Jainschigg

West Coast Editor
Mark O. Jansen

Assistant to the Editors
Barbara Edwards

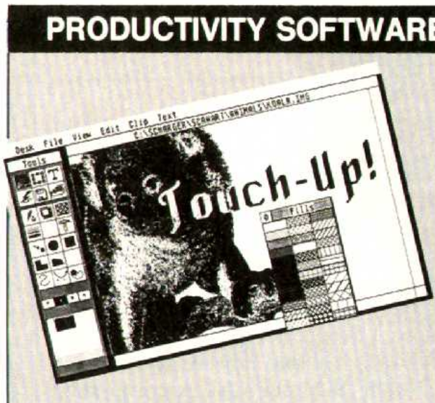
Contributing Editors
John J. Anderson
Stuart Dudley Dimond III
Arnie Katz
Frank Kofsky
Bill Kunkel
Owen Linzmayer
John S. Manor
Dave Noyes
Rick Teverbaugh
Joyce Worley

Art Director
Peter Kelley

NEW Products

New hardware, software, and accessories for Atari computers

PRODUCTIVITY SOFTWARE

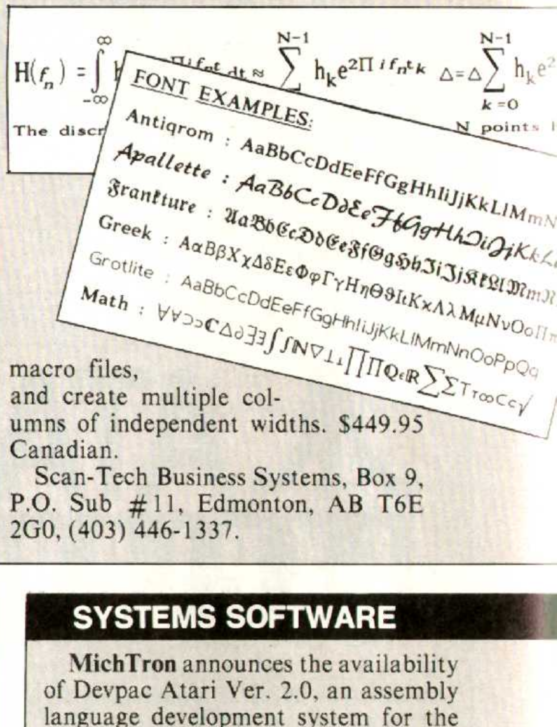


Migraph has announced *Touch-Up*, a comprehensive image design tool for producing high-resolution monochrome bit-mapped graphics on the Atari ST. Features include the ability to create and edit images larger than the screen size; extensive drawing tools, including Bezier curves and B-Splines; clip area functions, including masking, edging, contrast, flip, mirror, slant, and rotation; a complete paint program; scalable outline fonts in a variety of typefaces and styles; the ability to load color and monochrome formats, including .IMG, *NeoChrome*, *Degas*, *Printmaster*, *MacPaint*, and PCX; and the ability to save images in .IMG, TIFF, GIF, IFF/ILBM, and various paint formats. The program retails for "under $200."

Also available from Migraph is *Border Pack* for the Atari ST, a package that contains 40 object-oriented borders. Because the borders are object-oriented rather than bit-mapped, they print at the highest resolution of the printer. All borders are saved in .GEM format and can be used by any application that loads .GEM files. $34.95.

Migraph, 200 S. 333rd (220), Federal Way, WA 98003, (800) 223-3729, (206) 838-4677.

Signum 2 from **Scan-Tech** is a document processor for the Atari ST, which allows exact positioning of characters to within $^1/_{90}$" horizontally and $^1/_{54}$" vertically and provides high quality output on a 9-pin, 24-pin, or laser printer. Additional features include the ability to use up to seven fonts in a document, display each character in up to 64 different ways, import graphics from other programs, store system parameters in macro files, and create multiple columns of independent widths. $449.95 Canadian.


$H(f_n) = \int_{-\infty}^{\infty}$ ... $\approx \sum^{N-1} h_k e^{2\Pi i f_n t_k}$ $\Delta = \Delta \sum_{k=0}^{N-1} h_k e^2$

The discr ... N points

FONT EXAMPLES:

Antiqrom : AaBbCcDdEeFfGgHhIiJjKkLlMmN

Apallette : AaBbCcDdEeFfGgHhIiJjKkL

Frankture : 𝔄𝔞𝔅𝔟ℭ𝔠𝔇𝔡𝔈𝔢𝔉𝔣𝔊𝔤ℌ𝔥ℑ𝔦𝔍𝔧𝔎𝔨𝔏𝔩𝔐𝔪𝔑

Greek : AαBβXχΔδEεΦφΓγHηΘϑIιKκΛλMμNνOoΠπ

Grotlite : AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQq

Math : ∀∀⊃⊃ℂΔ∂∃∃∫∫ℕ∇⊥⊥∏∏ℚ∈ℝ∑∑T⊤∞ℂ⊂√

Scan-Tech Business Systems, Box 9, P.O. Sub #11, Edmonton, AB T6E 2G0, (403) 446-1337.

STereo Sound

Pratical Solutions announces Tweety Board, a device that elicits true stereo sound from the Atari ST. The small circuit board is designed to access the three existing channels "hidden" in the ST. It workc in parallel with the regular ST sound circuitry, making it compatible with all ST software.

Tweety Board plugs in inside the computer and provides three independent RCA jacks that can be connected to three amplified speakers. An adapter cable allows the device to be connected to a standard two-channel system. The board retails for $59.95.

Practical Solutions, 1930 E. Grant Rd., Tucson, AZ 85719, (602) 884-9612.

SYSTEMS SOFTWARE

MichTron announces the availability of Devpac Atari Ver. 2.0, an assembly language development system for the Atari ST, which incorporates an integrated editor/assembler/debugger, a stand-alone assembler and debugger, and a fast linker. The GenST Assembler is a full-featured, two-pass Motorola standard macro assembler that assembles up to 80,000 lines of code per minute and can produce directly executable programs or linkable object code in GST or DRI format. $99.95.

Also available from MichTron is PowerBasic, a Basic compiler for the Atari ST that conforms to Microsoft Basic standards and extends the latest specifications for Basic to take advantage of the large memory and special characteristics of 68000 machines. $79.95.

WERCS, also from MichTron, is a

Hi-Tech Advisers announces version 4.00 of *Sales-Pro* and **Sales-Pro Plus** for Atari ST computers. Enhancements include increased speed; new defaults; support of all dot matrix, parallel printers; 20-character inventory item ID field; new file utilities; up to 999 line items per transaction; and new reports, lists, and labels. $99 and $199, respectively. Upgrades are available to current program owners.

Hi-Tech Advisers, P.O. Box 7524, Winter Haven, FL 33883, (813) 294-1885, (800) 882-4310 (for orders).

Double Eagle Software has announced that the Atari ST version of *The Tax Advantage* is now capable of importing data from *Phasar* and IBM PC versions of *Dollars and Sense*, *Managing Your Money*, and *The Home Accountant*. The new version also includes the following new forms: Schedule R, Form 2119, Form 3903, Form 8598, and Form 8615. $59.95.

The program is also available for Atari 8-bit computers. $49.95.

Double Eagle Software, 2340 Plaza del Amo, Ste. 215, Torrance, CA 90501, (213) 212-6611.

Precision Software has released *Superbase Professional 3* for Atari 520, 1040, and Mega ST computers. Among the new features available in the upgraded program are telecommunications facilities, a more powerful forms editor, and cross-file validation and lookup. $349.95.

Registered owners of earlier versions of *Superbase Professional* can upgrade for $25.00.

Precision Software, 8404 Sterling St., Ste. A, Irving, TX 75063, (214) 929-4888.

Schaefer Supergraphics announces a line of medically related software for the Atari ST. *Diet Version 2.0* is a nutritional analysis program that can be used to calculate weight loss and daily caloric requirements, exercise caloric expenditures, and ideal body weight. It features an expandable GEM-based menu planner and complete documentation. $25. *Longevity* is a program that teaches users how risk factor modification, nutritional awareness, and sensible exercise can maximize life span. $39.95.

Code-Blue is a real-time cardiac arrest simulator, which continuously updates blood pressure, pulse, respiration, urine output, pH, pO2, pCO2, HCO3, and cardiac rhythm and displays them graphically on the computer screen. $69.95.

Gas-Lyte is a laboratory analysis tool with the following features: arterial blood gas analysis and differential diagnosis; electrolyte analysis for disturbances in sodium, potassium, calcium, and magnesium; calculation of creatine clearance, osmolarity, and anion gap; dosage recommendations for aminoglycosided and aminophiline. $25.

Schaefer Supergraphics, 1201 Wilder Ave. #1801, Honolulu, HI 96822, (808) 523-3353.

Disk Storage

The ArchiveDiskFile System #735 holds 75 $3\frac{1}{2}''$ disks in a corrugated cardboard drawer and sleeve that measures $4\frac{1}{4}''\times4\frac{3}{4}''\times11''$.

The files can be used individually or as modular, interlocking assemblies. Each drawer has a plastic pull and a large label for identification. A pack of three files sells for $21.00.

Weber & Sons, 3468 Highway 9, Freehold, NJ 07728, (800) 225-0044, (201) 431-1128.

resource editor for Atari ST computers. It permits creation of dialog boxes, menus, icons, and alert boxes with an intuitive GEM interface. An integrated icon/image editor supports standard editing features, such as clearing or filling a bitmap. $49.95.

MichTron, 576 S. Telegraph, Pontiac, MI 48053, (313) 334-8729.

ICD has released SpartaDOS X for Atari 8-bit computers. The cartridge-based DOS plugs directly into the computer and provides a piggyback extension for an additional cartridge. It makes full use of extra memory with RAMdisks supported up to 1Mb and high speed operation when teamed with either the Indus GT or Atari XF551 disk drive. $79.95.

ICD, 1220 Rock St., Rockford, IL 61101, (815) 968-2228.

GRAPHICS SOFTWARE

GFA Raytrace is now available for the Atari ST from **MichTron**. Raytracing is the process of simulating the path of a light ray from a source to a viewpoint to create a realistic image. The GFA program allows users to do this in up to 48 colors per scan line and up to 512 colors per screen. It is compatible with *GFA Artist*, *Degas*, *NeoChrome*, and *Spectrum 512*. $99.95.

MichTron, 576 S. Telegraph, Pontiac, MI 48053, (313) 334-8729.

NEW Products

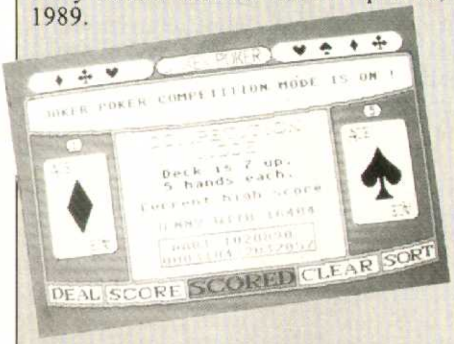
UTILITIES

New from **Migraph** is *OSpooler*, a configurable, communicating background file spooler and printer buffer. The program can be used to print any file on which the user can double-click from the desktop, such as a text file, plus any metafile that has been redirected to a spooler file. Drivers for 9-pin Epson graphics compatible printers, the Gemini 10X, and the Atari SMM804 are included with the package. Drivers for 24-pin printers, the HP Laserjet Plus and Series II, HP Deskjet, and HP, Roland, and HI plotters are available at additional cost.

Migraph, 200 S. 333rd (220), Federal Way, WA 98003, (800) 223-3729, (206) 838-4677.

ENTERTAINMENT SOFTWARE

Joker Software has released *Aussie Joker Poker*, a gambling game of skill and chance for the Atari ST. Up to 90 players can compete against one another and can tailor the deck sizes and rounds per player to suit their preferences. The introduction of the game is being backed by a sweepstakes, featuring $200,000 in cash and prizes. Twenty finalists will be flown to Las Vegas to compete in an *Aussie Joker Poker* competition with a top prize of $100,000. Finalists will be chosen at random from entry blanks received before April 20, 1989.

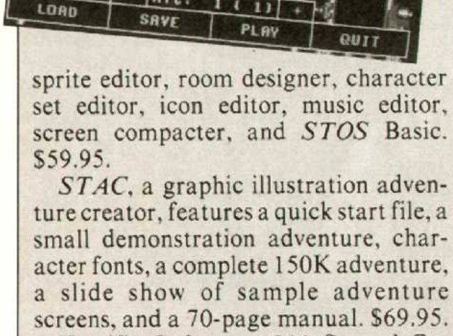

JOKER POKER COMPETITION MODE IS ON !

Deck is 7 up.
5 hands each.
Current high score

Mindscape is the distributor for the Joker line of non-violent entertainment software products in the US and Canada. The game sells for $49.95.

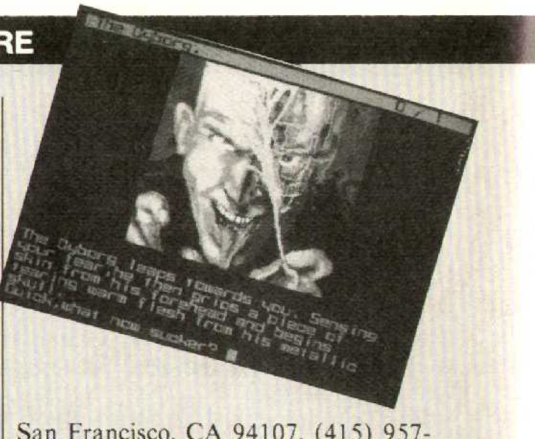
Joker Software, 1407 Market St., 4th Floor, San Francisco, CA 94103, (415) 621-0338, (800) 24-JOKER.

Terrific Software announces the US release of two game creation programs for the Atari ST. *STOS* offers an easy-to-use interface and 320 commands designed to suit any user level. Included with the program are three games, a sprite editor, room designer, character set editor, icon editor, music editor, screen compacter, and *STOS* Basic. $59.95.

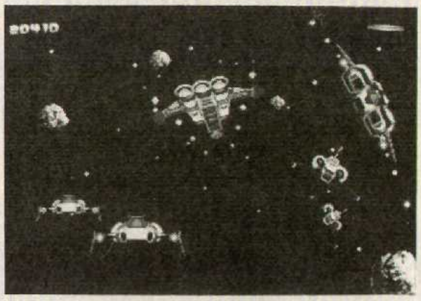


STAC, a graphic illustration adventure creator, features a quick start file, a small demonstration adventure, character fonts, a complete 150K adventure, a slide show of sample adventure screens, and a 70-page manual. $69.95.

Terrific Software, 544 Second St., San Francisco, CA 94107, (415) 957-0886.


The Cyborg leaps towards you. Sensing your fear,he then prics a piece of skin from his forehead and begins tearing warm flesh from his metallic skull.
Quick,what now sucker?

Galactic Conqueror from **Titus Software** combines the action of a coin-op game with the strategy requirements of a wargame. The player is cast as the Betadroid KAL, pilot of Thunder Cloud II, a Hypersustain fighter. His task is to decide which of the 416 planets in the galaxy should be liberated and to enlist the aid of the inhabitants in defeating the enemy invasion. $44.95.



Titus Software, 20432 Corisco St., Chatsworth, CA 91311, (818) 709-3693.

Microdeal has released four new entertainment products for Atari ST computers. *Talespin* is program designed to help users create graphic adventure games. Features include digitized sound effects, an integral art package, and the ability to link multiple text boxes to each drawing. $99.95.

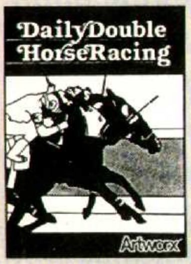
Jug charges the player with the task of eliminating the deadly tumor that is destroying the planetary brain of the planet Piraeus. $39.95.

International Soccer puts a complete soccer game, including weather, team formation, time of day, and uniform color, under the control of the player's joystick. Onscreen referees ensure fair play by one or two players. $39.95.

Zero Gravity is a futuristic version of volleyball played by space voyagers floating in the cargo bay of a spacecraft. The game is joystick-controlled by one player against the computer or by two players competing against each other. It features three difficulty levels. $29.95.

Microdeal, 576 S. Telegraph, Pontiac, MI 48053, (313) 334-8729.

Artworx announces *Daily Double Horse Racing* for the Atari ST. The player receives a racing form, which includes past race histories of 180 horses and 12 jockeys competing in nearly 400 races. With this information, he can handicap the races that are run on the screen. $29.95.

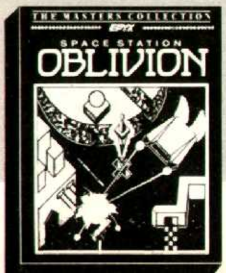


Artworx, 1844 Penfield Rd., Penfield, NY 14526, (716) 385-6120, (800) 828-6573.

Escape from Planet X by **Covox** for Atari 8-bit computers is a text adventure that offers the option of game control through voice command when played with the Covox Voice Master ($89.95) or Voice Master Jr. (49.95). The player trains the program to recognize his voice and then controls the action via the Voicemaster microphone. $19.95.

Covox Inc., 675 Conger St., Eugene, OR 97402, (503) 342-1271.

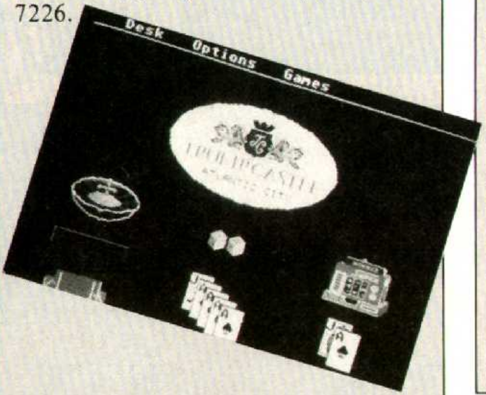
Epyx has announced *Space Station Oblivion*, a sci-fi thriller for advanced game players. The mission is to save the solar system by exploring and transporting required equipment to each of 18 sectors on an uncharted planet before it explodes. $49.95.



Epyx, P.O. Box 8020, Redwood City, CA 94063, (415) 368-3200.

Intracorp announces *Trump Castle*, a casino gambling simulation for Atari ST computers. The package includes six popular casino games—black jack, roulette, craps, keno, video poker, and nine different slot machines—and a certificate good for $250 worth of discount coupons redeemable at Trump Castle on the Bay in Atlantic City. $39.95.

Intracorp, 14160 139th Ct., Miami, FL 33186, (305) 252-9040, (800) 468-7226.



ST Keyboard Stiffener

Regent Software announces the availability of Megatouch, a keyboard stiffener for Atari 520 and 1040 ST computers. The device is intended to give the keyboard of a 520 or 1040 the firmer feel of a Mega ST keyboard.

Average installation time for Megatouch, which sells for $11.95, is "less than 10 minutes."

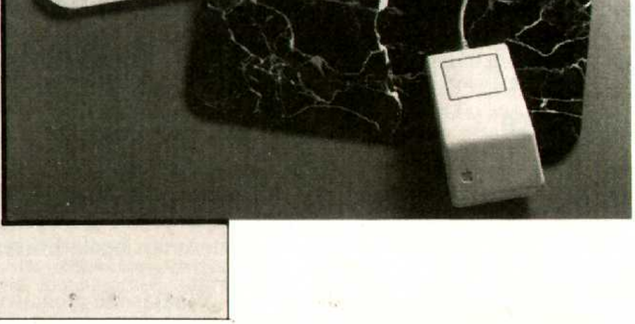
Regent Software, P.O. Box 14628, Long Beach, CA 90803, (213) 439-9664.

Yuppie Mouse Pad

Publishing Ink announces the Executive Pad, a mouse pad that features a decorative granite-, wood-, or marble-look surface on a functional pad.

The laminate surface of the 8½″×11″ pad is said to offer superior speed and traction for both left- and right-handed mousers. Retail price is $17.95.

Publishing Ink, The Computer Giftware Co., 521 State St., Glendale, CA 91203, (818) 500-7857, (800) 543-7326.



Puzzles & Problems

By DAVID H. AHL

In this issue we present a diverse collection of puzzles and problems in logic and arithmetic. All can be solved with pencil and paper, but for several of them, you will find a computer helpful. Answers are on page 74.

Six Beneficiaries

In his will, Jacob Baron left $1 million to his three married daughters and their husbands. But because some were better off than others, he did not leave equal amounts. The daughters received $396,000, distributed such that Jean received $10,000 more than Kate, and May $10,000 more than Jean. Albert Smith got 50% more than his wife, Henry March got as much as his wife, while Thomas Hughes inherited twice as much as his wife. Who did each daughter marry and how much did each inherit?

Sum of Combinations

Which four-digit number is three times the sum of all two-digit combinations that can be made of its digits?

Simple Division by Three

What multi-digit number can be divided by three simply by shifting its last (rightmost) digit to the front (left)?

Simple Division by Four

What multi-digit number can be divided by four simply by shifting its first (leftmost) digit to the rear (right)? There are three solutions.

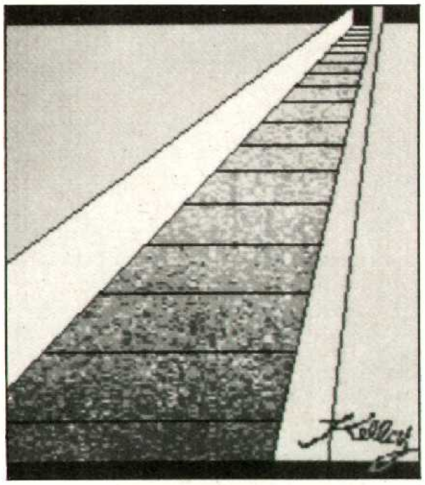
Two Towers

Two towers, one 30 rods, the other 40 rods high, are 50 rods apart. There is a milestone on the straight line between these towers. From the tops of both towers, two crows fly off simultaneously at the same speed toward the milestone, reaching it simultaneously. How far is the milestone from each tower?

Cryptic Peace

In the following cryptic addition, can you determine what integer is represented by each letter?

```
   U S S R
 + U S A
 P E A C E
```



The Escalator

Anxious to get out of the subway, two boys decide to race up the escalator. The tall boy climbs three times as quickly as the shorter boy, and as he runs up, he counts 75 steps. The shorter boy counts only 50 steps. How many steps of the escalator are exposed?

All Numerals From 1 to 9

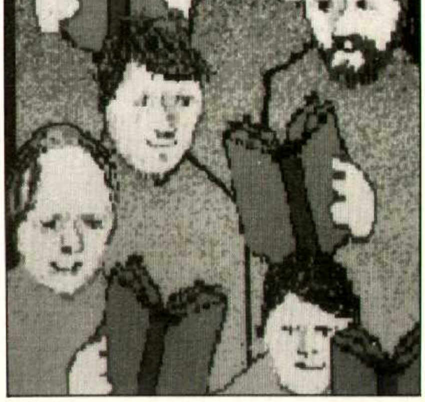
Both of the two products 12×483=5796 and 42×138=5796 contain all the numerals from 1 to 9 and yield the same product. Can you find another pair of products with the same peculiarities?

Six Authors

On the way to a booksellers' convention in New York, six authors find themselves seated in a train with three across facing the other three. Each of the writers—Allen, Baker, Chapman, Davis, Egmont, and Frazer—has a specialty, notably short stories, history, humor, novels, plays, and poetry. Each has brought along one of his books and has lent it to one of the others.

Allen is reading short stories. Chapman is reading a book by the fellow opposite him. Baker is sitting between the author of short stories and the humorist. The writer of short stories is sitting opposite the historian. Davis is reading a play. Baker is the brother-in-law of the novelist. Egmont sits next to the playright. Allen sits in a corner and is not interested in history. Davis sits opposite the novelist. Egmont reads a humorous book. Frazer never reads poems.

What is the specialty of each author?



ENHANCE YOUR IMAGE WITH THE ST SCAN IMAGE SCANNER

FOR YOUR ATARI ST OR MEGA SYSTEM

FLASH!!!
New software
adds copy machine for
Atari SLM804 laser printer.

The flexibility to introduce art into desktop publishing.

When you want to enhance your image, our *ST-SCAN Image Scanner* can transfer your line art, photographs, logos and other graphics into your computer.

Capture any image sharp and clear with resolutions up 300 dots per inch and with 32 shades of grey.

It takes less than 15 seconds

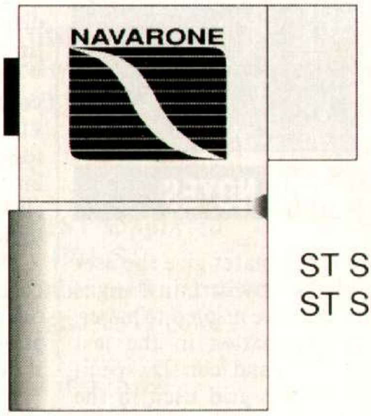
to scan in your image. Once digitized, you can use graphic programs like DEGAS™ and EASYDRAW with Supercharger™ to edit and crop your image.

Compatibility with desktop publishing programs:

You can put your image into final documents with PageStream™ by Softlogik, Publish-ST™ by Timeworks, Calamus™ by ISD, Fleet Street Publisher™ by Mirrorsoft, or save in PostScript to allow direct printing on PostScript devices such as the Linotronics 300™, Apple Laserwriter™, or QMS PS 800™.

Navarone's high speed interface.

Navarone combines the Canon IX-12 or IX-12F™ Image Scanners with its own High Speed Interface that plugs into the cartridge port of your Atari ST or MEGA™.



The *ST SCAN Image Scanner* comes complete with Canon Scanner (flatbed or sheetfed), high speed interface, cable, software and manual.

ST SCAN Flatbed	$1779.00
ST SCAN Sheetfed	$1239.00

Sophisticated, but easy to use software, allows scanning in both line art and half tone mode. The ST SCAN program operates under GEM™ with easy to use click on menus. "Setting up and becoming familiar with the (ST SCAN) system is simplicity itself" (Frank Kofsky, ATARI EXPLORER).

NAVARONE

To order, call toll free
1-800-624-6545
or in California
408-378-8177
or send M.O. plus shipping (call for rates) to Navarone Industries, 454 Kenneth Ave., Campbell, CA 95008. VISA, M.C., C.O.D. welcome. California residents add 7% sales tax.

Prices and availability are subject to change without prior notice. Postscript is a trademark of Adobe; DEGAS is a registered trademark of Batteries Included, Inc.; Softlogik and Publishing Partner are trademarks of Softlogik Corp.;Publish-ST is a registered trademark of Timeworks; Calamus is a registered trademark of ISD Marketing.; Easydraw and Supercharger are registered trademarks of Migraph.; Canon IX-12 and IX-12F is a registered of Canon, Inc., Atari ST is a registered trademark of Atari Corp.; Apple LaserWriter is registered trademark of Apple Computer; QMS PS 800 is a registered trademark of Quality Micro Systems; Linotronic is a registered trademark of Linotype; GEM is a registered trademark of Digital Research Inc.

User Friendly

Hints and out-takes from Atari user groups

By DAVID NOYES

Unquestionably, one of the all-time great third-party software packages for the 8-Bit Atari computers is *Print Shop* by Broderbund. *Print Shop*, along with Broberbund's *Graphics Library* and *Companion* disks and a myriad of other commercial and public domain icon and utility disks, is undoubtedly among the most-used programs in the libraries of many 8-bit Atarians.

Dave Dvorin, of the JACG (Jersey Atari Computer Group), in their July 1988 newsletter, reviewed a recently published book intended to help *Print Shop* users get more out of the program. One caveat: The book never once mentions Atari or Atari computers, but this merely shows the prejudice or myopia of the authors and should not detract from an Atari owner's ability to make use of the book.

The Official Print Shop Handbook

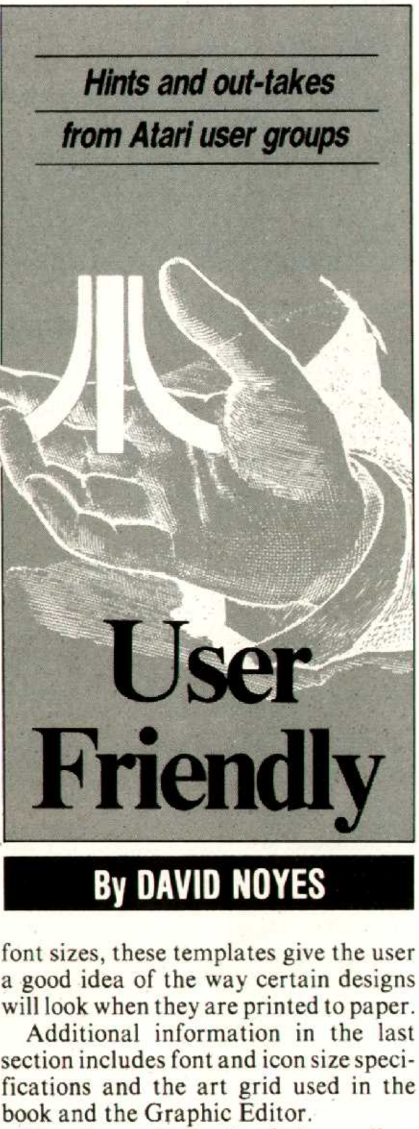
The Official Print Shop Handbook by Randi Benton and Mary Schenck Balcer offers hints for using *The Print Shop* and its family of software add-ons, which includes *Graphics Library 1*, *Graphics Library 2*, *Graphics Library 3*, and *The Print Shop Companion*, regardless of what computer you are using.

The book is a "must" for owners of *The Print Shop*. It is divided into four sections. The first is a description of the book itself, which explains why the book was written, what information is included, and where it can be found, and defines the symbols used in the text. In general, this section tells the reader how to get the most from the book.

The second section describes projects that can be created with *The Print Shop*, its family of software add-ons, and the *Handbook*. The 100-plus applications are categorized as home, party, learning materials, school/organization, and professional. To make this section more valuable, the authors include the following information for each application: the software required, step-by-step instructions, design notes, and alternate ideas.

The third section illustrates more than 60 new and modified icons. To make it easier to create them, they are presented on the same grid used in the Graphic Editor on the *The Print Shop* program disk. For modifying icons, the illustrations clearly mark which pixels are altered. Each illustration indicates the software needed, ideas for applications, and examples.

The last section describes the planning tools with which the authors generated the various icons and applications presented in the book. The tools consist of templates that show the icon sizes used in the placements allowed by *The Print Shop*. Coupled with the available font sizes, these templates give the user a good idea of the way certain designs will look when they are printed to paper.



Additional information in the last section includes font and icon size specifications and the art grid used in the book and the Graphic Editor.

Simply on the basis of the applications examples and additional icons, this book is worth the $16.95 list price. However, the authors' intentions for this book go beyond content. They hope that the reader will "find the ideas in this book useful—and inspiring." And it is true; the real value of this handbook comes from the ideas that the reader generates on his own.

This almost-300-page paperback can be purchased at most major book stores. If you use *The Print Shop*, regardless of whether you have only *The Print Shop* or the entire collection of add-ons, *The Official Print Shop Handbook* is for you.

ST Notes

Now for you 16-bitters, I came across some useful desktop hints in an article by Steve Marshall of the NWPAC in the October, 1988 issue of their excellent newsletter. I'm sure Steve's tips will help put new users on friendlier terms with their STs.

Desktop Tips
by Steve Marshall

If you have just brought home your new ST and find yourself trying to get a handle on windows, mouse clicks, and dragging files, I have some simple tricks that will make your mousing a little easier.

If you have a window showing a directory of a disk in the disk drive open and you swap disks, there is an easier way to get a new directory than closing the window and re-opening it. With the window still open, simply press the Esc key and the directory of the new disk will be displayed in the open window.

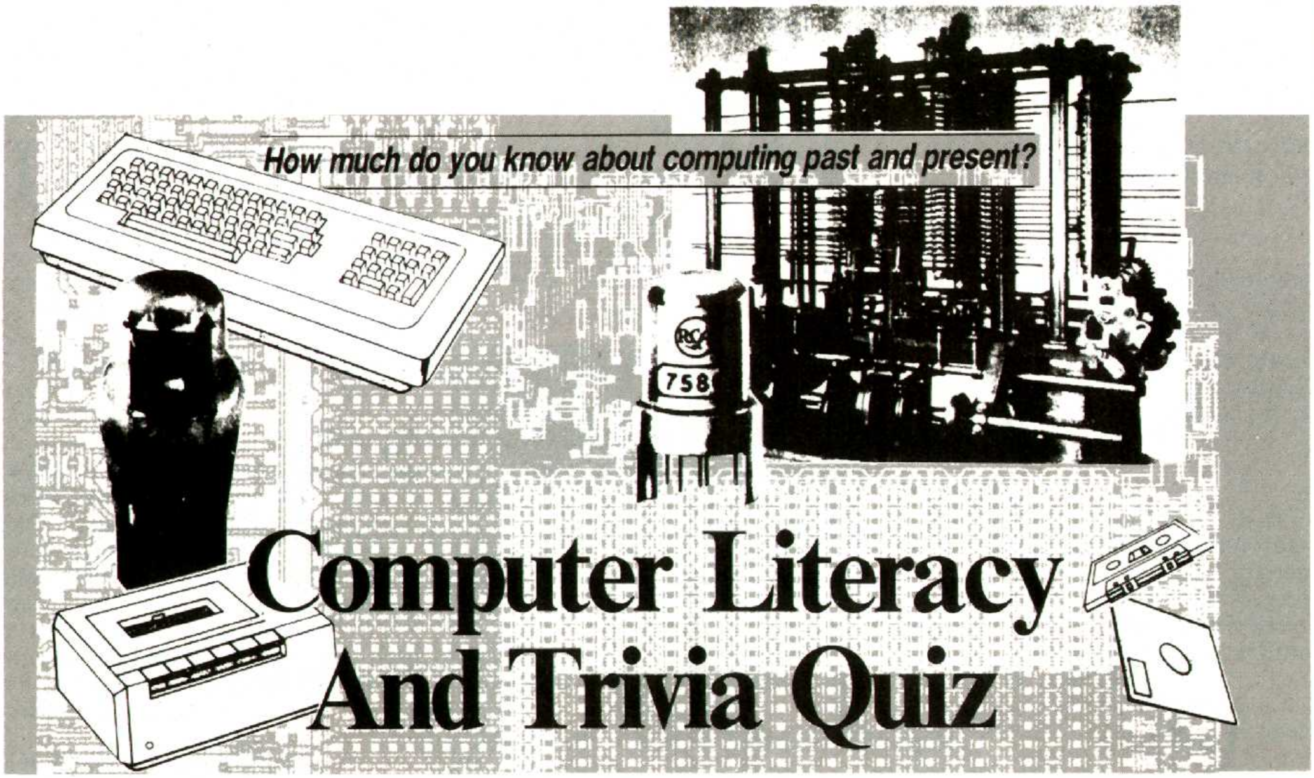
Maybe you have run across this situation: You have about 50 files on your disk and you want to copy file XYZ.TXT into a folder on the same disk, but the folder icon and the file icon can't be displayed on the screen at the same time. What do you do?

Well, you can reorganize the way your files are displayed by selecting VIEW from the desktop menu and trying to sort by size, type, or name. One combination will probably display your file close enough to your folder icon for you to drag it over.

If that doesn't work, simply open another window. You can have up to four different windows open on the desktop at one time, and there is no reason that they can't all be drive A if you want.

Suppose your mouse suddenly bites the dust and you just have to get in and edit a file. Or you have disconnected your mouse to plug in two joysticks to play a little game, not realizing that you need the mouse to start the game! Well, don't fret.

You can move the pointer on the desktop and even click on files without ever touching a mouse. Press the Alternate key and use the arrow keys to move the cursor around the screen. To click (or even double-click), just tap the Insert key (while holding the Alternate key). The result will be the same as pressing the left mouse button. ■



How much do you know about computing past and present?

Computer Literacy And Trivia Quiz

By DAVID H. AHL

How much do you know about the computers of yesterday and today? Take this quick test and find out. No reference sources allowed. Answers are on page 68.

1. The concept and use of punch cards was developed:
   a. Before 1900
   b. Around 1920
   c. Around 1940
   d. Around 1960

2. In the early days of computers, most programming was done in:
   a. Basic
   b. Fortran
   c. Machine language
   d. Univac

3. In the Turing test,
   a. a computer simulates a business environment to train executives in decision-making.
   b. a person imitates a computer to find program errors.
   c. a computer simulates a complex situation to provide a detailed study of alternative approaches.
   d. a computer pretends to be human to demonstrate artificial intelligence.

4. The largest user of computers in the U.S. Government is by:
   a. Internal Revenue Service
   b. Census Bureau
   c. Department of Defense and the military
   d. Congress

The next six questions are true/false:

5. NCIC is a method whereby checks printed with a special ink can be read by machine.

6. By 1950, more than 1000 electronic digital computers had been manufactured and put into service.

7. Large computer programs are apt to contain undetected errors, even after the programs have been in use for several years.

8. A Programming Language (APL) was originally conceived as a unifying mathematical notation.

9. In Boolean arithmetic, the value of 1 AND 1 is 11.

10. The function of a compiler is to translate a higher order language program into machine code.

11. Using an 8-bit code (such as transmitted by a modem), how many characters can be represented?
   a. 8
   b. 32
   c. 64
   d. 256
   e. 512

12. Who was the first U.S. President to use a word processor?
   a. Jack Kennedy
   b. Richard Nixon
   c. Gerald Ford
   d. Jimmy Carter
   e. Ronald Reagan

13. Which was the first personal computer?
   a. Altair 8800
   b. Apple I
   c. Commodore PET
   d. Kenbak 1
   e. Scelbi 8H

14. The largest computer manufacturing plant in the world (1989) is located in:
a. Japan
b. Korea
c. Taiwan
d. United States

15. Capt. Grace Hopper coined the term "computer bug" to refer to a ______ found in the Harvard Mark I computer.
a. beetle
b. moth
c. piece of foil shaped like a bug
d. elusive programming error

16. Which of the following were founded by Steve Wozniak?
a. Apple Computer
b. Atari Corp.
c. Cloud 9
d. Dial-a-Joke

17. Which of the following were founded by Nolan Bushnell?
a. Atari Corp.
b. Chuck-E-Cheese Pizzatime Theater
c. Axlon Inc.
d. Worlds of Wonder

18. What was the first million selling computer book?
a. Ahl: *Basic Computer Games*
b. Freiberger & Swaine: *Fire in the Valley*
c. Kemeny & Kurtz: *Basic Programming*
d. Norton: *Programmers Guide to the IBM PC*

19. Which of the following is *not* a computer?
a. MANIAC
b. SILLIAC
c. BRAINIAC
d. ILLIAC
e. JOHNIAC

20. With its 40 racks of equipment and 20,000 vacuum tubes, EINAC ran up daily electric bills of:
a. $1 (during WWII, wartime contractors were charged only a token amount for electricity)
b. $20
c. $60
d. $1000

21. One of the earliest methods for the mass storage of programs and data used tanks containing:
a. Mercury
b. Water
c. Chicken soup
d. Liquid hydrogen

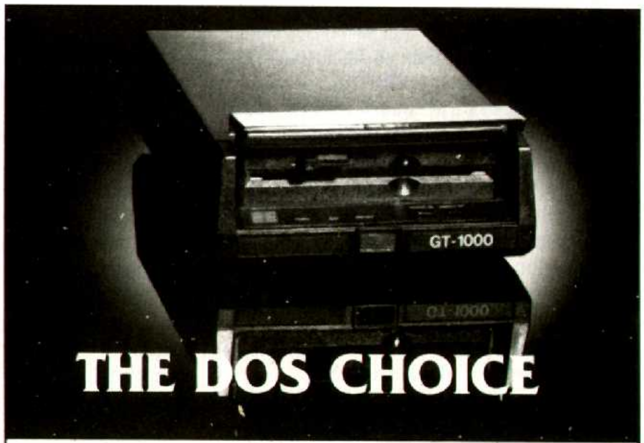
22. In 1900, before the computer age, astronomers calculated the orbits of astronomical bodies by hand to a precision of:
a. 5 places
b. 9 places
c. 20 places
d. 100 places

23. Herman Hollerith organized the Tabulating Machine Corp. which later became IBM. His punched cards were first used by which government organization:
a. Internal Revenue Service
b. Census Bureau
c. Department of Defense
d. Congress

24. Cybernetics, a term used in control theory, automation, and computer programming, was coined by:
a. Vannevar Bush
b. Thomas Watson
c. Samuel Morse
d. Norbert Wiener

25. An early electronic computer was developed at the Morse School of Electrical Engineering at the University of Pennsylvania by:
a. Martin & Lewis
b. Mauchly & Eckert
c. Sperry & Rand
d. Aiken & Wiener

Scoring: 6 or fewer right—you should stick to pencil and paper; 7 to 12—you're still in the calculator era; 13 to 18—you should read *Atari Explorer* more often; 19 to 24—you should write for *Atari Explorer*; 25—is your name Bill Gates? ■



TOS 1.4

Updates to the ST OS fix some old problems and add some new capabilities

By WAYNE RAMSEY

The first year of existence for an operating system can be considered a period of technical and marketing probation, during which pioneer developers and users test its capabilities and discover its severest bugs and limitations. By the end of this period, the OS has almost always been profoundly changed. Often, over a relatively short period of time, the first release is abandoned completely—the second becoming the real base for downward-compatibility with software. So it was with Microsoft MS-DOS and with the first release of TOS for the ST.

With the release of a "significant first revision," however, the OS usually becomes a more or less stable software platform—at least from the user's perspective. This is particularly true when the OS has been committed to ROM as in the case of the ST. Unless subsequent changes add significant capabilities to the system, they have a way of slipping between the cracks. Users of prior versions will ignore them, because developers typically render their new products downward-compatible with all revisions of the OS, and new users do not notice them because they lack a standard of comparison.

It was thus practically without fanfare that on August 17, 1988, Atari began releasing information to developers regarding TOS 1.4—the most recent revision of the ST operating system. The new release comprises significant revisions to the overall functionality of the system, including changes to the GEM desktop, EMDOS, the AES, the VDI, and the BIOS.

At press time, it was not known when TOS 1.4 would be ready for burning into ROMs, nor when it would be released to developers or to the general public. What appears below is an abstract of working documents and is subject to change.

Desktop Changes

A total of 34 more or less significant changes are being made to the functions of the GEM desktop. Autobooting of GEM and TOS applications (though not TTP programs) is now possible, through a revision of the Install Application function. Installed documents and applications no longer need to reside in the root directory of the boot disk, but may now reside, separately or together, in any folder on the system.

The function of the desktop copy operation (particularly single-disk copy) has been improved, and a Move File(s) function, which is accessed by highlighting the desired files and pressing Control before dragging them to the destination, has been added.

Disks are now formatted with a fully MS-DOS compatible boot sector, eliminating certain compatibility problems with IBM file transfers. Numerous small changes have been made to most "work in progress" and "error" dialogs, permitting more sensible management of desktop functions.

In addition, many previous uninterruptible operations can now be stopped via the Undo key, and media changes are now handled much more reliably and sensibly.

AES and VDI Changes

Of the eight changes made to the GEM Application Environment Services package, the most apparent is the complete reworking of the GEM file selector. Programs can now label the top line of the selector to reflect operations in progress, so the confusion of having different program functions (e.g., save file and delete file) use identical file selector dialogs can be eliminated. A new AES function, fsel_exinput( ) has been added to support this feature. Sixteen drive select buttons have been added, eliminating the laborious typing previously required to change the path to look at a new drive. And limits on the number of files that can be displayed by the selector and on the length of pathnames have been removed.

Other minor enhancements and fixes to the AES include the capability to combine editable and non-editable text in the same dialog box field and improvement of certain cosmetic aspects of window management. An All Windows Closed function, wind_new( ), has also been added and the documentation of certain important functions, particularly those comprising the AES "shell library," clarified.

A total of 34 more or less significant changes have been made to the functions of the GEM desktop.

Only three changes were made in GEM Virtual Device Interface subsystem. Of these, the most important is probably the expansion of input array (Ptsin) size to let the system draw polygons with up to 512 vertices. The vq_mouse( ) function has been reworked, and certain minor bugs eliminated.

GEMDOS (BDOS/BIOS/XBIOS) Changes

The internal memory management functions of the system have been improved, eliminating the so-called 40 folder bug. A history and explanation of this widely misunderstood bug is in order. On boot-up, the ST allocates 80 memory blocks to an internal pool that is subsequently drawn upon in several ways. Two blocks are occupied by each active folder, one block by each open file, and a quarter-block by each chunk of system memory allocated (or subsequently freed) by Malloc( ), the system's internal memory manager (Malloc( ) is called automatically when a program is loaded and may also be called by applications).

Formerly, a folder was considered

active when it was merely seen—getting a directory of a disk caused all the folders there to take up blocks in the pool. Also, blocks were not freed back to the system pool when folders became inactive, when files were closed, or when processes were terminated.

In TOS 1.4, the definition of an active folder has been refined. A folder now takes up space in the system pool if a) it is the root directory of a logical device, b) it is the current directory of an active process (each active process has a current directory on each logical device), c) it contains currently open files, or d) it is on the path of an active folder (i.e., if folder C:\WORK\DOCS\JANUARY is active, directory DOCS\ is active, as are directory WORK\ and the root directory, C:\). If a folder is active for more than one reason, it still occupies only two blocks in the system pool. Also, when folders become inactive, files are closed, or processes terminated, the blocks used to represent them are freed back to the system pool.

While it is still possible to consume all of system memory, these improvements have expanded the practical limits to the point where most users will never come up against them. Also, if absolutely necessary, the number of blocks allocated to the system pool can be increased further using the program FOLDRXXX.PRG, distributed with Atari's HDX hard-disk utilities package and widely available on bulletin boards. To use FOLDRXXX.PRG, you change the XXX to reflect the number of folders (at two blocks per folder) you wish to add, then place the program in your AUTO folder for execution at boot-up. Changing the name to FOLDR100.PRG, for example, will add 200 blocks to the system pool—room for 100 additional active folders, 800 memory chunks, or any combination thereof.

The performance of many GEMDOS functions has been improved. In particular, the FAT-search algorithm has been altered so that disk performance degrades in direct, rather than multiplicative proportion to load. Sector buffering has also been improved and corrected. These two improvements enhance hard disk performance dramatically.

Also, folders can now be renamed; the "duplicate filename" problem has been eliminated; and numerous other subtle improvements and bug fixes have been implemented in the GEMDOS functions used to create, delete, and change the attributes of files.

Console I/O functions can now be redirected with greater efficiency than ever, and keyboard repeat functions have been improved. Warm and cold reboot are now available from the keyboard (via Control-Alternate-Delete and Control-Alternate-RShiftt-Delete, respectively).

Process execution facilities have been updated and are now compatible with the Mega ROMs. RS-232 configure functions have been made more consistent, disk formatting is now 100% IBM PC-compatible. Finally, "media change" status is more reliably recognized.

In summary, this is a fairly impressive upgrade—made even more impressive by the fact that the great majority of ST software titles will not be rendered incompatible by it. According to Atari, only a few classes of program may have to be revised to be compatible with TOS 1.4—particularly programs that made use of undocumented portions of the DTA (Disk Transfer Address) structure used with Fsfirst( )/Fsnext( ).

Users can expect, however, to see certain programs re-released to take advantage of new and repaired system features. For example, since TOS 1.4 now handles archive attribute bits correctly, popular hard disk backup programs can be altered to make use of this feature on new machines. ■

***This is a fairly impressive upgrade** —made even more impressive by the fact that the great majority of ST software titles will not be rendered incompatible by it.*

RAMdisk Rescue

ST HELP KEY

The contents of a "reset-proof" RAMdisk often survive a software lockup or system crash. But if the disk you have in your drive when you press Reset doesn't contain either the RAMdisk program or a DESKTOP.INF file that includes installation specs for a RAMdisk icon, you may regain control of your system only to discover that you have no way of accessing the contents of the RAMdisk. Chances are, however, that the are still present . . . just not accounted for.

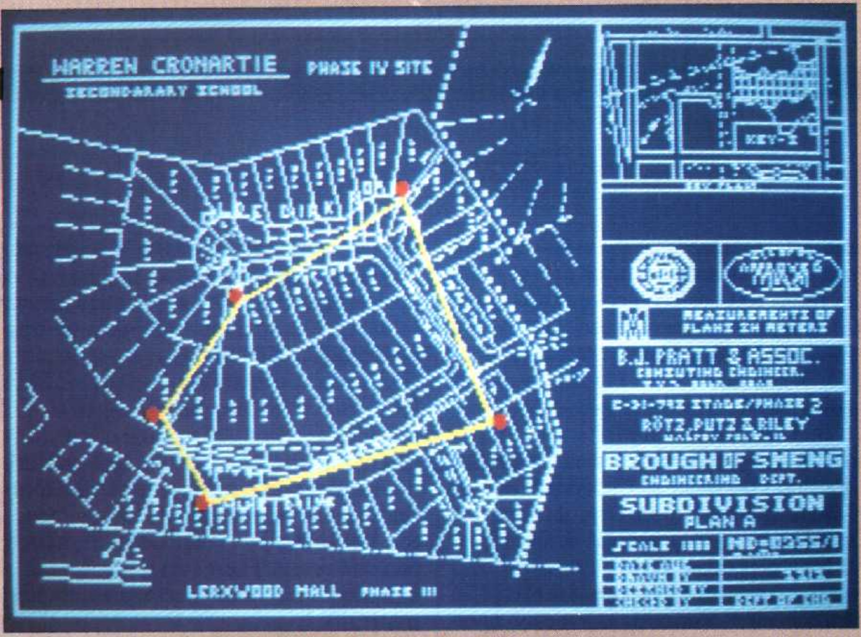
To reclaim the contents of the RAMdisk, first, try reinstalling the RAMdisk icon using the Install Disk Drive selection from the desktop Options menu. This is done as follows:

- Single-click on one of your floppy disk icons. The icon will be highlighted.
- Select Install Disk Drive from the Options menu. A dialog box containing the name and letter of the selected drive will appear.
- Alter the name and letter to conform to those you were using for the RAMdisk.
- Click on Install. A new disk icon—hopefully connected to your lost RAMdisk—will appear on the desktop. Double-click on it to see if you can recover your data.

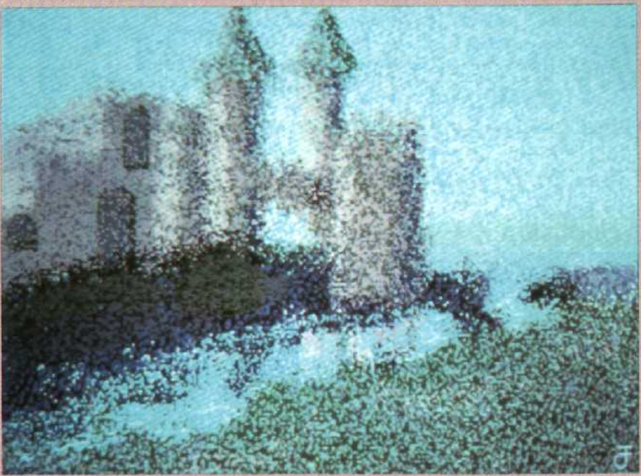
If this doesn't work, try putting a disk with the original RAMdisk installation program on it in your boot drive, and press Reset again. Sometimes, attempting to re-initialize the RAMdisk will repair the damage that disconnected it from the system in the first place, allowing you to recover some or all of your files.

From *The Atari ST Book of Tips, Instructions, Secrets and Hints*, © 1988 by Ralph C. Turner, Index Legalis Publishing Co., P.O. Box 1822-20, Fairfield, IA 52556, (515) 472-2293.

3 Phases (Degas) by Dalibor Lanik of Belgrade, Yugoslavia.



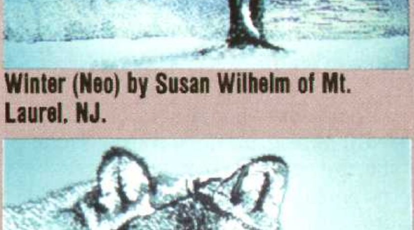
Subdivisions (Neo) by Michael Riley of Woodridge, IL.



Impressionistic Camelot (Degas) by John Finnegan of Greenville, NC.

Hulk (Degas) by Dusan Dimitrijevich of Belgrade, Yugoslavia.

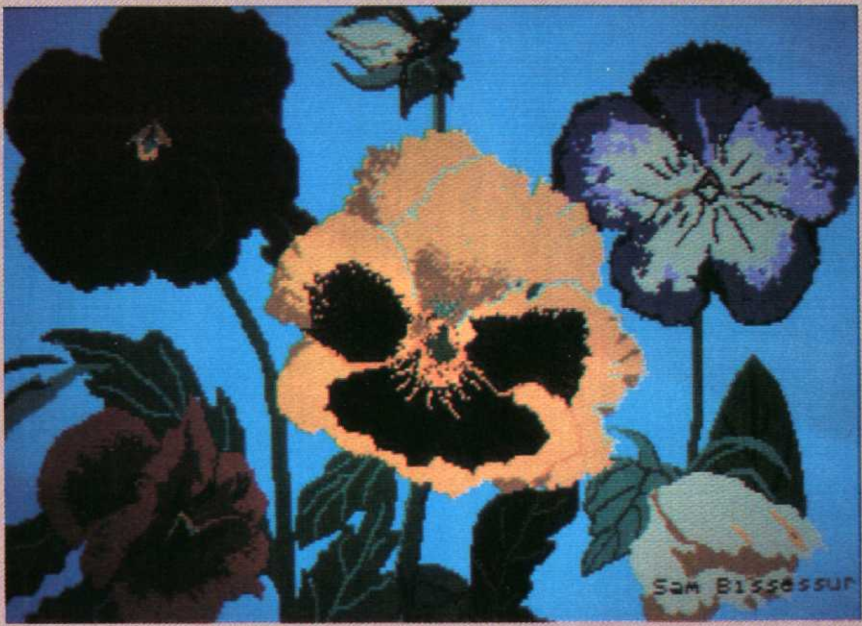
Winter (Neo) by Susan Wilhelm of Mt. Laurel, NJ.



Arctic Wolf (Degas) by David Board of Fairmont, WV.

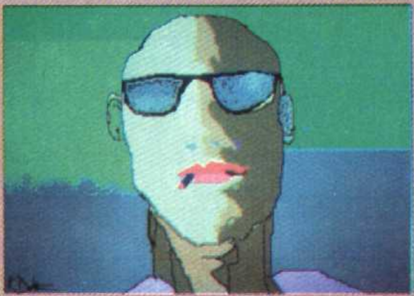
MISSION
SAN
JUAN
CAPISTRANO

Capistrano (Neo) by Chris Brisson of Long Beach, CA.



Pansy (Neo) by Sam Bissessur of Nanaimo, BC.

Blast Off (Degas) by Edwin Galarza of Brooklyn, NY.



Breeze (Degas) by Kent Dalton of Marlboro, MA.

Graphics Gallery

The contest for this issue drew more than 100 entries from places as far flung as Yugoslavia, Sweden, Italy, and Australia. Moreover, the wide diversity of subjects was fascinating. (We were getting just a little tired of space shuttles, hot cars, rock musicians, and Atari computers.)

With so many excellent entries, it was quite impossible to narrow the field to ten winners, so we chose the 17 images you see on these pages. Although *Degas* finalists outnumbered *NeoChrome* winners by ten to seven, the two images we judged best were both done using *NeoChrome*.

Winning a three-year subscription for his marvelous image of a bouquet of pansies was Sam Bissessur of Nanaimo, BC. Also the winner of a three-year subscription is Chris Brisson of Long Beach, CA, who submitted a lovely scene of the courtyard of Mission San Juan Capistrano.

Contest Rules

We invite you to enter our ongoing contest, but *please abide by the following rules*.

- Submit your image on disk in *NeoChrome*, *Degas*, or Tiny format. Print your name and address on the disk.
- Include a self-addressed, stamped envelope (preferably #10 size) with 45 cents postage for the return of your disk. We will return your disk with *ten new images*.
- Include on an 8½"×11" sheet of paper your name and address, the file names of your images, and the following statement: "I certify that the image(s) submitted is (are) my own personal work and that no portion was copied from any image belonging to another person or organization or from copyrighted printed or video material. I give *Atari Explorer* the right to print my image(s), use it (them) in promotional material, and/or distribute it (them) via telecommunications service or BBS."
- Winners will receive a subscription to *Atari Explorer*. If you are a already a subscriber, include an address label or copy so we can extend the correct subscription if you win. ■

By DAVID H. AHL

Picasso's Three Musicians (Neo) by Robert Rassmesain of La Mirada, CA.

Sundown (Neo) by Michael Green of S. San Francisco, CA.

Help
me
Vince

"Help Me" (Neo) by Vince Villanueva of Coral Springs, FL.

City Scene (Degas) by Matt McIrvin of Chantilly, VA.

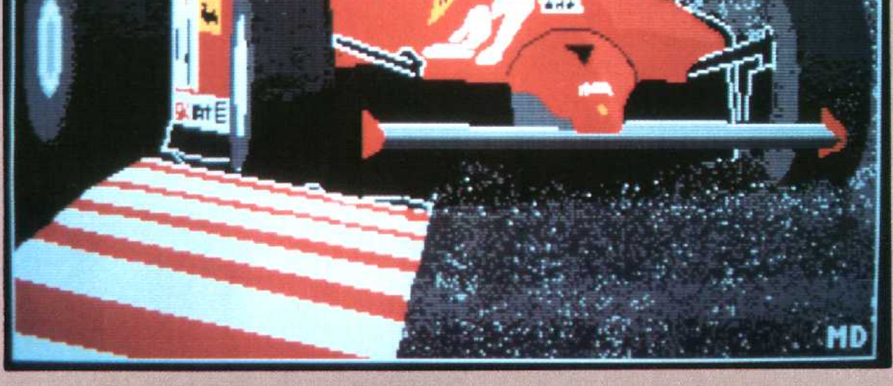
TEENAGE
MUTANT
NINJA
ALLIGATOR
GEEZ--
I DUNNO
MIKE...
AW, YA
LOOK
GREAT
K. KIMBALL

Alligator cartoon (Degas) by Keith Kimball of West Monroe, LA.

Graphics Gallery

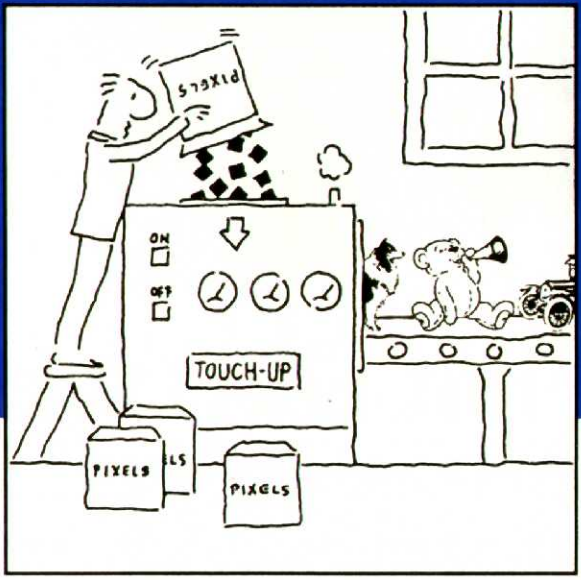
Bullfrog (Degas) by Jake Millspaugh of Fletcher, NC.

Gilles Villeneuve (Degas) by Montanari Davide of Faenza, Italy.



Give your desktop graphics that professional touch.

Introducing Migraph Touch·Up™



Migraph Touch-Up is the complete design tool for high-resolution monochrome images.

Create, edit, enhance—you can do it all with Touch-Up.

And do it better, because Touch-Up is the first "virtual page" graphics program for the Atari ST.® A sizeable advantage indeed!

Touch-Up can handle bit-mapped images of any size and resolution (based on available memory in your computer). Which means you can now produce pixel-perfect images for all your publishing projects.

Migraph Touch-Up. A powerful tool for professional-quality results.

Ask your dealer for a demonstration of Touch-Up, or call Migraph's toll-free number for more details.

MIGRAPH

200 S. 333rd St., Suite 220 Federal Way, WA 98003 (800) 223-3729 (206) 838-4677

Film Director "brings a work of art to life."

Art and Film Director

System: Atari ST
Required equipment: Color monitor
Copy protection: No
Summary: An art and animation program with perhaps more features than the average user needs
Price: $79.95
Manufacturer:
Epyx
P.O. Box 8020
Redwood City, CA 94063
(415) 368-3200

Art and Film Director

Epyx releases a multi-featured art and animation program for the ST

By KAREN KANE

I have always liked graphics programs. Even though I'm not the kind of person who would pull out a sketch pad or dabble with paints, I do enjoy scribbling and smearing shapes and colors on the computer screen. It is somehow easier to accept (and correct) mistakes on-screen, and the process of learning how to use the program makes me less self-conscious about the results.

My latest graphic experiments have been with *Art and Film Director*, a new release from Epyx. The software provides a range of tools for drawing and animating images sophisticated enough to lure computer artists and includes step-by-step tutorials that are clear enough to ease beginners along.

The package contains two programs—Art Director and Film Director. Art Director includes the features you would expect to find in any good drawing program and many more. Using up to 16 colors at once, you can, among other things, spray, zoom, outline, stamp shapes, create mirror images, define your own brush, fill a defined area, draw freehand with a variety of pencil points, and add text to a picture from a selection of font. Special effects rotate, bend, rescale, and otherwise distort your image, and color cycling adds a little flash.

Art Director

One of the features that sets Art Director apart from other graphics packages is the ability to define any kind of brush you desire. You can cut out a block of color, an entire image, or a piece of text from your canvas and paint with it. To create shadows, you simply select the Silhou option, which turns your brush into a silhouette of solid color.

You can also double, halve, flip, or rotate brushes for a variety of effects. If you get carried away and run out of room on your canvas, just scroll in any direction to reveal fresh screen waiting for design. Or switch to another page; you can have up to 16 pages in memory, and if you wish, each one can have up to seven 16-color palettes.

I like playing with the colors in my pictures. You can daw a beach scene with natural tones and then change to neon greens and acid yellows. Or you can use the melt function to blend strokes of blue and green in your sea. Or you can scrape away one layer of color to reveal others below. The possibilities are endless.

Art Director even offers a hint of animation with sprites and spins, but the program works best as a tool for creating handpainted images to be used in Film Director.

Film Director

Film Director is what really distinguishes this package from many others on the market for the ST. It is a powerful program that requires some concentration (and frustration) to learn to use effectively. The tutorial guides you through the steps of cel animation, but if you have never used an animation program, you may have some difficulty getting started.

Film Director isn't easy to navigate without the manual, and the manual isn't easy to follow. But once you learn the terminology (*pattern, group, actor*, etc.), you can start creating cels with components of images, layer the cels to design frames, and sequence the frames to produce the effect of animation.

Tools, such as tweening, can speed this frame-by-frame process. You simply define the first and last frames and determine the number of frames you want between them, and the computer creates them for you.

Editing tools help you perfect your animations, and several excellent demo films illustrate the effects of stacking animated sequences. In one, a park is bustling with activity—joggers, strollers, dogs, and butterflies move through an outdoor setting. It is easy to see how each separate activity was created and combined to create the film, which ends with a woman slipping on a stone and falling into a shark-infested pool.

Even with the interpolation feature, creating animation is a tedious process. To design a simple seaside film with seagulls diving for fish, for example, you must first draw the character positions and background with Art Director. You must show the front, side, and rear views of a seagull with wings up, down, and horizontal. Then you rotate the side view 45 degrees and flip the image—it is important that you consider all of the different movements your characters will make. A seagull, for instance, drops its legs as it approaches the water but tucks them away when in flight.

A cartoon with cel animation created with Film Director.

Once you have recorded all the positions you want, you switch to Film Director, which allows you to cut out and define each of your gull parts (or *patterns*) and piece them together in different combinations to form what the program calls *groups*.

Groups are actually single cels built from several pattern cels. To make your gull fly, you must design a number of group cels, each with the bird in a different flying position, and define a path for the sequence to follow. When that is done, you have created an *actor*.

If you want smoother motion between two phases of the bird's flight, you can instruct the program to create a number of intermediate phases automatically. Then all you have to do is place the actor—a swooping bird—on a stage of sea and sky, and you have a film.

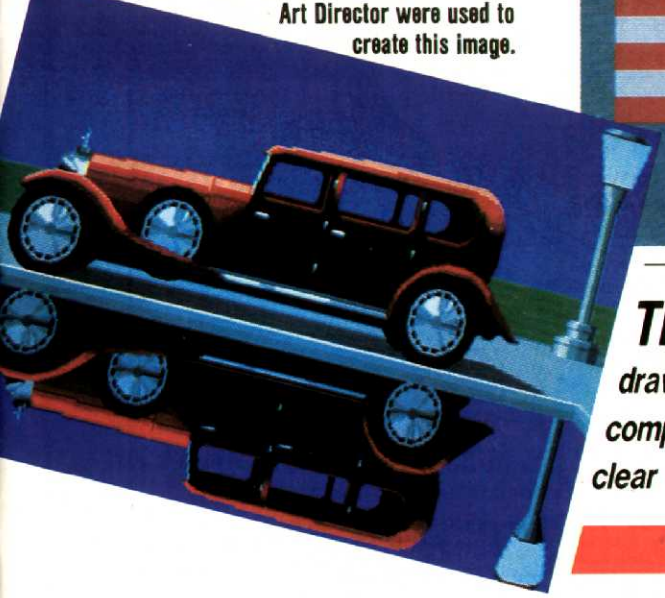
To add another gull gliding high in the sky, you have only to repeat the same steps. As you create more bird actors, you can decide where to place them and whether the fat gull flies in front of or below the dark gray gull. Add squawking noises and whatever else inspires you.

Keep in mind, however, that each of these steps involves many more details (and many more mouse clicks) than I have described. I barely have the patience to write about it, let alone to do it. But if you really want to create a graphic masterpiece (even if it's just for kicks), the tedium will pay off when you watch Jonathan Livingston soar across your screen.

A Few Negative Notes

For all its power and flexibility, *Art and Film Director* is still a little quirky; the operations are not as intuitive as I would like. In Art Director, for example, you can add text to a picture, but if you accidentally left-click before you

The creator of this image made good use of the Bend, Bulge, Shading, and Perspective features of Art Director.


The Perspective and Shading features of Art Director were used to create this image.

***The software provides a range** of tools for drawing and animating images sophisticated enough to lure computer artists and includes step-by-step tutorials that are clear enough to ease beginners along.*

If you want smoother motion ***between two phases of the bird's flight, you can instruct the program to create a number of intermediate phases automatically.***

type in your text, a little NO sign appears. If you type in a letter anyway, your picture disappears.

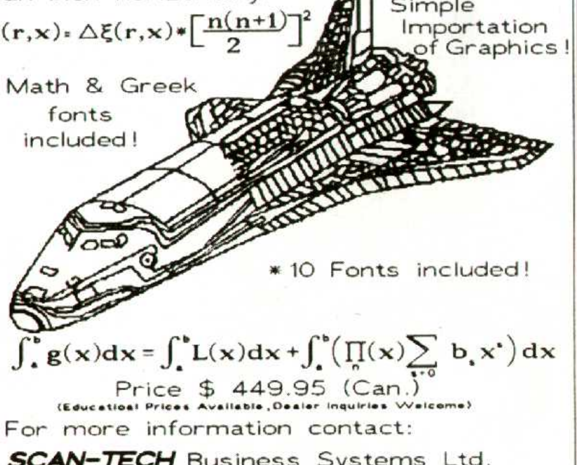
I discovered what was going on the hard way. I thought I had completely lost a drawing I was working on, but discovered later (after it was too late to salvage my masterpiece) that the letters I had typed in were actually keyboard commands. I had switched to another page that happened to be blank. Typing the letter A would have brought back my canvas.

This type of flexibility in a program can sometimes cause more confusion than it's worth. I prefer to use the mouse to navigate through the program, so I shouldn't have to worry about the effects of touching an active keyboard. I certainly think that the keyboard should be available for those who prefer to enter their commands that way, but having both command modes active at once doesn't seem like much of an advantage.

In many ways the whole program is like that. I frequently found myself stumbling over the plethora of options, but as I said at the beginning, I am strictly an amateur—a serious user may covet every one of the options that got in my way.

I put the program away for a week, and when I came back to it I had to relearn many of the procedures, a fact that confirmed my opinion that the commands are not as intuitive as they might be. Another minor complaint I have is that the program offers hundreds of tools for speeding up the painting and animation process, yet I often had to wait for the tools themselves to respond to my instructions. I hate to be so speed greedy, but even split seconds between mouse-clicks can add up quickly.

To take advantage of the full range of features offered by *Art and Film Director* requires practice, patience, and planning. The best way to ensure success, I found, was to visualize the final product, whether a cartoon, a business presentation, or an ad for a software package, then assemble the actors and scenery with artwork from *Art Director*, *NeoChrome*, or *Degas Elite*. The finishing touch can be music from files supplied with the program, and you can then record your masterpiece on videocassette for all to see. ■

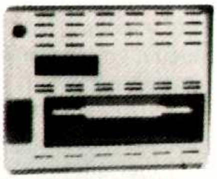


Astra gives up !!!

No we are not going out of business...
We are going out for business !!!

The demise of many Atari dealers, and the reluctance of many more to carry much inventory forces us to announce:

Astra Systems goes **Factory Direct !**



SYSTEM HD+

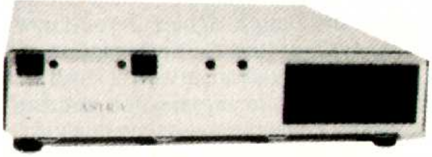
Our original offering for the ST computers is now available in 20, 30, and 40 megabyte capacities. With precision double sided 3 1/2" microfloppy and its tough construction it ranks as the one to beat.

SYSTEM HD+ 20	**$849.95**
SYSTEM HD+ 30	**$919.95**
SYSTEM HD+ 40	**$1069.95**

RM 60/120 & RM 80/160

Our rack mount units for the MIDI musician fit portable and permanent racks.

RM 60	**$1295.00**
RM 60/120	**$2095.00**
RM 80	**$1995.00**
RM 80/160	**$3495.00**
EXPANSION 60	**$ 875.00**
EXPANSION 80	**$1595.00**



THE EXPANDER

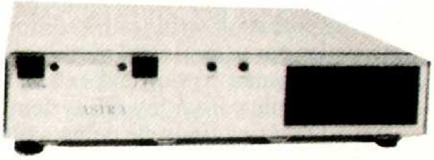
A truly expandable hard disk system. Not a chainable one as many competitors are offering. Contains space, cooling, and sufficient power supply to handle up to 120 megabytes of storage inside the under-monitor case. Built-in surge protection for its 4 AC outlets. Controls the CPU and peripherals from the front of the unit. Optional precision double sided 3 1/2" microfloppy. Twenty, thirty, and forty meg units.

EXPANDER 20	**$759.95**
EXPANDER 30	**$819.95**
EXPANDER 40	**$999.00**

VISA AND MASTERCARD ACCEPTED

MASTERCARD VISA

CALL (714) 549-2141
FAX (714) 546-1612
BBS (714) 546-5956



PRO MODEL

Four AC outlets control CPU and peripherals from front of unit. Available in 40, 60, and 80 megabyte units. Attractive under-monitor case. Built tough to take it !

PRO MODEL 40	**$899,95**
PRO MODEL 60	**$999.95**
PRO MODEL 80	**$1499.95**

SW2 MONITOR SWITCH

Switches both the power and the signal ! True one button switching. Faster and safer. Audio out and composite out (FM models only). Call for information on other exciting MODULAR units.

SW2	**$50.00**

Southern California Astra Systems Inc.
2500-L So. Fairview, Santa Ana, CA. 92704-9869

Revolver

The program that will change the way you use your ST

By JOHN JAINSCHIGG

In the process of writing and editing articles for *Atari Explorer*, I require yeoman service and extraordinary versatility from my ST system. A typical morning finds me doing a bit of word processing, a little data formatting and compression, maybe some compiled-language programming, a few graphics, a smidgen of database access, and, inevitably, some telecommunication.

A lot of this work is done in a "back and forth" manner—editing an article, uploading it, then doing some graphics, compressing them, uploading the compressed files, etc. So naturally, I am always looking for new ways to jump more quickly from one task to another.

Like most ST power users, I have implemented a variety of solutions in hardware and software—a hard disk, a RAMdisk, a print spooler, various graphic speedup programs, alternative file-selectors, and so on—anything to make the already bunny-quick 1040 kick a little more righteous you-know-what. All of these improvements increase the speed with which I can get out of one major application and into another. Unfortunately, they don't save me any steps, which means they don't attack my *real* problem—how to switch back and forth between tasks without losing my place.

"Get a multitasking OS," you say? But multitasking isn't what I need. I don't have to recalculate huge spreadsheets or generate mass-mailings in background mode, while keeping my shoulder to the wheel in the foreground. All I need, for example, is to be able to bounce back and forth between an article I'm writing in *1st Word Plus* and the diagram I'm drawing for it with *Degas*, checking the one against the other until I'm sure both are right. But what a drag to have to do this in the conventional manner! I have to save my article, quit *Word Plus*, switch to low-res, load *Degas*, load the diagram . . . my mouse-finger goes numb just thinking about it.

Well, I am happy to report that today I found the answer: a little program called *Revolver* by Intersect Software. *Revolver* calls itself a "switcher," which is not entirely accurate. In my lexicon, a switcher is a kind of user-friendly monitor or debugger that lets you load multiple applications, then (hopefully) jump from one to another via hotkey. True enough, *Revolver* lets you do that, in a sense—but not in the way a switcher would.

You see, the first generation switcher was a very unreliable beast. It couldn't

Revolver

System: Atari ST

Version reviewed: 1.0

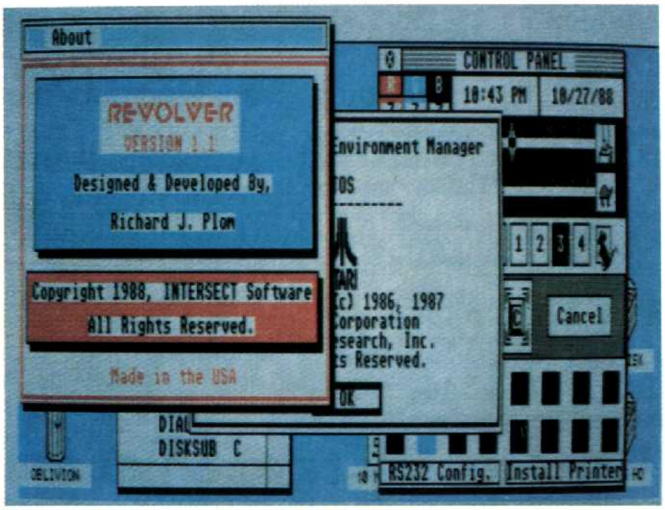
Required equipment: None, but the more RAM the better

Copy protection: None

Summary: Powerful and versatile "switcher" with built-in utilities. The answer to many users' productivity prayers.

Price: $49.95

Manufacturer:
Intersect Software
2828 Clark Rd., Ste. 10
Sarasota, FL 34231
(800) 826-0130
(813) 923-8774


About

REVOLVER
VERSION 1.1

Designed & Developed By,
Richard J. Plom

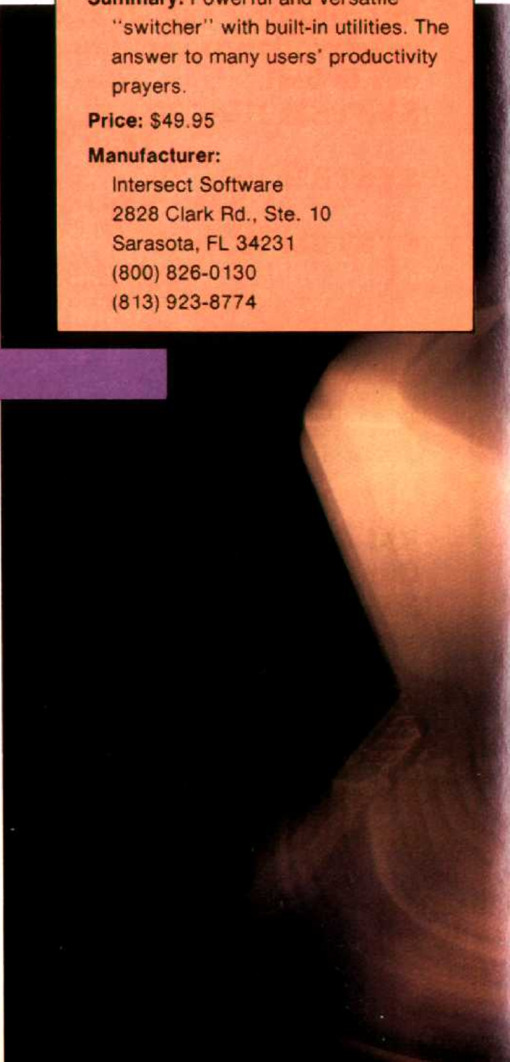
Copyright 1988, INTERSECT Software
All Rights Reserved.

Made in the USA

CONTROL PANEL
10:43 PM 10/27/88

Environment Manager
TOS

(c) 1986, 1987
Corporation
search, Inc.
s Reserved.



tolerate client programs that insisted on playing with hardware or interrupts or wanted to write directly to screen RAM instead of using high-level OS calls—heaven forfend! And because the switcher switched only applications, not environments, it could support only programs that could run under identical runtime configurations.

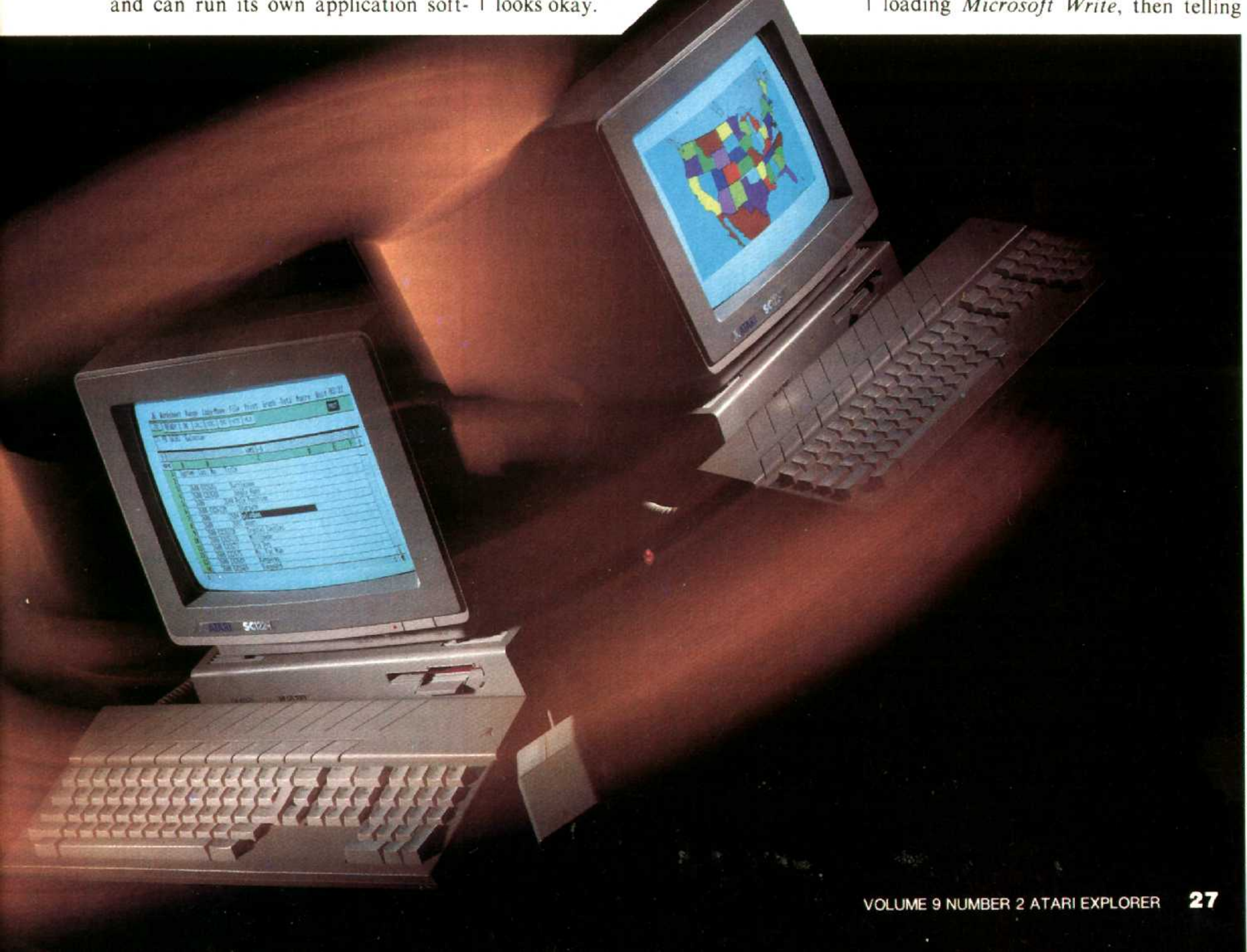
No, *Revolver* is a different kind of program entirely. What *Revolver* does is set up virtual machines inside your ST. Each virtual machine is, in effect, a fully-functional, independent ST computer. It boots independently, can be reset independently, has its own hardware and software *state* (i.e., video resolution, color settings, serial port parameters, timer chip settings, mfp interrupt vectors, system variables, and what-have-you), can be configured with its own group of AUTO programs and desk accessories, has full access to all disk drives and peripherals (while active), and can run its own application software. Oh, and of course, you can jump from one machine to another via hotkey—maybe *that's* why they call it a switcher.

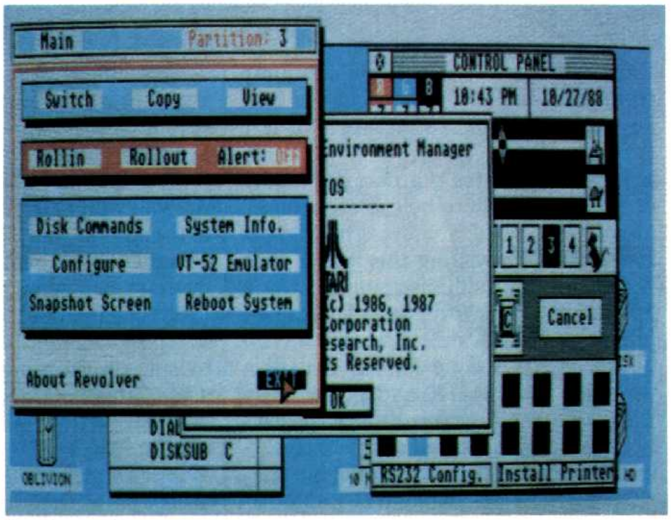
If your jaw isn't already hanging down to somewhere below your knees, maybe you need a more graphic description. I am writing this article with *Word Plus* in medium-resolution on a 437K, plain vanilla virtual ST. Now, by pressing Control-Shift-Alternate, I can switch to my second virtual machine, which also has 437K and is currently showing me a medium-res desktop.

Employing the usual magic passes, I use Set Preferences to throw the machine into low-res and load *Degas* in 16-color mode. Everything works normally and looks fine. I begin drawing a diagram . . . oops! I have to check the caption. Control-Shift-Alternate (followed by a peal of chimes) and I'm back in *Word Plus*, and my cursor is just where I left it. Caption looks okay.

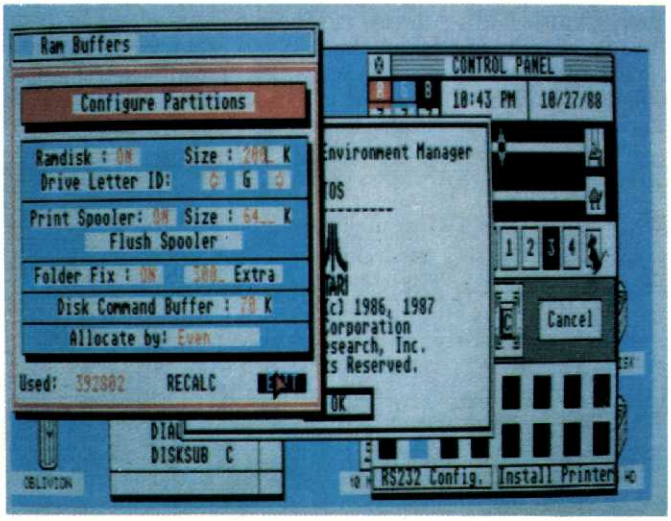
Bouncing back to the *Degas* machine, I add finishing touches to the picture, save it on disk, and exit to the desktop. Hmmm . . . I have never really liked the low-res desktop. It looks clunky. So, just to prove a point, I reach around the back of the console and press the reset button. The system re-boots, and I'm back in medium res. Now to finish the article . . . bounce!

It is like having two STs running side-by-side on the same desk, except that one of them is asleep when you are not working with it. And that's not all the program can do—not by a long shot. For example, from right here in *Word Plus* (on virtual machine #1), I can bring up the *Revolver* control panel and search my hard disk for a file called MWRITE.MIM. That file is what we call a *snapshot* in *Revolver*-speak—a heavily-compressed memory-image of a specially-configured virtual machine, created by booting VM #2 with GDOS, loading *Microsoft Write*, then telling


Photograph by Jeff MacWright

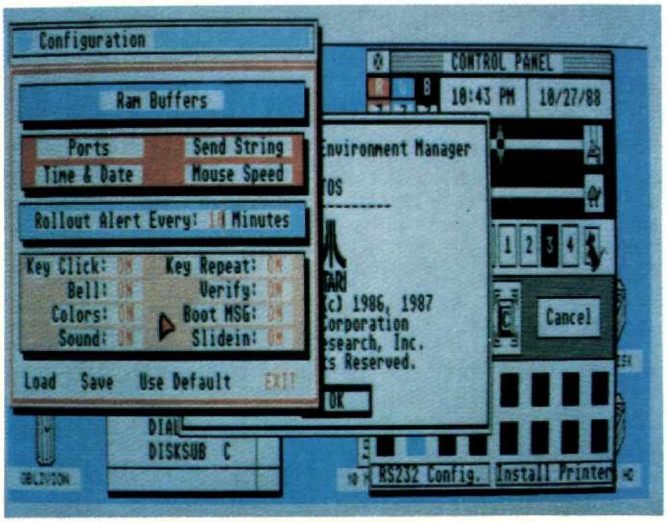


Revolver main menu.



Randisk : ON Size : 200 K
Drive Letter ID: G
Print Spooler: ON Size : 64 K
Folder Fix : ON 300 Extra
Disk Command Buffer : 70 K
Allocate by: Even
Used: 392802

Menu used to configure Revolver partitions.



Rollout Alert Every: 10 Minutes
Key Click: ON Key Repeat: ON
Bell: ON Verify: ON
Colors: ON Boot MSG: ON
Sound: ON Slidein: ON

Menu used to configure RAM buffers.

Revolver to roll the whole mess out to disk.

Now, by telling *Revolver* to roll in that file (a process that takes about ten seconds), I can turn VM # 2 into a fully-GDOS-equipped, laser-breathing *Microsoft Write* monster, rarin' to accept keystrokes.

And when you consider that I can do it with a keystroke and two mouse-clicks from inside another application, running on what is, in effect, another (in this case non-GDOS-equipped) ST, in about one-fourth the time that it would take to actually reboot the ST (or, for that matter, VM # 2) under GDOS and load *Write* in the conventional manner and all without even having GDOS.SYS in my AUTO folder . . . well, I must say that I really like this program.

How it Works

Revolver is the brainchild of Richard Plom, himself a sort of brainchild (he is about 21 years old) whose prior programming credits include *Music Construction Set* for the ST (distributed by Electronic Arts). Richard began designing *Revolver* around a very simple inspiration. He figured that if he could find a way to save an exact image of the memory of the ST, its hardware state, and its processor registers, all at a given instant, he could later restore the machine, the software, the whole kit-and-kaboodle to exactly the way it was before it was interrupted.

Of course, doing this turned out to be a great deal more involved than generating an interrupt and spooling registers and memory out to disk. Reading memory is easy, but determining the hardware state of the ST at a given instant is something else again. Though most of the hardware is indeed controlled by writing into registers mapped into dummy memory locations, reading those locations will not, in many cases, give you a clue as to how the hardware is currently set up.

When normal programs fiddle with the ST hardware, they generally get around the problem by brute force, initializing the hardware at a *known state* and keeping a record of that state in a variable. But a program like *Revolver* can't get access to these private variables. Instead, it is obliged to deduce the state of certain chips by examining the behavior of the system and then applying algorithms.

This is a very difficult task, but Plom succeeded remarkably well. As a result, *Revolver* is capable, for example, of switching transparently between virtual machines running several different brands of MIDI sequencer, each of which (one assumes) handles the sound chip, the system timers, and the MIDI-keyboard interrupt in its own way.

According to Plom, the only aspect of the ST hardware that *Revolver* v.1.0 does not save and restore is that part maintained in the intelligent keyboard subsystem—an (electronically-speaking) "offboard" unit with its own RAM, I/O chips, and microprocessor. This accounts for the occasional failure of the program to restore accurately the state of software that reprograms the IKBD processor—particularly games that alter the IKBD joystick interface.

The original prototype of *Revolver*, finished in about three weeks, was a proof-of-concept utility that saved a snapshot of the ST to disk, then spooled it back in again. But Plom didn't stop there. The next step was to arrange to save these snaps in protected memory, letting the user switch between them in instants, rather than seconds. Then came the idea of cloning GEM in each partition. Finally, other ease-of-use features were grafted in.

Installation and Use

To function correctly, *Revolver* must be the first program loaded from the AUTO folder. An installation program sets this up for you by copying out the contents of the AUTO folder on your boot disk, copying in *Revolver*, then copying back your other AUTO programs.

Thereafter, a warm or cold restart boots your main virtual machine, at which point, *Revolver* installs itself, establishes protected memory areas and partitions, then reboots the virtual machine. If you watch the screen, it looks as if the initial portion of the boot cycle proceeds normally, then briefly hangs before repeating to completion. In the process, *Revolver* announces itself with chimes and displays a brief message showing how much memory has been allocated to VM #1.

The typical user will swiftly find himself staring at the GEM desktop, configured in whatever manner is specified in DESKTOP.INF. However, it should be noted that since *Revolver* installs itself before GEM is initialized, you are not necessarily obliged to end up at the traditional desktop on this or any other virtual machine. Presumably, other auto-booting replacement desktops will work, though I haven't tested any.

I have, however, tested several TOS command shells, including MichTron's DOS Shell and Mark Williams' MSH, both of which can execute as AUTO folder programs. Both work fine under *Revolver* with one small hitch.

Because GEM is not yet initialized when such shells are auto-executed, you can't give commands to *Revolver* while the shells are running—the hotkeys don't work, and the *Revolver* dialogs can't operate. However, when you exit the shell and the boot cycle completes, GEM initializes and *Revolver* commences to operate normally.

Once the initial boot cycle is finished, you can begin using the virtual machine in normal fashion, executing applications up to the limits of the RAM available in your VM. *Revolver* defaults to two, equal-sized partitions, meaning that an unenhanced 520 ST will be split into two 256K machines, a 1040 into two .5Mb machines, and so on.

Many common applications will face memory limitations in a 256K virtual machine (more on reconfiguration later), but most will tolerate and function well in a .5Mb or larger environment. Even in a worst-case situation, most GEM applications will tell you when you bump up against these limits, then simply abort, returning you to the desktop. Other software may lock up, but in the vast majority of cases, a simple warm start will clear and reboot the virtual machine, leaving other resident VMs (if any) quite safe and sound.

You can then proceed to boot (or roll in from a snapshot file) any or all of your remaining virtual machines. The former is accomplished by placing whatever boot disk you wish to use in your main drive, then simply switching to the desired VM (either by pressing Control-Shift-Alternate or via the *Revolver* control panel). If inactive, the VM boots automatically at this point, coming up in whatever configuration the boot disk dictates.

Unfortunately, *Revolver* contains no dodge to support multiple boot images on the same disk, so hard disk users who normally boot from drive C will typically find all their virtual machines booting in the same format, at least initially. The "roll out-roll in" capability of *Revolver* handles this problem quite nicely, however. It is very easy to configure a VM the way you want it, then save a snapshot of it. Rolling such a snapshot into a VM partition large enough to contain it configures the VM in one fell swoop, usually much faster than the same configuration could be accomplished by booting in the conventional manner.

Once your available VMs are booted or rolled in, you can switch from one to another in an instant. If you merely need to see the screen of a given VM (as I do when I compare documents and diagrams), you can call up the *Revolver* control panel and ask to see the screen of the VM in question. The control panel disappears, and the screen you want to see is instantly superimposed on that of your current application.

The advantage here is that you can use the mouse to slide the superimposed image up and down over your current screen like a window blind—making comparison easy. Difficulties arise, however, if you try this trick between VMs operating at the same screen resolution but with different color palettes, or between VMs operating at different resolutions. The superimposed image takes on the attributes of the current screen, which can make for some pretty weird images. In such cases, it is better simply to switch back and forth between the two machines.

Another, related problem sometimes arises with the control panel, which normally employs the color palette of the application from within which it is invoked. If this color palette provides insufficient contrast, it may be difficult to make out control panel elements or text. However, if you don't mind what may be a temporarily displeasing visual effect on your current application, *Revolver* can be set to impose its own pal-

Revolver sets up virtual machines, ***each of which is, in effect, a fully-functional, independent ST computer.***

ette on invocation, then reinitialize the palette of the application on exit.

As noted above, the default configuration of *Revolver* establishes two, equal-sized virtual machines. This configuration can be altered from the control panel and saved to disk as a .DAT file in the AUTO folder. The system must be cold-booted to establish a new configuration; thus there is no practical way to save multiple configurations and invoke them at runtime.

Revolver allocates memory in blocks of 256K minimum (v. 1.1 allocates memory in 128K blocks), so a 520ST owner cannot improve on the default configuration. In fact, because each VM must contain at least 130K for GEM (if GEM is allowed to boot), a partition of less than 160K would be unusable anyway. Practically speaking, even 256K partitions are fairly tight, though usable. Almost anything can run at least marginally well in a partition of .5Mb or more (except special applications such as desktop publishing). If you have enough memory, *Revolver* will permit you to configure up to eight partitions, distributing memory among them in one-block units. A thoughtful addition: the *Revolver* configuration dialog contains a button that will distribute memory equally among any number of VMs.

Power users will find ample challenge configuring *Revolver* for all sorts of special applications. The only constraint the program itself imposes (aside from the one-block minimum unit for memory allocation) is the fact that VM snapshots can only be rolled into a VM of size equal to or greater than that of the VM from which they were rolled out. A system thus achieves maximum flexibility when all its VMs are of equal size. (Additional variables must also be taken into account in configuration; namely, the memory to be allocated to the *Revolver* rebootable RAM disk, print spooler, and disk transfer buffer. More on these, below.)

Roll Ins, Roll Outs

As I have mentioned repeatedly, one of the most impressive features of *Revolver* is the ability to roll out a compressed image of a virtual machine to disk, then roll that image back into any virtual machine of equal or greater size. Aside from this absolute size consideration, the only constraint is that you must not be working with the virtual machine you wish to target for a roll out/roll in operation. This is undoubtedly because the image *Revolver* saves and restores is a copy of the machine image it saves in its switcher memory when you switch from one VM to another in RAM.

Revolver performs compression as it rolls a machine image out to disk. One supposes that the degree of compression achieved is partially controlled by the RAM contents of that machine. By experiment, I have found that my 437K partitions roll out to files of about 240K—an almost 50% size saving. Naturally, if you use a hard disk, you can save a large number of VM snapshots without difficulty. Single-sided floppy users, however, may find themselves coming up against the disk-size wall with alarming frequency.

Rolling a .5Mb virtual machine out to hard disk can take as little as 20 seconds; saving to floppy disk, between 1 and 2 minutes. The roll in operation is quicker, because verification is not required: about 10 or 15 seconds from hard disk; a minute from floppy. This seems like a long time, but when you consider the enormous amount of work that is being accomplished—saving and restoring the entire machine state and configuration, all DAs (desk accessories) and other background applications, a primary application, and all of its data—it seems like a bargain.

By comparison, think how long it would take to boot a hard disk system from scratch with GDOS, fonts, a few DAs, and various "helper programs," then load *Microsoft Write*, find a document, load it in, find your place, and get ready to write. It takes much less time to roll in an image of the same machine, saved at the end of your last *Write* session.

Users of *Revolver* can exploit roll in/roll out in a wide variety of ways. I, myself, have begun developing several different classes of memory image file, organized by directory. One group contains different basic desktop configurations—low res, medium res, GDOS, no GDOS, etc. I can load one of these into a virtual machine, then go on to load applications appropriate to the environment.

Another group contains my various "work suites," each consisting of multiple virtual machine images. The Programming environment, for example, contains an image of my standard ST running under the Mark Williams' C-programming system; another that portrays the same ST running under AssemPro; and a third, plain vanilla desktop image that I use as a test bed for programs under development. Since all my VMs are the same size, I can mix and match at will. Users who save images of different-sized virtual machines should adopt some system of organization at the outset.

Other Features

Revolver incorporates a wide variety of additional features that serve to maximize its utility while minimizing problems that may arise in its use. The need for these special features became obvious to me the first time I booted the program. Normally, my system comes up with a variety of utilities and DAs installed, among them *Universal Item Selector*, an enhanced file selector that replaces the GEM standard FSEL and provides a range of disk commands from within its own DA; a "40-folder fix" utility; a 300K RAMdisk; a 64K software print spooler; EMULATOR.ACC; and the Control Panel. On a normal 1040, this leaves me with plenty of RAM for applications. On a 437K virtual machine, however, the RAMdisk refuses to boot outright, and all those DAs and fixes leave me crying for memory.

If I had been using my Mega 4 at the time, I probably would have configured larger partitions for my VMs and left it at that. But then I thought about it: if I were to boot every VM with the same combination of RAM-resident stuff, I would end up with an enormous, redundant overhead on the system. Still, how else to have access to these facilities in most or all of my virtual machines?

Luckily, Plom anticipated this problem and built 95% of the stuff average users need into the RAM-resident portion of *Revolver*. A full set of disk commands is available, including directory, copy, move, rename, append, delete, lock/unlock, hide, search disk (with pattern-matching), show file (much nicer than the desktop function—you can scroll up and down through your file with the mouse keys), make/kill folder, and format (standard or extended, no less).

Also available are a reconfigurable 40-folder fix, a VT-52 emulator, a partial control panel, a software cold start function that totally clears and reinitializes memory in a partition prior to reboot, a screen snapshot utility that saves a NEO-compatible screen image to disk, a utility that lets you send character strings to your printer, a time-and-date setter, a "roll out alert" function that tells you when it's time to roll out a partition, and even a "process dump" utility that shows the current VM memory configuration, 68000 program-counter location, and the contents of other processor registers. (I can't think of any real use for these last two features, but heck, he had space for the buttons in the dialog, and the code probably fell out quite naturally from the overall system design.) In addition, *Revolver* supports a command buffer system that speeds up disk access and a reconfigurable RAMdisk and print spooler that are equally accessible to all virtual machines.

Memory for the RAMdisk, print spooler, and disk command buffer must be allocated when *Revolver* is configured. (Users who don't require the RAMdisk or spooler can elect to turn these facilities off.) A proportion of the common memory used by these constructs is subtracted automatically from each virtual machine partition.

Like the VM structure of *Revolver*, the RAMdisk, spooler, and command buffer are reset-proof, installing on cold start when *Revolver* itself is initialized. Virtual machines that need to use the RAMdisk must have appropriate disk icons installed.

Any program that employs BIOS or higher-level print functions can automatically exploit the spooler, and multiple print jobs from multiple applications and/or VMs can be queued, up to the limit of the spooler memory. This last is a very nice touch; you can start a print job from within a program on one VM, wait a second for the data to pass into the spooler, then switch to another VM or application and spool out another print job. The jobs will print in sequence, without problems, providing that the software you are using correctly resets the printer for each print run. Chances are that the spooler will not work with the Atari Laser Printer, however, because the software that drives it is not based on the normal printer functions of the BIOS.

Another unusual added feature is a function that lets you control the way the mouse cursor moves in relation to the mouse itself. You can choose a normal, linear unit-of-screen-movement/unit-of-mouse-movement relationship at normal (1:1) or high speed (8:1) ratio. Or you can opt for proportional control, where the speed of the cursor is determined by the acceleration of the mouse. This offers a very interesting compromise between mouse speed and accuracy, which is great for working with certain kinds of graphics programs. Regardless of the mouse configuration, holding down the Alternate key will return the mouse, temporarily, to the normal linear, 1:1 relation.

You can start a print job ***from within a program on one VM, wait a second for the data to pass into the spooler, then switch to another VM or application and spool out another print job.***

The overall quality of the *Revolver* utility suite seems very high, most functions performing better than the desk accessories and helpers they were designed to supplant. While users will still want certain general purpose DAs on their virtual machines (I, for example, could never live without *Universal Item Selector*, even though most of its functions are duplicated in *Revolver*), most common ancillary software can be safely eliminated, freeing up memory for optimal VM configurations.

Graphics, Docs, and Other Details

When the *Revolver* main control panel is invoked, it appears in a fixed location in the upper-left-hand corner of the screen and cannot be moved. The panel and its subsidiary dialogs are beautifully designed—colorful, attractive, and clearly presented (see illustrations).

While interacting with *Revolver*, the mouse cursor is changed to a shaded, tail-less arrow, and its movement is limited to the borders of the dialog. Selectable items are highlighted as the mouse passes over them, so you are always sure of what will happen when you click. *Revolver* informs you politely of its progress in every operation and asks that you confirm most operations that could have catastrophic effects.

Most disk operations are performed through the *Revolver* file selector, a nice piece of work that lets you change drives by single-click and contains other amenities. The only annoying feature of the selector is that it does not display file size or creation date, both of which would be useful, particularly in determining whether a given memory image file is appropriate for rolling into a given partition.

For such a powerful program, *Revolver* comes with surprisingly little documentation—just a nicely-printed, reasonably well-written 30-page booklet that tells you all you need to know. Each feature of the program is adequately described, and the manual is acceptably well-organized, containing a table of contents but no index. A README file on the distribution disk contains last-minute information about application compatibility.

Limitations

Most productivity software (with the significant exception of *Word Perfect*, which seems not to like the way *Revolver* alters the mouse handlers) appears to work trouble-free in a *Revolver* partition. I have tested a variety of popular word processors, desktop publishing programs, graphics packages, spreadsheets, databases, programming environments, and telecommunications programs, and have not run into any problems worth mentioning.

In fact, I have found the capacity of *Revolver* to save and restore the hardware state of the machine between switches to function uncannily well in most situations. For example, I have tried running two different sequencers on differently-configured virtual machines, switching back and forth between them while both were in the process of sending a musical sequence to my MIDI equipment. The result was just as if I had been switching MIDI cables back and forth between two separate STs.

On the down side, most games and pure entertainment packages don't work so well. In general, I get the feeling that the more ambitious (or carelessly-written) a program is, the more trouble it will have in accommodating *Revolver*. Games, for example, often employ high-performance animation methods that require buffers fixed at particular places in memory, which may conflict with the repartitioning of the system.

Programs that will boot only from a cold started distribution disk (copy-protected games are a typical example) will certainly not work. Neither will programs that analyze physical hardware in order to take over the whole machine or that manipulate the keyboard processor (often the case with games that employ joysticks).

Revolver is also incapable of handling unusual application environments such as Magic Sac and *PC Ditto*, which, like certain game programs, must themselves control the entire memory map of the machine in order to do their stuff. One can hope that future revisions of *Revolver* will permit you to configure in low memory a 512K or 1Mb main partition that could house an emulator. Imagine having a Macintosh in one partition and an ST in another!

Programs that call system functions to access the IKBD time-of-day clock (as opposed to the internal system clock) will not be properly restored after a switch, because *Revolver* cannot currently save factors of the IKBD hardware. In such cases, the program's subjective idea of what time it is when restored will differ from the IKBD time, which keeps ticking away even when the application is temporarily asleep (the Rip Van Winkle effect). An AUTO-folder patch program, which redirects the time functions of the system to return only the internal time is included with *Revolver*. This may solve certain compatibility problems.

Most disk operations are performed through the file selector, a nice piece of work that lets you change drives by single-click and contains other amenities.

Certain other limitations are inherent to working in a multiple virtual-machine environment. While *Revolver* is not a true multitasking system and thus does not introduce the problem of blocking simultaneous I/O to the same resources by different processes, it is still potentially vulnerable on the I/O front.

In particular, it is wise to avoid attempting to switch VMs when an application is writing to or reading from disk. *Revolver* has built-in safeguards against allowing a task-switch while disk I/O is in progress, and these work quite well (they have saved me on a few occasions). Still, there is always the possibility that a program will bypass these defenses by some strange hardware manipulation.

While the lack of support for multitasking doesn't provide background downloading capability, I have experimented with doing VM switches between a machine running *Flash* connected to CompuServe and a machine running an unrelated application, and they work fine. The best general rule to adopt, then, is simply to stick within a given virtual machine until things are quiet—not inactive, just quiet—on the I/O front.

TOS programs present a unique problem for *Revolver*, since executing them renders GEM temporarily quiescent. Because the *Revolver* user interface depends on GEM, you cannot switch machines or access other *Revolver* functions from within a TOS environment (though, once you exit the TOS program and return to the desktop or GEM application, *Revolver* is ready and waiting). This presents a particular problem for people accustomed to doing much work from within a Unix or DOS-compatible shell.

In such cases, the fix is to execute a GEM application from within the shell, then do the switch after the application comes up. The program in Listing 1 is a minimal GEM application that I wrote to open up the Mark Williams MSH shell to permit a switch. It should work equally well under other popular shells, though it may look messy.

```
#include <aesbind.h>
#include <osbind.h>

main()
{
appl_init();              /* GEM application init. */
printf("Revolver hotkeys will now work.\n");
printf("Press any key to return to shell prompt.\n");
while (!Bconstat(2));     /* Wait for keypress, but don't eat it. */
Bconin(2);                /* Now eat it. */
appl_exit();              /* GEM application exit. */
}
```

Listing 1. C program to permit a Revolver switch from within a TOS shell.

Conclusion

In looking back at this review, I realize that it must read less like a ruthless evaluation and more like a long love letter to a product. Yet I find it very difficult to criticize *Revolver*; the program simply does its job too well. A lot of software passes across my desk in the course of a year, and having been burned by some of it, I have developed strongly paranoid instincts about installing new products on my production system—particularly programs that do such radical things. Yet there is something about *Revolver* that inspires confidence, even when important work is at stake.

After using *Revolver* for a while, for example, you stop reflexively saving documents before performing a switch, confident that they will be there when you switch back. I wrote this review over a period of three days, during which time I used *Revolver* in a hundred different ways—and I never lost a word. My suggestion: buy it. It will completely change the way you use your ST. ■

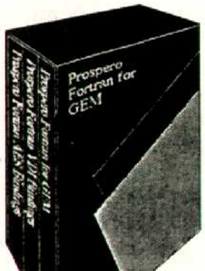
Version 1.1

As this review was going to press, we received a copy of *Revolver* v. 1.1. The most significant revisions include a reduction of the minimum block (hence virtual machine) size to 128K, enhancement of the mouse driver to afford compatibility with *Word Perfect* and other finicky applications, and a copy partition function that lets you start an unbooted virtual machine by copying an active virtual machine to it, speeding the startup process.

The *Revolver* control panel has also been enhanced, and a boot manager utility that lets you control which AUTO programs and accessories are booted into each virtual machine has been incorporated.

Come to grips with GEM!

Pascal $149
Fortran $199



800 pages of documentation included

Prospero Pascal for GEM and **Prospero Fortran for GEM** - two new products for the Atari ST - with:

- Complete programming environment with editor and workbench
- High performance compiler (Pro Pascal or Pro Fortran)
- Linker, Run-time Libraries, Librarian, X-referencer, Symbolic Debugger
- Compiled Pascal or Fortran GEM bindings
- Complete language and GEM documentation
- Access to BIOS, XBIOS and Line A routines

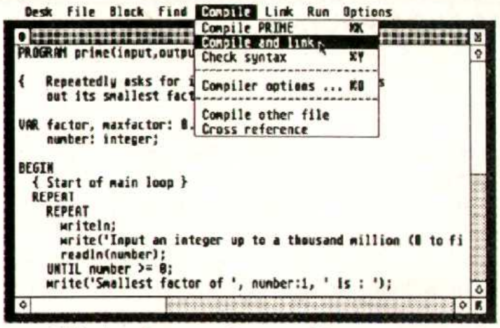
The **programming environment** is designed to stay resident in your Atari while you are programming. It controls the editor, the compiler, the linker and utility programs, and allows you to run the program you have compiled or any other program.

With the **four-window editor** you can load up to four different source files, and cut and copy between them - the editor understands Wordstar® command sequences. It has block copy and move as well as powerful search and replace functions.

The **compiler** is Prospero's well established Pro Pascal or Pro Fortran-77 compiler, both of which conform fully to ISO and ANSI standards.

The **linker** is fast and efficient; assembler language libraries may be introduced.

The **debugger** provides complete source line tracing and source variable display capability; break points can be set; the calling sequence may be shown, the last ten lines executed can be listed, as can any source lines from the main program or any libraries; you can execute SID or any other program; screen switching separates program text and GEM output.



```
PROGRAM prime(input,outpu

{   Repeatedly asks for i                        s
    out its smallest fact

VAR factor, maxfactor: 0.
    number: integer;

BEGIN
  { Start of main loop }
  REPEAT
    REPEAT
      writeln;
      write('Input an integer up to a thousand million (0 to fi
      readln(number);
    UNTIL number >= 0;
    write('Smallest factor of ', number:1, ' is : ');
```

Windowing and graphics support is provided by GEM; the documentation gives all the explanation needed to use these powerful functions.

The three volume **Documentation** pack includes:

- Installation and operating instructions
- Implementation details
- The programming language specification
- Detailed descriptions of all 109 VDI Bindings and all 101 AES Bindings, with example programs.

TO BUY NOW CALL 1-800-327-6730.
Visa, Mastercard & CODs taken.

Prospero Software, Inc. 100 Commercial Street Suite 306, Portland, ME 04101. Tel (207) 874 0382. Distributor and dealer inquiries invited.

For international sales call London 01-741 8531 or write to Prospero Software Ltd, 190 Castelnau, London SW13 9DH, England.

Prospero Software
LANGUAGES FOR MICROCOMPUTER PROFESSIONALS

IB Computers

1519 SW Marlow Ave.
Portland, OR 97225
(503) 297-8425
9AM-6PM Mon.-Sat.

IB Copy

- Copies most protected ST disks
- Copies selected protected ST disks to work from a hard disk
- Copies most IBM software from 5 1/4" to 3 1/2" drives
- Not copy-protected
- Works with 1 and 2 drive systems, single and/or double sided drives

$34.95

IB Disk Utility

- Edits files or sectors on floppy disk
- Edits files or sectors on hard disk
- Multiple edit buffers
- Mouse or keyboard command entry
- Recovers deleted files
- Not copy-protected

$34.95

New!!

IB HD Backup

- Backup to floppy or hard disk partition
- Supports single or double-sided formats
- Works on color or monochrome
- 520, 1040 and Mega compatible*
- Not copy-protected

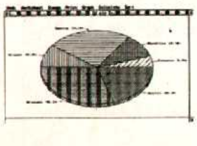
$34.95

** Fast ramdisk backup option available on systems with 1 megabyte or more of memory.*

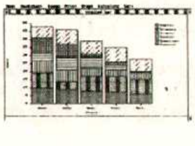
ANALYZE!

The spreadsheet for your ST! Analyze! is a powerful spreadsheet program with extensive ma functions, a full macro language, and outstanding graphing capabilities.

Analyze! is .WKS-file (VIP/1-2-3) compatible and makes full use of the GEM environment.



$34.95



So... You say You want a REVOLUTION

Our Peerless Switcher

Have multiple programs in memory at once, each Reset Proof in its own partition

Mixed Partition Sizes, as few as 2, as many as 8

Full disk utilities with wildcards, drive search, etc

Save a partition to disk for future continuation, even after a Cold Boot

Built in Ram Disk, Print Spooler and 40 Folder Fix

Snapshot screen to .NEO file

VT52 Emulator and most Control Panel Options are built in

Easy to use and very reliable

THE ULTIMATE UTILITY

$49.95

REVOLVER

Minimum One Meg RAM Recommended

Our Superior Terminal

GEM Background Downloading for XMODEM, YMODEM and Batch YMODEM

VT100, VT52, IBM3101,PC-Graphics and ATASCII Graphics

Call Logging and Cost Calculator

Mini-BBS (Answer Mode)

Disk Commands including high density format

Unparalleled Type Ahead Buffer (Chat Mode)

Loadable protocols for terminal emulation and file transfers make INTERLINK the one to grow with

FIRST WITH THE MOST

$39.95

INTERLINK

Modem Required

Check with your dealer or call us: INTERSECT Software, 2828 Clark Rd. Suite 10 , Sarasota, FL 34231

1-800-826-0130 or 1-813-923-8774

ST Control

If you don't mind a little programming, this universal control program can help you turbocharge your ST

Like so many other victims of technology (always looking for faster and more powerful ways to slough off the chains of honest labor), I am a great fan of macros. A macro, for those of you who are unfamiliar with the terminology of "power computing," is a recording of the manual steps required to make a program do something useful. Such a recording can be played back under software control, making the program respond as if a user were in the driver's seat.

For example, suppose you wanted to write a business letter with your favorite word processor. You might begin by typing Control-N (for New Document), then hit Return four times to get down to the fifth line, Tab five times to get over to the right-hand side of the page, enter the first line of your address, press Return, and tab five more times before entering the second line of your address, then repeat the process for the third line. Next, you would hit Return twice and Tab five more times before typing Control-D to insert the date, then conclude by hitting Return three times, leaving your cursor in position to enter the first line of the address of the person to whom you are writing.

As with everything you do with an application program, the whole process can be expressed in terms of keystrokes, as follows: <Control-N> <Return> <Return> <Return> <Return> <Tab> <Tab> <Tab> <Tab> <Tab> Your Name <Return> <Tab> <Tab> <Tab> <Tab> <Tab> Your Street Address <Return> <Tab> <Tab> <Tab> <Tab> <Tab> Your City, State, Zip <Return> <Return> <Tab> <Tab> <Tab> <Tab> <Tab> Date <Return> <Return> <Return>.

A word processing program with a good macro facility would let you enter this procedure in symbolic form as shown above or would record your keystrokes as you performed the procedure manually. Thereafter, you could cause the macro to be played back at the touch of a single function key, saving dozens of keystrokes every time you wanted to write a letter.

Technical snobs will be quick to say that this isn't *real* programming, because the actual interface to target data—i.e., your business letter—is an application program, rather than a set of basic machine operations. And it's true—no matter how well a macro system is designed, it won't let you teach a computer to do something that an application doesn't exist to cover.

ST Control

System: Atari ST

Version reviewed: 1.5

Copy protection: None

Summary: A versatile, comprehensive macro processing program that can help you turbocharge your ST.

Price: $69.95

Manufacturer:
Trio Engineering
P.O. Box 332
Swampscott, MA 01907
(617) 964-1673

In fact, macros—in the traditional sense—pose some severe problems and impose some severe limitations. First of all, most macro systems are built into individual application programs, which means that you have to learn a new macro system for each application you use, and that none of these individual application macro systems can help you when you are outside an application—i.e., on the GEM desktop.

Second, keyboard-oriented macro systems ignore the most important part of the ST user-interface, the mouse. Third, traditional macros are just collections of keystrokes; they have no internal logic and can't adapt to circumstances—*you* have to decide when it is appropriate to fire off one macro or another.

The ideal solution would be to devise a macro system that incorporated a fully-interactive programming language—one that could act as an intelligent interface between the user, with his mouse and keyboard, and the entire ST system—GEM and applications.

Enter *ST Control* from Trio Engineering, a desk accessory that provides full macro recording, programming, compilation, and execution facilities from outside both GEM and applications.

Programming? Well, yes. Though the introduction to the *ST Control* manual does waffle a bit on the old P-word, the largest portion of that manual—like the largest portion of the program's capabilities—is devoted to the writing, compilation, and debugging of . . . well, there's no other word for them—programs.

To be sure, these programs can be generated (wholly or in part) by recording sequences of manual operations (like macros), but the real strength of *ST Control* lies in its ability to tie such recorded sequences together into interactive meta-applications that go far beyond traditional macros in enhancing productivity.

Consider the problem of someone who sends a lot of electronic mail. It would be convenient for such a person to be able to send an electronic letter directly from his word processor. Unfortunately, the perfect combination of high-quality text editor and telecommunications package has yet to be devised. Nevertheless, using a properly designed *ST Control* program, it would be possible to interface say, *1st Word Plus* and *Flash* into an electronic mail composition and delivery system.

You could compose your letter in

By JOHN JAINSCHIGG

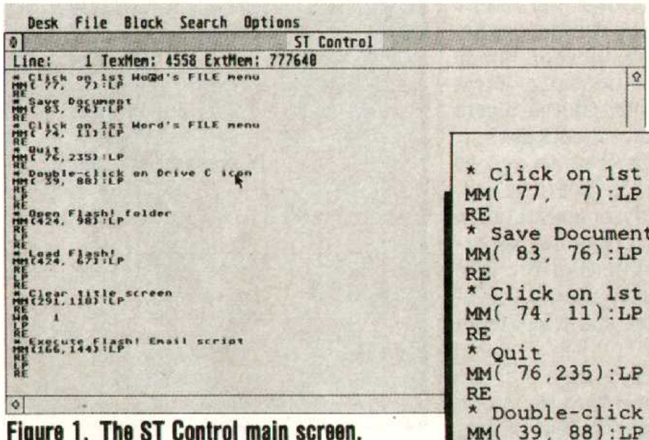
Word Plus in a format acceptable to the target network, then activate an *ST Control* script that would save the file, exit *Word Plus*, load *Flash*, execute one of the *Flash* autodial macros (which would handle the job of connecting with the system, navigating to the Email area, uploading, and going offline), then wait for a continuation command before exiting *Flash*, and return to *Word Plus*—all in less time than you could possibly do it by hand.

Sound like fantasy? I have such a script running on my system right now (see Figure 1). It took only about ten minutes to set it up, and—to be fair—most of that time was spent in figuring out how to automate access to CompuServe's Email network via *Flash*, not in pulling together the application itself. That's real productivity power!

ST Control is a desk accessory *that provides full macro recording, programming, compilation, and execution facilities from outside both GEM and applications.*

The Desk Accessory

ST Control is designed around a desk accessory that occupies only about 27K on disk, though in-memory allocation will vary according to how much macro text and code are currently stored in the *ST Control* buffers.



```
Line:    1 TexMem: 4558 ExtMem: 777648
* Click on 1st Word's FILE menu
MM( 77,  7):LP
RE
* Save Document
MM( 83, 76):LP
RE
* Click on 1st Word's FILE menu
MM( 74, 11):LP
RE
* Quit
MM( 76,235):LP
RE
* Double-click on Drive C icon
MM( 39, 88):LP
RE
LP
RE
* Open Flash! folder
MM(424, 98):LP
RE
LP
RE
* Load Flash!
MM(424, 67):LP
RE
LP
RE
* Clear title screen
MM(291,118):LP
RE
WA    1
LP
RE
* Execute Flash! Email script
MM(166,144):LP
RE
LP
RE
```

Figure 1. The ST Control main screen, showing a recorded script being edited.

Installation of the program is very straightforward. You just copy STCONT.ACC to the root directory of your boot disk, reboot, and you are ready to go. Clicking on the *ST Control* desk accessory causes a new menu bar to appear and a full-screen, non-resizable editing window to open up. The effect is much like being in an interactive programming environment.

In addition to the standard Desk menu (from which other desk accessories are accessible), the *ST Control* menu bar sports File, Block, Search, and Options headers.

As expected, the File menu supports the loading and saving of text and compiled code, the allocation of memory for new documents, and the clearing of buffers. The Block menu controls the saving, insertion from disk, and printing of marked text blocks.

The Search menu implements basic search and replace functions. Lastly, the Options menu controls the compilation (with or without Trace) and execution of macro programs. In addition, the Options menu supports the recording of keypress and mouse-movement sequences and their translation into macro source. It is this last capability that makes *ST Control* at least marginally accessible to users who have no interest in programming.

The Editor

The *ST Control* editor is a general purpose ASCII text editor that can be used to edit any kind of text file. The usual range of block copy, move, delete, search and replace, and other standard functions is implemented. The mouse can be used to select whole lines and groups of lines for block operations, although, like *Word Plus*, it cannot grab text with arbitrary start and end points.

The editor is problematic in the way it handles memory. When a new document is initialized, it is given 5K of space—enough room for a substantial amount of typing in the current session. This 5K allocation, however, is considered a hard limit; when you reach the end, the program tells you to "Please save, reload, and try again." (The save/reload procedure is required, because *ST Control* always allocates 5K extra to any file loaded from disk.)

To be sure, the manual does mention that dynamic memory allocation is a particular problem for desk accessories, because the system does not free their allocations when resolutions are switched, and there are other problems with the ST memory-allocation functions that limit the number of small chunks of RAM that a program can grab onto, moment by moment. Still, this kind of memory management seems primitive.

In addition, although *ST Control* normally saves your current text in RAM between invocations, the manual mentions that text may be lost (and a crash may result) if a document is initialized when *ST Control* has been invoked from within a running program, then exited, and the main application is subsequently terminated.

The editor itself runs in a non-resiza-

```
* Click on 1st Word's FILE menu
MM( 77,  7):LP
RE
* Save Document
MM( 83, 76):LP
RE
* Click on 1st Word's FILE menu
MM( 74, 11):LP
RE
* Quit
MM( 76,235):LP
RE
* Double-click on Drive C icon
MM( 39, 88):LP
RE
LP
RE
* Open Flash! folder
MM(424, 98):LP
RE
LP
RE
* Load Flash!
MM(424, 67):LP
RE
LP
RE
* Clear title screen
MM(291,118):LP
RE
WA    1
LP
RE
* Execute Flash! Email script
MM(166,144):LP
RE
LP
RE
```

Meta-application for uploading Email automatically—a typical ST Control script.

ble GEM window with a status line that bears a record of the amount of buffer and system memory remaining (just a little note to remind you that not all is cozy in memory-allocationsville) and the line you are currently on. The mouse cursor flickers on and off as you type, and as you move up and down in text, the entire status line is re-drawn to update the Current Line field. Still, it is no less usable than many "hi kwality" GEM-based editors.

On the plus side, horizontal sliders are employed so that you can enter up to 80 characters per line in any resolution, and search, replace, file transfers, and block operations are exceptionally fast. These factors add to the utility of the *ST Control* editor as a general purpose tool, as well as a mechanism specifically designed for editing *ST Control* programs.

The Record Facility

As noted above, *ST Control*—like most macro systems—is capable of recording a sequence of mouse and keyboard actions. It can then translate these actions into *ST Control* language, which can be edited, added to other *ST Control* programs, or compiled directly into executable form and played back. The record facility thus provides both a means for the non-technical user to get something out of the program, and—for the programmer—a means of generating the long keyboard and mouse-move sequences that typically make up the bulk of an *ST Control* script.

Two types of recording process are available: Record, which maintains a straight record of keypresses, mouse locations, and clicks; and Record with Waits, which takes note of the time (actually the number of interrupts) between events, as well. The latter type of recording is used primarily for managing special situations in which inter-event time is significant from the user's point of view—self-running programs that demonstrate applications are a good example.

On the machine side, *ST Control* normally syncs itself to the OS to avoid having events "drop on the floor," but a range of commands exists to alter playback speed and defeat synchronization in special situations. More on this later.

Starting to record is simply a matter of invoking *ST Control* and selecting one of the Record options from the Options menu. The accessory then evaporates, and you are back in your application (or on the desktop). Recording begins when you press the 8 key on the main keyboard, stops or pauses when you press 0, and restarts when you press 9 (which also serves to mark the recording in that location so that a carriage return is placed there in the final output source). When Record is active, you must enter the characters 8, 9, and 0, and their shifted equivalents (the symbols *, (, and)) from the numeric keypad.

ST Control records only keypresses and releases, and mouse button presses and releases (along with the location of the mouse cursor when these events occur). It does not record mouse movements in general, because this would involve keeping track of an enormous amount of data—data that is of ambiguous value in most situations.

The upshot is that *ST Control* generally records enough data to reproduce the *effect* of user behavior, without reproducing the appearance of user behavior. Think about it: copying a file from one window to another demands that you go to the file icon, hold down the mouse button, then go to the destination window and release the mouse button. As long as your cursor touches the icon and the destination window in proper synchrony with the button press and release, the system doesn't care where the cursor goes in the meantime.

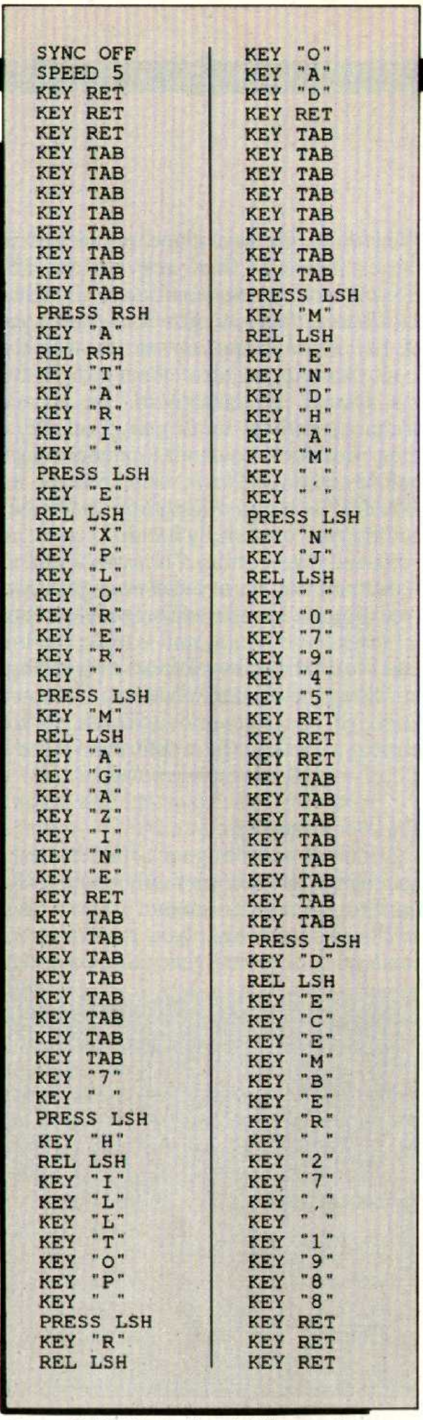
Aside from its inability to record freehand drawing, which is of little consequence, there is only one situation I encountered in which it becomes important to consider the failure of the program to record absolute mouse movement. This occurs when you select from drop-down menus.

Normally, it is necessary only to move the mouse pointer into the menu bar to elicit a drop-down, but if you want to have *ST Control* record this action, you must click on the menu item so that *ST Control* records the mouse position at that moment. Later, in editing the recorded script, you can remove the unnecessary click, which doesn't have any effect under GEM, anyway).

Each time you perform an action that *ST Control* considers significant, the event is recorded and your machine makes a low, buzzing sound to indicate the fact. About 200 separate events can be recorded during a single session. When 80% of buffer capacity is reached, the sound changes; and when the record buffer is full, the sound stops and recording terminates.

The limit of 200 events is not burdensome, since source code generated from the recording can be duplicated, edited, and added to other code. Thus you can build up comprehensive scripts over several recording sessions.

When you are finished recording, you re-invoke *ST Control*, open a document (if one isn't already open), and select Write to convert the recording into *ST Control* language. The text appears rapidly, already highlighted as a block for copying or moving into pre-existing text. If you have written the block into an empty document, you can assume that it is syntactically correct (*ST Control* doesn't make syntax errors when



```
SYNC OFF
SPEED 5
KEY RET
KEY RET
KEY RET
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
PRESS RSH
KEY "A"
REL RSH
KEY "T"
KEY "A"
KEY "R"
KEY "I"
KEY " "
PRESS LSH
KEY "E"
REL LSH
KEY "X"
KEY "P"
KEY "L"
KEY "O"
KEY "R"
KEY "E"
KEY "R"
KEY " "
PRESS LSH
KEY "M"
REL LSH
KEY "A"
KEY "G"
KEY "A"
KEY "Z"
KEY "I"
KEY "N"
KEY "E"
KEY RET
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY "7"
KEY " "
PRESS LSH
KEY "H"
REL LSH
KEY "I"
KEY "L"
KEY "L"
KEY "T"
KEY "O"
KEY "P"
KEY " "
PRESS LSH
KEY "R"
REL LSH
KEY "O"
KEY "A"
KEY "D"
KEY RET
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
PRESS LSH
KEY "M"
REL LSH
KEY "E"
KEY "N"
KEY "D"
KEY "H"
KEY "A"
KEY "M"
KEY ","
KEY " "
PRESS LSH
KEY "N"
KEY "J"
REL LSH
KEY " "
KEY "0"
KEY "7"
KEY "9"
KEY "4"
KEY "5"
KEY RET
KEY RET
KEY RET
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
KEY TAB
PRESS LSH
KEY "D"
REL LSH
KEY "E"
KEY "C"
KEY "E"
KEY "M"
KEY "B"
KEY "E"
KEY "R"
KEY " "
KEY "2"
KEY "7"
KEY ","
KEY " "
KEY "1"
KEY "9"
KEY "8"
KEY "8"
KEY RET
KEY RET
KEY RET
```

Figure 2. ST Control script for starting a business letter. Note use of SPEED and SYNC commands to hurry up typed output.

generating its own language), so executing it is a breeze. You just select Compile from the Options menu, and in a few seconds, you are ready to go.

Thereafter, selecting Run returns you to your application or to the ST Desktop. The macro run is started (or restarted, if paused) by pressing Control-Alternate. Left Shift-Alternate stops or pauses playback. Playback can be paused in progress by holding down Control-Left Shift. Restart is immediate when the keys are released.

During playback your ST appears to be under the control of a mechanical ghost. The cursor moves around, things click and double-click, windows open, and keystrokes fill the air—all without benefit of human hand. It's neat. While the default playback speed is extremely deliberate, in order to be sure of staying in synch with GEM and your application, this can easily be changed by editing the script in minor ways (see Figure 2 for an example).

Unfortunately, *ST Control* can save only one text file and one compiled module in memory at a time. This makes it difficult to use collections of macros in a transparent fashion, at least if you intend to assemble them entirely by recording. The solution to this problem is to learn *ST Control* language.

ST Control Programming

A typical macro processor lets you record collections of macros and assemble them in memory or on disk, each coordinated with a particular keystroke or behavior that elicits it. This type of macro processing has the advantage of being fairly simple and easy to use. The disadvantage is that it creates macros that consist only of keystroke representations. They contain no logic of their own and cannot respond to circumstances.

The cursor moves around, things click** and **double-click, windows open, and keystrokes fill the air—all without benefit of human hands. It's neat.

ST Control is designed on quite a different basis. Under *ST Control* there is no simple, systematic framework for organizing collections of recorded processes. Instead, there is a nonsystematic basis of immense power—a programming language capable of tying recorded processes together in comprehensive meta-applications.

ST Control language is much like Basic. It supports numeric variables and arrays, can evaluate mathematical and logical (Boolean) expressions, and has a range of decision-making (IF), flow-control (GOTO>label, GOSUB>label, RETURN), and looping (FOR, NEXT) structures that can be used to express logic of arbitrary complexity. It also supports, as you would expect, a rich set of commands and functions for controlling and receiving input from the *ST Control* environment—from the user, on the one hand, and the application on the other.

Writing programs in *ST Control* is not difficult—anyone who has programming experience in Basic will feel at home right away. Moreover, the largest portion of an *ST Control* program will typically consist of procedures produced painlessly by direct recording—i.e., you must write only such code as is required to tie the whole thing together.

For example, suppose you wanted to assemble a comprehensive collection of macros for use with *1st Word*. The first step might be to decide on a range of keypresses to associate with your macros—say, A, B, C, D, and E. Then, you would devise a logic that would receive a keypress and execute the associated macro script.

This simple logic is easy to set up in *ST Control* language (see Listing 1). As you can see, the pattern can be extended to manage any number of macro routines, each of virtually arbitrary complexity. The routines themselves can be recorded one-by-one and simply pasted into the control logic. When the program is complete, it can be compiled and saved to disk. Thereafter, you need only load and execute the compiled module to have access to the full set of macros by two-key command.

This kind of logic, although simplistic, solves the general problem of assembling macro collections very nicely. You need load and run the compiled code only once at the beginning of the session. Thereafter, to execute a macro, you simply press Control-Alternate to execute the program and another key to select the macro you want. When the macro is finished executing, you press Left Shift-Alternate to put *ST Control* back to sleep.

```
* Start of program
>START
* Examine keypress from user, exit if not A, B, C, D, or E
K = INKEY
IF K = 0 GOTO START
* Now execute chosen routine ...
IF K = "a" GOTO R1
IF K = "b" GOTO R2
IF K = "c" GOTO R3
IF K = "d" GOTO R4
IF K = "e" GOTO R5
BREAK

>R1
* Routine corresponding to "a" goes here
GOTO START
>R2
* "b" Routine goes here ... etc.
GOTO START
>R3
* "c" Routine
GOTO START
>R4
* "d" Routine
GOTO START
>R5
* "e" Routine
GOTO START
```

Listing 1. ST Control logic for managing a simple macro collection.

In examining the program in Listing 1, certain things about *ST Control* language should become more clear. *ST Control* is capable of capturing input from mouse and keyboard, with or without passing it on to a controlled program. This permits the programmer to erect conditional filters that "enhance" his input to a program in certain situations and leave it alone in others. The available input and output functions do not comprise a rich field of possibilities on their own, but they can be combined fairly easily to provide the equivalent of more powerful functions.

For example, *ST Control* does not support string input, nor does it support string variables. However, such constructs are easy to derive by using single-character input in loops, assembling received character codes in a numeric array (which can have up to 64 dimensions).

ST Control logic is simple and linear. IF/THEN decision-making statements are always terminated with GOTOs or GOSUBs. Yet, as every programmer knows, these simple facilities suffice to produce any imaginable kind of flow-control.

ST Control executes its programs in $^1/_{60}$ of a second interrupt intervals, and its designers have intelligently opted to leave timing control up to the individual programmer.

Commands exist to synchronize the output functions of an *ST Control* program with GEM (eliminating the possibility that events requested by *ST Control* will be ignored by the operating system) and to turn this synchronization off, allowing more speed in certain situations.

Commands also exist to control the number of interrupts *ST Control* will let pass between attempting output operations and the number of non-output (i.e., math and logic) statements it will execute during an interrupt interval. Thus, although defaults guarantee fair performance in any situation, the programmer retains considerable authority in optimizing the performance of *ST Control* for specific circumstances.

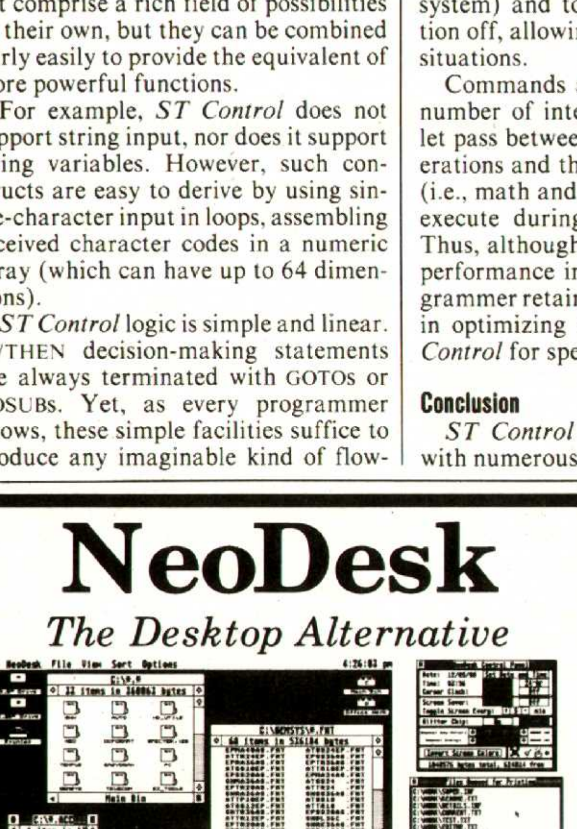
Conclusion

ST Control is a powerful program with numerous potential applications. I can imagine it being used as the basis for distributing application-specific macro packages or meta-applications for consumption by the non-technical user. The program also has clear utility as a device for developing and distributing self-running demonstrations and tutorials relating to applications software. For the programmer, it presents an interesting challenge and a new way of looking at the whole question of user-computer interaction.

ST Control is not, however, a simple macro processing system. The truly cyberphobic non-techie should probably avoid it on principle, since there is really no way to use it effectively without learning how to program.

The productivity purist and the power user should consider *ST Control* in comparison with other, simpler macro processing systems, to determine whether the ease-of-use/capability tradeoff makes sense in his particular situation.

So, consider your needs carefully, and if, after doing so, you conclude that you need the Cadillac of macro processors, buy *ST Control*. ■



Atari SFP004

A floating point coprocessor peripheral and developer kit for the Mega ST

By WAYNE RAMSEY

Atari has begun releasing to developers small quantities of its SFP004 Floating Point Coprocessor peripheral for the Mega STs: a daughterboard containing a 16 MHz Motorola 68881 FP processor. In the current generation of STs, floating point calculations are performed in software by the 68000—a relatively difficult and time-consuming process. The 68881 on the SFP004 peripheral performs these same calculations in hardware, in parallel with the CPU, vastly improving system throughput for math-intensive applications.

Naturally, software must be modified to take advantage of the SFP004. Specifically, the 68881 is accessed as a peripheral device via nine registers mapped into high memory. The most important of these registers are the control register, used to give commands to the chip; the status register, used to determine the current status of the chip; and the operand register, used to pass operands to the chip and recover results from it. The usual order of operations for using the 68881 is as follows:

1. Wait for the chip to be idle (determined by polling the status register).
2. Write a command to the command register.
3. Wait for the chip to be ready (poll status register).
4. Transfer operands by writing to the operand register.
5. Await results (poll status register).
6. Recover results by reading from the operand register.

Note that the SFP004 does not operate via interrupts, since the processing of an interrupt exception actually takes more time than an FP calculation. Programmers are nevertheless free to employ the 68000 as they wish during the polling intervals at steps 1, 3 and 5.

The 68881 is designed to process operands represented in IEEE P754 floating point format. Though it is capable of converting from other formats, such conversions typically consume most of the speed advantage gained by using the 68881 in calculations.

An IEEE float is represented in four bytes, comprising an overall sign bit, 8 bits of exponent (a bias of 127 is automatically subtracted from the exponent field, permitting the representation of exponents between 128 and -127), and 24 bits of mantissa (another 32 bits of mantissa is added for double- precision values). Currently, Alcyon C, Megamax Laser C, and Manx (Aztec) C all use the IEEE format for representing floats—Mark Williams C employs the DECVAX format, but reports that it is in the process of conversion to the IEEE standard.

How software will support the 68881 is up to the designers. The SFP004 peripheral will not affect system performance at all for software that is not aware of it. Software designed to make use of the peripheral can do so by addressing the 68881 directly (in which case the software will fail if the SFP004 peripheral is not in place) or by employing emulation functions that check for the presence of the SFP004 and use it if it is available, substituting software floating point functions if it is not.

The advantage of the former approach is that it permits maximum speed of execution and opens the way to exploiting opportunities for running the 68000 and the 68881 in parallel—at the expense of software compatibility with all STs. The advantage of the latter approach is that it permits software to exploit some of the speed advantage offered by the 68881, while retaining compatibility with all systems in the market. The first approach is the one most likely to be chosen by vertical-market products that would be unfeasible without the coprocessor; the latter by applications of more general appeal and feasibility.

The 68881 is designed to process operands represented in IEEE P754 floating point format.

C-compiler manufacturers will probably modify their compilers to enable them to produce either type of code on demand. In the meantime, provided the FP format of the compiler is IEEE-compatible, it is possible to use macros to access the 68881 directly. A group of such macros is illustrated in the SFP004 documentation. Manx, Megamax, and Mark Williams have all said that support for the SFP004 is forthcoming.

The SFP004 is currently being distributed with a demonstration disk containing FP benchmark, ray tracing, and Mandelbrot set generation programs. Some of this software can be set in three different modes—to use the 68881 directly, to use only software floating point emulation, or to use "compatibility" routines that check for the presence of the 68881 each time an FP calculation must be performed and use it if it is available—providing a complete standard for performance comparisons.

Compatibility routines used in the demos were created using Alcyon's LIB81 library, which replaces LIBM—the math library—to provide transparent 68881 support for Alcyon C programs. Neither of these libraries is distributed with the SFP004 Developer's Kit, due to license restrictions. However, developers may receive the binary files for LIB81 at no charge by signing and returning a form enclosed with the SFP004 documentation.

As of this writing (December, 1988) it is not known when the SFP004 will be available in quantity. ■

Turboword

Micromiser introduces a capable, full-featured 80-column word processor for Atari 8-bit systems

By DAVID NOYES

If you have been waiting for a reason to purchase the Atari XEP80 Interface Module, wait no longer. *Turboword*, a capable 80-column word processor, is now available for any 48K Atari 8-bit computer.

If you have or decide to purchase the XEP80 and *Turboword*, I highly recommend a monochrome monitor, because, while the 80-column display is legible on a composite color monitor, I have found the text quality on the monochrome monitor to be every bit as good as the quality of text output by equipment originally designed for 80-column display.

An XEP80 handler has been built in to the program, so *Turboword* is ready to boot as it comes out of the box. It is not copy-protected and comes with a spelling checker on the flip side of the disk. The program can be used with most DOSes, including SpartaDOS, MyDOS, and DOS XL, and will accommodate most RAMdisk arrangements for those users who want increased speed.

Turboword is feature-laden and, as is the case with most of the word processors I have purchased over the years, contains several functions that I will probably never use. I am sure, however, that some future Pulitzer prize candidate will find them useful.

Mail Merge and Macros

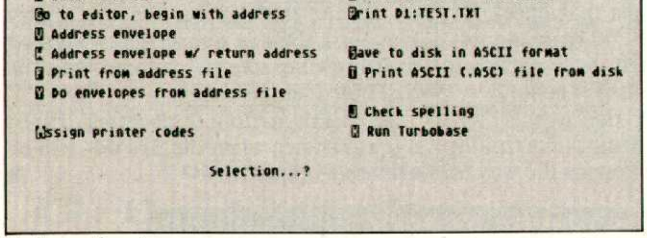
Turboword is menu-driven, which means that, you don't have to store a great many commands in your already over-taxed brain. If you do get stuck, you can access the Help screen from the editor for more details.

As you type and edit, formatting takes place automatically. Change margins, for example, and instantly your text appears as the new parameters dictate.

The mail merge features allows you to create form letters by merging in variable information such as names and addresses. A letterhead file allows you to load in a variety of letterheads, should you happen to need more than one. To be honest, this is a feature that I have never used with any word processor, but I can see that it would be useful for people who operate several businesses or who need to write business letters on personal letterhead.

Similar to mail merge is the macro feature, which allows you to store and insert frequently used phrases and passages into your documents without retyping. Again, I consider this a frill, but others may find it a significant timesaver. Also available to save you typing are automatic headers; you type the heading that you want to appear at the top of each page only once, and the program sees that it gets printed on every page you produce.

In addition to some of these more esoteric functions, *Turboword* offers the basic functions that you expect to find in any well-designed word processor. Blocks of text can, of course, be moved and deleted, and strings of text can be searched for (you know that somewhere in this 42-page document you misspelled *receive*, and you want to locate it . . . no problem). The program does not, however, offer a search and replace option; once the offending word is located, the replacement must be done manually.



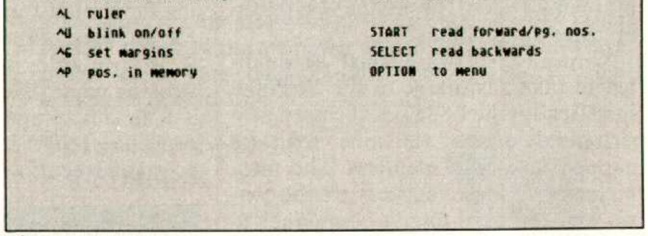
```
               Turboword - Copyright 1988 MicroMiser Software

                          Starting D1:TEST.TXT

0-8 change drive #                      Begin pg #1 print pgs 1 to 9999
Name file                               Z Number page 1 - *NO*
Editor -- file D1:TEST.TXT              Margins 5,75
                                        Lines text per page 54
Disk operations                         Overall page lines 66
Quick directory                         Hold between pages - *NO*
Init ramdisk (drive #8)                 Copies 1
                                        Review D1:TEST.TXT
K Enter address                         U Printer port:  *parallel*
Go to editor, begin with address        Print D1:TEST.TXT
U Address envelope
C Address envelope w/ return address    Save to disk in ASCII format
F Print from address file               T Print ASCII (.ASC) file from disk
Y Do envelopes from address file
                                        J Check spelling
Assign printer codes                    X Run Turbobase

                          Selection...?
```

Turboword main menu.



```
              -TURBOWORD- Copyright 1988 Micromiser Software

   ^Q  cursor top                       ^T  top of file
   ^A  cursor left                      ^B  bottom of file
   ^Z  cursor bottom                    ^X  next screen/pg. nos.
   ^S  cursor right                     ESC, ^ESC  blank and return
   ^H  cursor horiz center
   ^V  cursor vert center               ▪...▪  mark block
                                        ^E  erase all ▪'s
   ^I  insert/overwrite mode            ^D-^D  delete block
   ^R  show spaces, returns             ^C-^C  copy block to cursor
   ^F  find search string               ^M-^M  move block to cursor
   ^L  ruler
   ^U  blink on/off                     START   read forward/pg. nos.
   ^G  set margins                      SELECT  read backwards
   ^P  pos. in memory                   OPTION  to menu
```

Editor help screen.

Using the Program

On several key fronts, *Turboword* offers you a choice. Insert a letter or type over it (I prefer insert mode) . . . the choice is yours. The same goes for cursor selection—blinking or solid, you choose.

For navigating around your document, you use Control-arrow key combinations, and you can scroll up and down simply by pressing the Select or Start key, which is a very handy feature.

For the ultimate in ease-of-use, try using a RAMdisk with *Turboword.* I created a 1028-sector RAMdisk on my 1-meg ICD MIO Board using the public domain MyDOS 4.5 and copied to it all files from both sides of the *Turboword* disk, including the spelling checker. Read/write operations were much quicker, as was the spell checking process.

Speaking of speed, *Turboword* is a mixture of machine language and Basic routines. The use of Basic XL or Basic XE will increase the speed of those routines, and *Turboword* along with them.

Because *Turboword* forces a Save to the original filename when you exit the editor (via the Option key), I highly recommend that you save any important or lengthy file to another disk prior to editing. Likewise, I recommend saving in a normal fashion to another disk before Saving ASCII. Although you can load ASCII files into *Turboword*, the first four bytes of each file disappear when you do so, wreaking havoc with the format of your document.

Micromiser has promised that the January update of *Turboword* will provide the ability to exit the editor without invoking as Save. This will ameliorate the problem somewhat, but the ability to save to a different filename would eliminate it.

Turboword

System: 48K Atari 8-bit

Required equipment: Printer, XEP80, monitor (monochrome recommended)

Copy protection: None

Summary: Very capable word processor provides high quality 80-column output.

Price: $49

Manufacturer:
Micromiser Software
1635-A Holden Ave.
Orlando, FL 32809
(407) 857-6014

While using *Turboword* I came across a feature which I felt was necessary, yet missing—the ability to format a disk from within the word processor. I later learned that the feature is implemented but undocumented. When DOS functions are selected from the menu and the first screen of choices appears, you have only to type F to format a disk in single (single only) density.

Be careful! The potential for disaster is enormous; you could lose your current file, all your files, and even your program. The presence of this feature underscores the importance of making backups. *Turboword* is not copy-protected, so there is no excuse for failing to have a backup copy of the program, and of course, good practice dictates that you back up your data disks frequently.

My discovery of the undocumented formatting capability served to highlight another nice feature of *Turboword*—support. Micromiser takes calls from 1:00 to 5:00 p.m. EST Monday through Friday to answer your questions, guide you through confusing operations, and even listen to your suggestions on ways to improve their products.

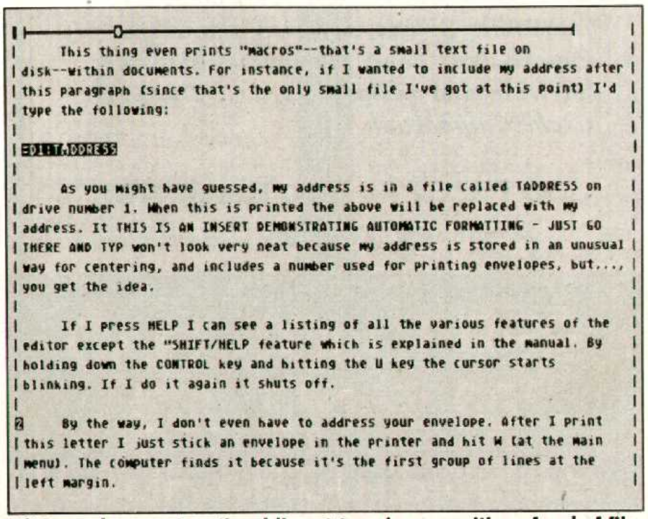
Printing

Currently, *Turboword* will not print through the printer interface of ICD's MIO Board or through the built-in interface of Trak drives. It can, however, use the interfaces in the XEP80, the P:R: Connection (by ICD), and the Atari 850 interface. A spokesman for Micromiser told me that the P:3 enhancement in the update should allow use of the printer interface of the MIO Board.

He also noted that a LISTable version of the Basic print routines, along with documentation, will be made available for Basic programmers who want to make their own modifications. The cost will be nominal.

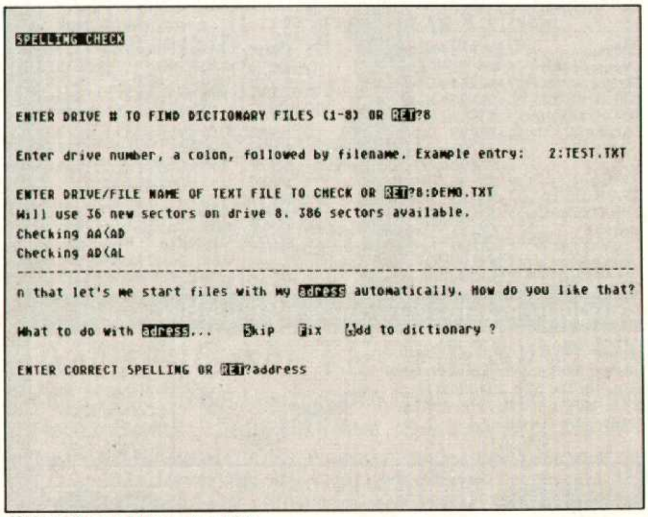
Turboword comes with a built-in Epson-compatible printer driver; if you don't own an Epson or compatible, you can assign printer codes as a main menu selection. And by using inverse characters to embed codes in edited text, you can turn boldface, italic, condensed, and expanded print typefaces on and off from within a document.

Strange as it may seem, the only real problem that I had with *Turboword* was getting used to the 80-column display. I was so accustomed to *not* seeing what I



```
|        This thing even prints "macros"--that's a small text file on          |
|disk--within documents. For instance, if I wanted to include my address after |
|this paragraph (since that's the only small file I've got at this point) I'd  |
|type the following:                                                           |
|                                                                              |
|D1:TADDRESS                                                                   |
|                                                                              |
|        As you might have guessed, my address is in a file called TADDRESS on |
|drive number 1. When this is printed the above will be replaced with my       |
|address. It THIS IS AN INSERT DEMONSTRATING AUTOMATIC FORMATTING - JUST GO    |
|THERE AND TYP won't look very neat because my address is stored in an unusual |
|way for centering, and includes a number used for printing envelopes, but..., |
|you get the idea.                                                             |
|                                                                              |
|        If I press HELP I can see a listing of all the various features of the|
|editor except the "SHIFT/HELP feature which is explained in the manual. By    |
|holding down the CONTROL key and hitting the U key the cursor starts          |
|blinking. If I do it again it shuts off.                                      |
|                                                                              |
|        By the way, I don't even have to address your envelope. After I print |
|this letter I just stick an envelope in the printer and hit W (at the main    |
|menu). The computer finds it because it's the first group of lines at the     |
|left margin.                                                                  |
```

Editing a document; optional line at top shows position of end of file in text buffer (25K).



```
SPELLING CHECK



ENTER DRIVE # TO FIND DICTIONARY FILES (1-8) OR RET?8

Enter drive number, a colon, followed by filename. Example entry:   2:TEST.TXT

ENTER DRIVE/FILE NAME OF TEXT FILE TO CHECK OR RET?8:DEMO.TXT
Will use 36 new sectors on drive 8. 386 sectors available.
Checking AA<AD
Checking AD<AL
----------------------------------------------------------------------------
n that let's me start files with my adress automatically. How do you like that?

What to do with adress...    Skip   Fix   Add to dictionary ?

ENTER CORRECT SPELLING OR RET?address
```

Using the spelling checker.

was going to get on the printed page, that it took me quite a while to get used to the luxury of WYSIWYG.

When it came to actually using *Turboword*, I found that I was able to navigate fairly easily without frequent reference to the 28-page manual that comes with the disk. This is not as much a measure of my skill or intelligence as it is a measure of the user-friendliness of *Turboword*, and it is a good thing, too, because the manual is a bit difficult to use and lacks an index.

More on the Update

The update disk to which I have referred was scheduled for release after January 15, 1989 (at no charge to owners of record up to January 31, 1989). It will be sent automatically to registered owners of the original version of *Turboword*.

The updates consist of several enhancements to the program. Among the new features are the following: SpartaDOS compatibility, right justification, two-column printing, even/odd (page) print capability, and word count capability. In addition, the upgraded version will allow you to exit at will from the spelling checker, print to P:3 (which should allow printing through MIO), autoprint files (via a statement at end-of-file), copy files with one drive, and return to the menu without saving.

***I have found the text quality** on the monochrome monitor to be every bit as good as the quality of text output by equipment originally designed for 80-column display.*

On a somewhat grander scale, the new version of *Turboword* will be compatible with the 80-column version of *Turbobase*, Micromiser's full-featured, business-oriented database. It will also be compatible with Micromiser's soon-to-be-released *Turbofile*, an 80-column database that will lack some of the features of *Turbobase* but should be more appropriate for the non-business environment and will be priced accordingly.

Just as years ago, *VisiCalc* alone was, for many people, useful enough to justify the purchase of a computer, *Turboword* may well be useful enough to justify the purchase of the XEP80. It was for me, and I haven't been disappointed.

Turboword is definitely a worthwhile investment of users of 8-bit Atari computers who need the flexibility and capability of a full-featured word processor. The program is great the way it is, and the promised updates should make it even better. ■

American Techna-Vision

For Orders Only - 1-800-551-9995
CA. Orders / Information 415-352-3787

- No surcharge for VISA/MasterCard
- Your card is not charged until we ship

800 4 PIECE BOARD SET

Includes Main Board, Power Supply Assembly, CPU Module and 10K Revision B Operating System Module. All boards are new, tested and complete with all components. $\$28^{50}$

WORD PROCESSORS

• PaperClip (Disk)	$29.95
• AtariWriter (Cart.)	$29.95
• Bank Street Writer (D)	$17.50
• Cut & Paste (Disk)	$17.50
• Letter Wizard (Disk)	$17.50

FLAT SERVICE RATES

1050 DISK DRIVE	$75.00
810 DISK DRIVE	$69.50
850 INTERFACE	$39.50
800 COMPUTER	$49.50
1200XL COMPUTER	$49.50

Flat rates include Parts & Labor, 60 day warranty. Include $7.00 shipping & insurance

800/400 MODULES

NEW PARTS COMPLETE WITH IC'S

$\$9^{50}$ EACH

- CX853 16K Ram Module
- 800 Main Board
- 800/400 CPU with GTIA
- 800 10K "B" O.S. Module
- 400 Main Board
- 400 Power Supply Board

800 Power Supply Board $14.50

INTEGRATED CIRCUITS

$\$4^{50}$ EACH

• CPU	CO14806
• POKEY	CO12294
• PIA	CO14795
• GTIA	CO14805
• ANTIC	CO12296
• CPU	CO10745
• PIA	CO10750
• CPU	CO14377
• DELAY	CO60472
• 2600 TIA	CO10444
• PIA	CO12298
• CPU	6507
• PIA	6532
• RAM	6810
• CPU	6502B

CO61598	$20.00	CO21697	$15.00
CO61991	$15.00	CO25953	$9.50
1050 Rom	$13.50	5713	$5.25
CO61618	$20.00	CO24947	$15.00

1050 MECHANISM

Factory fresh TANDON mechs. make difficult repairs a snap. Units are complete with Head, Stepper, Spindle motor, belt etc. Just plug in, no difficult alignments or adjustments required. $\$47^{50}$

VISICALC SPREADSHEET

Unleash the computing power of your 8 bit Atari with Visicalc. Compute everything from home finances to high powered financial projections. Hundreds of uses. $\$19^{95}$ DISK

XL 40 PIN LSI CHIP SET

A Complete set of 40 Pin Large Scale Integrated Circuits for your 800XL, 600XL or 1200XL computer. Great for quick repairs! Set contains one each of the following: CPU, GTIA, ANTIC, PIA AND POKEY. $\$16^{95}$

REPAIR MANUALS

SAMS Service Manuals for the following units contain schematics, parts listings, labelled photographs showing the location of checkpoints and more! A special section gives oscilloscope and logic probe readings allowing you to narrow the malfunction down to a specific chip or transistor!
800, 800XL, 130XE, 400, 1025 and 1050 .... $19.50 each
520ST Service Manual. $37.50

MISC. HARDWARE

13 Pin Serial I/O Cable	$5.95
1050 Track 0 Sensor	$8.50
2793 1050 Controller IC	$19.50
Paddle Controllers (Pair)	$6.50
400 3 Piece Board Set	$19.50
Fastchip for 800/400	$15.50
Animation Station	$74.95
Rambo XL w/o Ram IC's	$39.95
Supra 2400 Baud Modem	$157.50
ACE Joystick	$7.95
850 or PR Modem Cable	$14.50
850 or PR Printer Cable	$14.50
Printer Interface	$39.95
I/O 13 Pin PC mount	$4.50
I/O 13 Pin Plug Kit	$4.50
ST 6' Drive Cable	$14.00
810 Door Latch Assy.	$10.00
1027 Transformer	CALL
U.S. Doubler	$29.95
ST Monitor Connector	$5.50
ST Drive connector plug.	$6.50
ST to 5/14" Drive Cable	$23.95

POWER PACKS

Exact replacement transformer for 800/400, 1050, 810, 1200XL, 850, XF551 & 1020 units. Part #CO17945. $\$14^{50}$

XL/XE SUPPLY

Power Pak for 800XL, 600XL 130XE, 65XE & XE Game. $\$25^{00}$

THE BOOKEEPER AND CX85 KEYPAD

You get both Atari's 8 bit professional bookeeping system and the handy CX85 numeric keypad for one low price. Packages factory sealed. $\$19^{95}$ 4 DISK SET

KEYBOARDS

New and complete subassembly. Easy internal replacement.

130XE/65XE	$35.00
800	$40.00
800XL	$29.50
400	$12.50

COMPUTER BOOKS

Atari Playground Workbook	$7.95
Hackerbook	$5.00
Inside Atari Basic	$5.00
Atari Basic Ref. manual.	$5.00
How to 6502 Program	$5.00
Programmers Ref. Guide	$14.95
Basic-Faster & Better	$22.95
Assembly Language Guide	$21.95
XE Users Handbook	$21.95
XL Users Handbook	$21.95
Advanced Programming	$19.50
Write Your Own Games.	$5.00
Mapping the Atari	$18.50

SERIAL I/O CABLE

High Quality, 13Pin $5.95

MAC-65 CARTRIDGE

6502 Machine language Macro-Assembler. First class tool for serious programmers... $59.95

ATARI 850 INTERFACE

Bare PC Board with parts list and crystal $7.50
Board & all plug in IC's ... $39.50

PR: CONNECTION

Serial/Parallel Interface for connecting modems and printers...... $65.00

BASIC CARTRIDGE

Exact replacement for 800/400/1200XL $15.00

EPROM CARTRIDGES

16K Eprom Board with case. Specify dual 2764 or single 27128 style. Gold contacts. $6.95

ANIMATION STATION

Graphics Design Tablet $74.95

•8 BIT SOFTWARE •

Cartridges for all 8 bit Atari computers

Millipede Cartridge	$10.00
AtariWriter Cart.	$29.95
Pac-Man Cartridge.	$4.00
Deluxe Invaders Cart.	$4.00
Journey to the Planets Cart.	$4.00
Donkey Kong Cart.	$5.00
Crossfire Cartridge.	$7.50
Springer Cartridge	$7.50
Turmoil Cartridge	$7.50
Choplifter Cart.	$10.00
Zone Ranger Cart.	$12.50
Silicon Warrior Cart.	$12.50
Math Encounter Cart.	$12.50
Learning with Leeper Cart.	$12.50
Up For Grabs Cart.	$12.50
Eastern Front Cart.	$10.00
PILOT Language Package	$17.50
Edt/Asm Cart. w/o man.	$15.00
ACTION O.S.S.	$59.95
MAC-65 O.S.S.	$59.95

DISK SOFTWARE

PaperClip	$29.95
Visicalc Spreadsheet	$19.95
Bookeeper + CX85 Keypad	$19.95
Mission Asteroid Disk	$4.00
Frogger Disk	$5.00
Fort Apocalypse Disk	$5.00
Spider Man Disk	$5.00
The Hulk Adventure Disk	$5.00
Wizard & The Princess	$5.00
Ulysses & Golden Fleece	$5.00
Human Torch & The Thing	$5.00
Musical Pilot Ed. Disk	$5.00
Con*putation Disk	$5.00
Debug Childware Disk	$5.00
Crystal Raider Disk	$5.00
Dispatch Rider Disk	$5.00
Master Chess Disk	$5.00
Speed King Disk	$5.00
Last V-8 Disk	$5.00
Chambers/Zorp Disk	$5.00
Pathfinder Disk	$5.00
Match Racer Disk	$5.00
Encounter/Questar Disk	$5.00
Alien Ambush Disk.	$7.50
Wombats Adventure (D)	$7.50
Sports Spectacular Disk	$7.50
The Gambler Disk	$7.50
Stratos Disk	$7.50
Fun in Learning Disk	$7.50
Fun in Numbers Disk	$7.50
Mind Mazes (Educational)	$7.50
Honey Craze Math (D)	$7.50
Dig Dug Disk	$9.95
Repton Disk	$9.95
Rear Guard Disk	$9.95
TechnaColor Dream	$9.95
Freaky Factory Disk	$9.95
Laser Hawk Disk	$9.95
Rocket Repairman Disk	$9.95
Cest'e La Ve Disk	$9.95
Womper Stomper Disk	$9.95
Tutti Frutti Disk	$9.95
Bug Off Disk	$9.95
Olin Emerald (Jr., Adv.)	$9.95
David's Midnight Magic Disk	$12.50
DropZone Disk	$12.50
Castle Wolfenstein Disk	$12.50
Beyond Castle Wolfenstein	$12.50
Aliants Disk	$12.50
Mouse Quest Disk	$12.50
Electronic Drummer Disk	$12.50
Stock Market Game	$12.50
Pirates of the Barbary Coast	$12.50
Nickerbocker Disk	$12.50
Midway Battles	$14.95
MoonMist Adventure (D)	$14.95
Mind Shadow (64K)	$14.95
Sea Stalker (D)	$14.95
Master Typing Tutor	$15.00
Bank Street Writer	$17.50
Cut & Paste W.P. Disk	$17.50
SpartaDOS Const. Set	$39.50

DATASOFT DISKS

Zorro Disk	$9.95
Saracen Disk	$9.95
Gunslinger Disk XL/XE	$9.95
Crosscheck Disk	$12.50
Mercenary Disk	$12.50
The Goonies Disk	$12.50
Conan Disk	$12.50
Never Ending Story (64K)	$12.50
221 Baker Street (64K)	$17.50
Letter Wizard W.P. (32K)	$17.50

CALL TOLL FREE 1-800-551-9995

IN CALIF. OR OUTSIDE U.S. CALL 415-352-3787

AMERICAN TECHNA-VISION

Mail Order: 15338 Inverness St., San Leandro, Ca. 94579
Repair Center: 2098 Pike Ave., San Leandro, Ca. 94577

Terms: NO MINIMUM ORDER. We accept money orders, personal checks or C.O.D.s. VISA, Master/Card okay. Credit cards restricted to orders over $20.00. No personal checks on C.O.D. - Shipping: $4.00 shipping and handling on orders under $150.00. Add $2.25 for C.O.D. orders. In Canada total $6.00 for shipping and handling. Foreign shipping extra. Calif. residents include 7% sales tax. All items guaranteed 30 days from date of delivery. All sales final..

Prices subject to change without notice. Send SASE for free price list. Atari is a reg. trademark of Atari Corp.

DOS XE— A First Look

Atari DOS grows up!

By ARTHUR BOCCHINO

Atari DOS has undergone relatively few significant revisions since its initial introduction in 1980. Of the four DOSes released by Atari for its 8-bit line, only DOS 2.0S and DOS 2.5 have achieved the degree of user acceptance and third-party support necessary to qualify them as "stable working platforms." Now, almost eight years after DOS 2.0S burst upon the scene, a completely new Atari DOS is on its way to dealer shelves. It is a grown-up DOS for the XLs and XEs that extends and builds upon the user-friendly features Atari users have become familiar with over the years, providing a level of functionality far greater than anything that has come before.

The DOS XE project began about the same time that Atari engineers first proposed the XF551 disk drive, Atari's first true double-sided, double-density peripheral for the 8-bit line. It became apparent that more than a trivial upgrade to DOS 2.5 was required if users were to exploit the added capacity of the XF551 conveniently. What was needed was a true hierarchical filing system—a grown-up system that would ease the transition into larger and larger mass storage media by providing the means to organize the hundreds or thousands of separate files such media could conceivably hold. Ready to take on the project was Bill Wilkinson, author of the original Atari DOS and of Atari Basic.

Wilkinson warmed to the task, but circumstances were to make circuitous what should have been a straightforward design and implementation process. First, the personal computing market hit a slide, forcing Wilkinson's company, Optimized Systems Software, into a retreating position. Much of the preliminary work for ADOS, as it was then called, was done on a lone Atari XE system, surrounded by packing crates, pressed into a corner of what had once been a fairly princely suite of offices.

DOS XE

System: Atari 8-bit

Summary: A powerful, user-friendly, well-designed operating system

Price: $10.00, plus $3.50 shipping

Manufacturer:
Atari Corp.
P.O. Box 61657
Sunnyvale, CA 94088

Then, commercial considerations first delayed, then accelerated the release date of the XF551. The upshot: the 551 was released in the second quarter of 1988—with DOS 2.5. Early adopters of the new drive had only SpartaDOS to turn to if they wished to exploit the native DSDD capability of the new hardware.

Now, in early January of 1989, the new Atari DOS, renamed DOS XE, is complete. It is bigger than was originally planned, and because it exploits memory in parallel with the OS ROM for certain of its functions, it is incompatible with the old 400 and 800 systems.

Still, even given this disappointment, it is hard to look at DOS XE and say "too little, too late." The fact is, the new DOS is too exciting. After a half-year of using the powerful XF551 as a pretty, color-matched 1050 replacement, it is a pleasure to plug in DOS XE and soar. Powerful, user-friendly, and well-designed, DOS XE is a grown-up platform that can carry the enhanced 8-bit line forward into a future of enhanced productivity.

I Think That I Shall Never See . . .

. . . another non-hierarchical filing system on my XE. DOS XE is the first Atari 8-bit DOS to offer a true, tree-structured directory system, just like MS-DOS and Mac OS HFS. Earlier Atari DOS filing systems were "flat"—a directory listing would reveal the name of every file on a disk.

By contrast, each DOS XE disk has a top-level directory (called the *root*) which can contain either files or subdirectories. These subdirectories may, in turn contain files or further subdirectories. The structure you end up with is like a sideways tree, with the root directory at the left, as shown in Figure 1.

The root directory, D1 (which also identifies the disk drive), contains two subdirectories, AWRITR (for AtariWriter files), and BASIC (for Basic files). The AWRITR subdirectory contains two of its own subdirectories: one for letters and one for reports. Each of these subdirectories contains a number of actual documents stored as text files. The BASIC subdirectory contains two files, GAME1.BAS and ARTIST.LIS, but no further subdirectories.

To refer to a file or a directory under DOS XE, you use an expression called a *pathname*, that traces the path from the root, down through the directory hierarchy to the file or directory you wish you identify. For example, in Figure 1, the file FEB.TXT could be referred to as D1>AWRITR>REPORTS>FEB.TXT. (Note the use of the > (greater than) symbol as a separator.)

Users accustomed to DOS 2.0S and 2.5 may find hierarchical directories and pathnames a little cumbersome and pointless at first. And so they are, perhaps, when only a maximum of 64 files can be saved on a single disk. But DOS XE has also lifted this limitation. A DOS XE directory can contain up to 1250 files and subdirectories, and there is no theoretical limit to the number of subdirectories (hence, files) a disk can contain, short of the absolute capacity of that disk.

Moreover, DOS XE is capable of accessing devices (or hard disk virtual partitions) containing up to 16Mb of data and of handling files up to 8Mb in length. At these capacities—and even at the 368K capacity of an XF551 DSDD disk—the organizational advantage of a hierarchical filing system becomes obvious. Figure 1, for example, shows how files of like type (AtariWriter text) can be grouped together but separated by purpose (into LETTERS and REPORTS).

In effect, then, a single DOS XE disk works like multiple disks set up in parallel or nested together to arbitrary limits of complexity. The only practical limit to this nesting, outside of disk size, is the fact that DOS XE cannot accept pathnames greater than 80 characters in length. To help you navigate this structure, DOS XE, like all hierarchical operating systems, never shows you more than the contents of the directory you are currently "in"—your *working directory*.

When you first boot DOS XE, your working directory will typically be the root, and a simple files listing will display only those files and the names of those subdirectories contained within the root directory. If you then change your working directory to one of the subdirectories and take a files listing, you will see only the files and subdirectories it contains.

Being able to set a working directory is like being able to specify the first part of a pathname automatically. For example, in Figure 1, if your current working directory was D1 (the root directory), you would be able to access the file JAN.TXT only by using the pathname AWRITR>REPORTS>JAN.TXT. But if you changed your working directory to REPORTS, you would be able to refer to that same file simply as JAN.TXT.

As a general rule, specifying the name of a file or directory *subsidiary to your current working directory* is simply a matter of supplying DOS with whatever additional information it needs to complete a path to whatever data-object you are interested in (this is called a *relative pathname* in operating system talk, because it describes a route to a file or directory *relative* to your current directory). Interestingly, however, it is more difficult within this framework to refer to the contents of directories at the same level as your working directory, or at higher levels.

In effect, a single DOS XE disk works like multiple disks set up in parallel or nested together to arbitrary limits of complexity.

In Figure 1, for example, if your working directory was REPORTS and you wanted (without changing directories) to see the files contained in the directory BASIC, you would have to give Files Listing a complete (*absolute*, in operating system talk) path for the contents of the directory in question—D1>BASIC>*.*. All of the file-handling functions of DOS XE work with reference to the contents of the current working directory unless a relative or absolute path is specified.

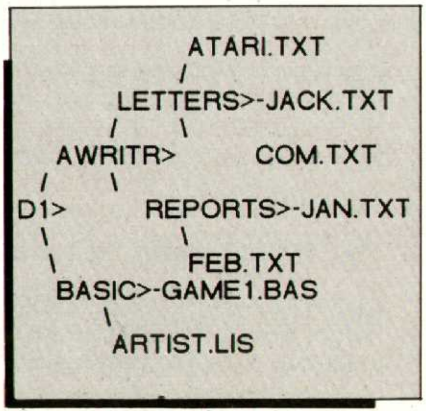


Figure 1. A sample DOS XE directory tree.

DOS XE menus

As you might expect, the facilities DOS XE places at your disposal to manage this kind of powerful filing system are sophisticated, indeed. When you first boot DOS XE, you are confronted with a simple main menu. From there, a keystroke will take you to the File Access menu, the Machine Language Access menu, or the System Functions menu, or assuming Basic is resident or a cartridge is installed, permit you to exit to either Basic or that cartridge. If not, the exit option is not offered.

The first thing I noticed about the main menu was that there was a date displayed in the upper right-hand corner of the screen. Yes, folks—DOS XE actually has a function to let you set the current date (not the current time, though) and uses this information to maintain the Date Created and Date Last Modified fields it stamps on your files.

The File Access Menu

Pressing F takes you to the File Access menu, where most common DOS functions are performed. In addition to the expected menu entries for getting a directory, initializing a disk, and copying, renaming, erasing, locking, and unlocking files, you will find quite a few new entries. You can view files, create and delete directories, append one file to another, and set the current working directory—all from this one menu.

Following and improving upon the tendency of earlier Atari DOSes to prompt for required input, the functions of DOS XE provide complete "training wheels" and appropriate defaults to lead even the most timid beginner through all the steps of basic disk management. A sample interaction with the Copy Files function is shown in Figure 2. Note that, in keeping with the new filing system, Copy Files can work with files alone or with entire subdirectories.

Where practical, all DOS XE file-management functions can accept wildcard specifications and can accept input and/or produce output to files or to devices (S:, E:, or P:). The Files Listing function, for example, asks LIST TO WHERE? after you have specified a source path (or accepted the default—a listing of the files in the current working directory). If you press Return, the listing is sent to the screen. Entering P: will send it to the printer, while entering D1>DIREC.LIS will send it to a newly-created file called DIREC.LIS.

The Machine Language Access menu

The second of the three DOS XE menus offers access to a variety of functions useful to programmers and power users. Commands on the Machine Language Access menu include those for running, loading, and executing binary files, saving ranges of memory in binary format, appending ranges of memory to compound binary files, and examining and changing the contents of RAM. Entries to these functions are made in hex or in decimal, again in response to carefully-worded prompts.

Although this is fairly high-tech stuff, the DOS XE manual not only explains what each option is for and how it works, but gives several practical examples of each function. Reproduced on the Machine Language Access menu are the all-important Files Listing and Working Directory commands, simplifying access to the system while performing the finicky and sometimes risky stunts associated with machine-level manipulation of the Atari.

Of particular interest here are the functions to display and change memory, and to append memory image units to compound binary files, since these constitute almost all the functions of a classic M/L monitor, less assembly and disassembly of single machine instructions. The Display Memory function lets you specify a start and end location, displaying the intervening bytes in hex and, where practical, as characters (screen control codes are represented as periods and inverse characters are normalized).

The Change Memory function lets you specify a start location, then begins displaying the values currently found at that and higher locations, letting you enter new values in hex or decimal, or not, as you prefer. The Append function lets you tack on a block of data, along with its load address, to a compound binary file. When the file is loaded, this block will also be loaded to position. The possibilities here are endless—the manual gives one great example of Appending the contents of system variables LMARGN and RMARGN to a binary utility program, causing these variables to be set (and the screen margins to be adjusted) automatically when the utility loads.

The System Functions menu

The third menu—System Functions—provides the ability to execute batch files of DOS XE commands, save an optionally configured version of DOS XE, duplicate an entire disk, and access disks formatted for DOS 2.0 and 2.5.

```
   COPY FILES

CAUTION!
THIS OPERATION DESTROYS PROGRAM AREA

PUSH START TO CONTINUE
PUSH SELECT TO STOP NOW
[user presses Start]

CONTINUING

COPY FROM WHAT FILE?WELCOME.BAS [return]
COPY TO WHAT FILE?NEWFILE.BAS [return]
COPY SUBDIRECTORIES OR ONLY FILES?
    [S]UBDIRECTORIES/[F]ILES(S or F)? F
[user presses Return to select files only]

VERIFY EACH NAME BEFORE PROCEEDING?
    [V]ERIFY/[N]O VERIFY(V OR N)? V
[user presses Return to choose verify]

ONE DRIVE, FILES ON SAME DISK?
    [S]AME DISK/[N]OT SAME(S OR N)? N
[user presses Return to choose different disk]

PUT FROM DISK IN DRIVE 1

PUSH START TO CONTINUE
PUSH SELECT TO STOP NOW
[user complies and presses Start]

COPYING: WELCOME.BAS
PUSH START TO COPY THIS FILE
PUSH SELECT TO BYPASS IT
[user presses Start to authorize copy]

CONTINUING

PUT TO DISK IN DRIVE 1

PUSH START TO CONTINUE
PUSH SELECT TO STOP NOW
[user complies and presses Start; copy is performed in one pass]

SELECT ITEM OR ESCAPE OR RETURN:
```

Figure 2. Using the DOS XE Copy Files function.

To a limited extent, DOS XE permits the advanced user to bypass (or rather, "hurry through") its menus by entering all the information required by a command on a single line. For example, the normal procedure required to create a new directory involves pressing the N key (for New Directory) from within the File Access menu, pressing Return, waiting for a prompt, then entering a path for the directory you wish to create. This procedure can be shortened by entering N *pathname* on the command line, and pressing Return at the end. DOS XE will interpret the space between N and your pathname as a Return keypress, display the intervening prompt, then fill it in automatically with the path you specified.

```
! CREATE SUBDIRECTORY "TEMP"
. FILEMENU NEWDIR TEMP
! LOAD EDITOR
. MLMENU RUNFILE MEDIT.COM
```

Figure 3. A sample batch file.

Any DOS XE command that can be completed with only normal keyboard entries can be expressed this way. Potentially dangerous commands that require function keypresses for authorization (PRESS START TO COPY FILES) cannot.

Still, this leaves quite a few commands that can be effectively executed in command-line form, and DOS XE can execute such commands from an ASCII batch file as easily as from the keyboard. For example, Figure 3 shows a batch file that might be used to create a directory called TEMP, then execute a text editor called MEDIT.

The file demonstrates several of the conveniences DOS XE offers for dealing with batch files. First, the "." operator works like the Esc key to move from any submenu to the main menu of DOS XE. Using it permits this macro to incorporate functions from two different submenus. Second, because DOS XE examines only the first character of each entry on a command line and ignores the rest, you are free to make substitutions for the sake of clarity—to use explanatory terms like FILEMENU instead of the cryptic F to activate different menu selections. Finally, the character ! causes the remainder of a batch file line to be taken as a comment and ignored by DOS XE for purposes of execution.

Batch files can be up to 511 bytes long and can be chained together using the Run File option from the System Functions menu (i.e., to cause one batch file to execute another, make the last line of your first batch file . S R *newbatch*). When DOS XE is booted, it searches for and autoexecutes (if found) any batch file called AUTORUN.BAT that is saved in the root directory.

DOS XE is configured using a program called SETUP.COM—a binary file run from the Machine Language Access menu. SETUP does three things. It creates an AUTORUN.SYS file that can initialize the 130XE RAMdisk, load the RS-232 driver, and auto-execute a Basic program. It lets you specify which drive types are attached to your system, including Atari 810, 1050, XF551, 130XE RAMdisk, or generic SSDD and lets you set trivial configuration options for these drives (writes with or without verify, etc.) And it lets you specify detailed information about the function of any drive. Once you have run SETUP.COM, you must use the Create DOSXE.SYS option of the System Functions menu to save a reconfigured version of DOS XE to disk.

DOS XE permits virtually transparent access to DOS 2.0S and 2.5 disks via the Allow DOS 2.X Access option on the System Functions menu. Code for older DOSes is loaded only when you request it, to save on memory during sessions when only DOS XE disks need be accessed. Once the Access function has been chosen, you can refer to any DOS 2.X disk in any drive on your system by beginning file specifications with A instead of D. All functions except those involving disk initialization and those which are dependent on the hierarchical file system unique to DOS XE can be performed on DOS 2.X disks.

DOS XE permits the advanced user ***to bypass its menus by entering all the information required by a command on a single line.***

DOS 3 is, once again, the orphan. It is not directly supported by DOS XE. However, a file called COPY3—XE.COM, included on the DOS XE master disk, permits conversion of files from DOS 3 disks to DOS XE format.

Software Compatibility

For the most part, any software that runs under DOS 2.X will run under DOS XE, since memory map and CIO entry point compatibility have been preserved with these earlier systems. DOS XE pathnames present only a minor problem.

First of all, programming languages and other software that permits the entry of long pathnames will be able to access DOS XE files in a straightforward manner. Programs that send filenames to the operating system in the form D1:FILE will at least be able to work with files in the current working directory, because DOS XE interprets colon-separated (i.e., DOS 2.X-type) filespecs with respect to the working directory—D1:FILE is interpreted as D1>*working directory*>FILE.

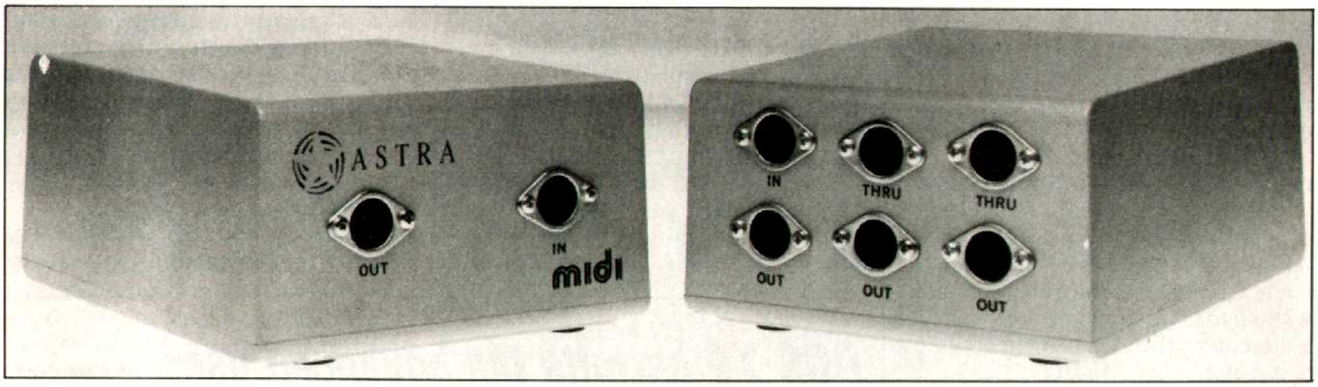
Finally, DOS XE contains a kludge that will even permit most software to change the working directory. The procedure involves using the Load File function to load a file which is really a subdirectory, terminating the expression with an = sign. When DOS XE is passed the name of a directory terminated in this way, it makes that directory the working directory and returns an EOF, for End-Of-File. Most software will interpret this as a simple error . . . but subsequent file access will be to a new working directory. Tricky, eh?

Documentation

In keeping with the friendlier approach of the new operating system, the DOS XE manual is among the clearest, simplest, and most complete manuals Atari has ever produced for an 8-bit product. Although most of the manual is written for the non-technical end user, ample information for programmers is also included.

Basic programmers will appreciate the detailed rundown of how DOS XE changes the use of NOTE and POINT and how new XIO function numbers have been pressed into service to give access to DOS XE extended functions from within Basic programs. Disk hackers and software developers will appreciate the detailed maps of DOS XE file structures.

DOS XE is a profoundly complicated product, and we haven't spent enough time with it to judge it in detail. In upcoming issues, we will be examining some of the technical minutiae of the system and assessing its impact on the user and on the third-party software and hardware marketplace. ■



Astra MIDI Distribution Box

Increasing the number of ports on the ST makes MIDI easier to manage

MIDI Distribution Box

System: Atari ST

Summary: Passive signal-splitter with two MIDI IN, two MIDI THRU, four MIDI OUT ports

Price: $59.95

Manufacturer:
Astra Systems
2500 South Fairview
Santa Ana, CA 92704
(714) 549-2141

The original MIDI specification suggested that each piece of MIDI equipment be provided with three MIDI ports, each with a distinct purpose. MIDI IN was to accept the incoming MIDI data stream from another device. MIDI OUT was to output a MIDI data stream commencing from the instrument and bearing no direct relation to data appearing at MIDI IN. MIDI THRU, which precisely reproduced the data stream received at MIDI IN (usually simply by wiring the two ports together), was to be provided as a courtesy, permitting the unit so equipped to act as a passive coupler in a MIDI ring or linear network, passing the data stream on to devices further down the line.

Unfortunately, many manufacturers didn't take the hint, and as a result, very few MIDI devices provide a MIDI THRU port that functions in the specified manner. Some, like the Casio CZ-series synthesizers, don't provide one at all—the result being that such units cannot be employed in a linear network, except at the end of the line.

Others, like the Yamaha RX-series drum machines, do support MIDI THRU, but do so by wiring MIDI IN to MIDI OUT, making the latter port function like MIDI THRU, so long as the device itself doesn't start producing MIDI data.

The result is that slaving multiple MIDI instruments to a single master sequencer—the normal object in a simple MIDI studio—can be a complex and confusing business. Dispersing outgoing sequencer data can be difficult in itself, given the peculiar collection of ports on most devices. Moreover, as more and more THRU ports are linked together in a chain, delays are eventually introduced and can become perceptible.

The solution to this problem is to multiply the number of MIDI OUT ports on the master sequencer, which is what Astra Systems' MIDI Distribution Box is designed to do. The Astra MIDI Box has MIDI IN and OUT ports on one side, matched to the MIDI interface of the ST. On the other side are an IN port, two THRU ports, and three OUT ports. The device is entirely passive, using no power.

Using the Astra MIDI Box, it is possible to connect a single ST master sequencer directly to up to five slave devices connected to the OUT and THRU ports, each of which will receive the entire sequencer data stream. A *star* network like this is much easier to manage than a linear or *ring* setup, and because the MIDI Box is a separate device with readily accessible plugs, it is easy to make changes to the system configuration required by the record-playback cycle.

The box is sturdily constructed of molded ABS plastic, color-matched to the ST. Astra warrants the box for 90 days against defects, though the most likely cause of failure in a device of this kind—detachment of a wire from a solder-point—can probably be fixed by most hobbyists at home. ■

By LAURA LIVINGSTON

ATARI ST POWER

COLOR COMPUTEREYES™

Incredible COLOR video digitizer. • The first and only full color digitizer for the ST • Uses standard video inputs like video camera, VCR, or video disk. • Works in all ST resolutions. Low res provides 16 shade black and white or full color pictures. • Pictures can be used with Degas, Neochrome, Powerprint and others. • Automatic calibration of contrast, brightness and white balance. • Plugs into cartridge port for easy set-up. • Capture your picture or that of your favorite star. **ONLY $199.95**

SPECIAL OFFER – Buy both Computereyes and Powerprint and SAVE 20.00 from the total.

MEGADISK

Ultra high speed solid state disk drive • 500% Faster than a Hard Disk • Provides almost instant booting • Like a RAM disk that's always loaded with your favorite programs and ready to use • One megabyte of Solid State storage • Built in battery back-up in case of power failures

MEGADISK is actually one megabyte of RAM that simply plugs into your cartridge port. It acts as an added disk drive that's ultra fast and always ready for use. Like a Hard disk, MEGADISK won't loose its memory when your computer is turned off. It comes with its own power supply and battery back-up system so its independent of your computer.

Megadisk can be configured according to your needs. • Set it up as one large disk • An 800K double sided disk and a 200K hardware print buffer • Or as two 400K single sided disks and a print buffer

Megadisk will work fine with your current system whether you have a hard disk and two drives or you're just getting started.

Megadisk is perfect for those who want the high speed of a hard disk for a lower price. Its even better for power users or software developers who may already own a hard disk and two drives but want extra speed and power. Megadisk can also emulate other cartridges for testing and back-up. In addition Megadisk can be used with Switch/Back to allow you to instantly jump between two full size one meg applications.

$299.95
Price Subject to change

Megadisk Clock Option – Adds a Clock/calendar card to your Megadisk cartridge. Contains replaceable Three year battery 29.95

Polydisk

Polydisk is a 512K version of a Megadisk. Polydisk gives you the same fast boot features, the high speed access, and the print spooler. Polydisk has a power supply (like Megadisk) but does not contain a battery back-up.

Note: Those with only 512K of main memory can use Switch/Back with a Polydisk, just like those with one Meg.

Polydisk (512K Solid state drive) **Only $199.95**
(Clock option card is also available for Polydisk $29.95)

PowerPrint ST — BLOW YOURSELF UP

Imagine your picture on a 6 foot poster. Create a business graph that can cover a wall. Quality output for posters, t-shirts, news letters, and more.

POWERPRINT

Whether it's a photo digitized with ComputerEyes, a masterpiece created with Degas, or the winning screen from your favorite game, POWERPRINT can print it with unequaled clarity and resolution. PowerPrint supports **ALL** ST resolutions. It prints multiple sizes up to **GIANT WALL SIZED POSTERS.** Print 16 shades for incredible detail. Print the whole screen or **ZOOM** in on just the part you want. POWERPRINT offers unique effects, including rotate, mirror and inverse options. Selective shading option allows you to print multi-color pictures on any printer by printing one color at a time (using color ribbons). Powerprint lets you capture and print almost any ST screen. Works with Star, NEC, Citoh, Gemini, EPSON, XM8048 and compatible printers. **ONLY $39.95**

Digisound ST

High Quality sound digitizer for the ST. This powerful hardware and software package lets you sample real world sounds and play them back on any Atari ST. Add special effects like Echo, Reverse, looping, pitch manipulation, mixing and envelope control. Turns your Atari keyboard into a musical instrument to play songs with your digitized sounds (also works with any MIDI keyboard). Digisound makes it simple to add sound to your own program, too! Unleash the incredible sounds in your ST with DIGISOUND. Supports sampling from 5 to 40Khz. DIGISOUND is the choice of the professionals. DIGISOUND was.used to create the voice in Chessmaster 2000, and other commercial programs.

DIGISOUND **ONLY $89.95**

DIGISOUND PROFESSIONAL

All the excellent features of DIGISOUND plus these great extras

LOGARITHMIC SAMPLING – Special hardware extends the sound quality far above the other ST sound digitizers. Logarithmic sampling and playback (external amplifiers only) greatly extends the dynamic range while reducing distortion and noise.

Internal Real Time Mixing – Input from a stereo and a microphone so you can sing over a tape. **$149.95**

SWITCH/BACK - Turn your ST into two; requires 1 meg - Hardware & Software	$69.95
ST PROTECTION TECHNIQUES Book & Disk Package	$39.95

ATARI 8-BIT POWER

ALPHA SYSTEMS is constantly innovating to provide more power for your 8-bit Ataris

NEW HACK PACK Special OFFER

The Alpha Systems HACK PACK contains all our finest products for making Back-up copies, Analyzing, Understanding and Protecting your Atari programs. It comes complete with Atari Protection Techniques (Book and Disk I), Advanced Protection Techniques (Book and Disk II), The Chipmunk, The Scanalyzer, The Impersonator and Disk Pack 1000. Worth over $150. Get them all for the special price of **Just $99.95**

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**

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.

(If you need a full list of what Chipmunk copies, call or write for our free catalog) **$34.95**

Scanalyzer Automatically scan & analyze commercial programs. Unlock programming secrets and learn from the masters **$29.95**

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

NEW CHEAT

Get more from your games with CHEAT. Tired of spending days trying to beat a game? Tired of getting stuck just when you need another life? Cheat is an innovative new product that gives you the chance you need to beat your favorite games. Cheat works with hundreds of Atari games to give you unlimited lives or power. End the frustration and get hours more enjoyment from your games. (Call or write Alpha Systems for our free catalog with a full list of the programs that work with Cheat) **ONLY $24.95**

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 • Player/Missile control • Load & Save Picture files • Sorting and Searching • Special Effects Graphics • Incredible Speed • Much, Much More • Over 150 programs. You've heard of the power of Assembler, now harness it for your own needs. **$24.95**

NEW PARROT II

An All New Parrot sound digitizer for your Atari. Parrot II is a sophisticated new hardware device that plugs into your joystick port. Parrot II has two inputs, One for a microphone and one for a powered source such as a tape player, radio or Compact Disk.

The Powerful Parrot II software lets you record sounds into your computer and play them back on any Atari. Parrot II turns your computers keyboard into a musical instrument with nine different sounds covering three octaves each. The sounds can be anything, a dogs bark, a piano, a complete drum set, a symphony or your own voice.

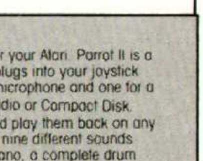
Parrot II lets you modify the sounds on a graphic display to create brand new sounds and special effects. Best of all, the sounds and voices can be put into your own programs that can be used on any standard Atari. Explore the world of digital sound and music. ONLY **$59.95**

Pre-Recorded Sound Disk More pre-recorded sounds for Parrot	**$4.95**
PARROT II Demo Disk (Does not require Parrot to run)	**$5.00**

NEW POP-N-ROCKER

a fast paced, multi-player trivia game that mixes questions with real songs (digitized with Parrot). Be the first to identify the songs and answer the music trivia questions. *Pop-N-Rocker* comes with three data disks and lets you add new questions so it will never get old. You can use a Parrot Sound digitizer to add new songs too! Use any kind of music from Rock to Classical to Nursery Rhymes. A new concept in entertainment and a perfect add-on for Parrot **$24.95**

COMPUTEREYES & MAGNIPRINT II +



GIANT WALL SIZED POSTERS.

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**

YOUR ATARI COMES ALIVE

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

24 HOUR HOTLINE 216-374-7469

VISA & MASTERCARD, ORDER BY PHONE, OR SEND MONEY ORDER TO:

ALPHA SYSTEMS 1012 SKYLAND DRIVE MACEDONIA, OH 44056 **FREE BONUS:** DELUXE SPACE GAMES (3 games on a disk) Free with any order of 3 or more items. Include $3.00 shp & hdlg (US Canada) Ohio res. add 5 1/2% sales tax. Foreign orders add $8.00 shp & hldg. Call or write for free catalog. Customer Service Line (216) 467-5665 M-F 9-3.

Total 100

Solution to an old friend . . .

By DAVID H. AHL

You may recall the innocent looking problem we posed in the Puzzles & Problems column of the July/August 1988 issue titled "Total 100." It read:

With the digits 1 through 9 arranged in order, using only addition and subtraction, it is possible to have them total 100. For example:

1+2+3-4+5+6+78+9=100

How many possible ways are there to do this?

We didn't publish the answer but challenged readers to submit a computer program to generate all the possible sequences.

The programs finally slowed to a trickle last month, and we are ready to choose a winner.

Incidentally, there are 12 ways to produce the sum of 100 using all nine digits in order. More than half of the programs—including some nicely coded ones—were disqualified because they missed one or more of the solutions. The solution most frequently missed was the one that begins with a minus sign:

-1+2-3+4+5+6+78+9=100

Programs were submitted in a wide variety of languages, including Fortran, C, Modula-2, Pascal, and several versions of Basic. This was a problem in which analysis of the approach to a solution and attention to good programming technique really mattered. Indeed, some programs took more than an hour to run, while others produced all 12 solutions in a matter of seconds.

We selected two programs as overall winners, a Basic program by Joseph Wrobel of Rochester, NY, and a Pascal program by Pierre Beauchemin of Hull, PQ.

The Basic Solution

Joseph's program (Listing 1) is written in GFA Basic for the ST. Among the Basic entries, it was the speed demon of

TOTAL 100

ATARI KEY

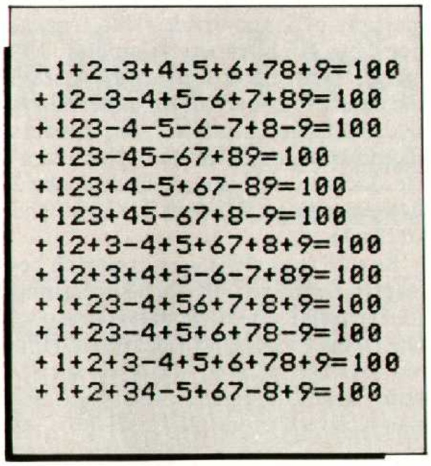
- Any Atari ST Computer
- GFA Basic

Listing 1.

```
Target_total%=100
Dim Sum%(9),Term%(9),Sign%(9)
Sum%(0)=0
Term%(0)=0
For Sign1%=-1 To 1 Step 2
  @Sumterm(Sign1%,1)
  For Sign2%=-1 To 1
    @Sumterm(Sign2%,2)
    For Sign3%=-1 To 1
      @Sumterm(Sign3%,3)
      For Sign4%=-1 To 1
        @Sumterm(Sign4%,4)
        For Sign5%=-1 To 1
          @Sumterm(Sign5%,5)
          For Sign6%=-1 To 1
            @Sumterm(Sign6%,6)
            For Sign7%=-1 To 1
              @Sumterm(Sign7%,7)
              For Sign8%=-1 To 1
                @Sumterm(Sign8%,8)
                For Sign9%=-1 To 1
                  @Sumterm(Sign9%,9)
                  If Sum%(9)+Term%(9)=Target_total%
                    @Print_equation
                  Endif
                Next Sign9%
              Next Sign8%
            Next Sign7%
          Next Sign6%
        Next Sign5%
      Next Sign4%
    Next Sign3%
  Next Sign2%
Next Sign1%
End
'
Procedure Sumterm(Sign%,Digit%)
  Sign%(Digit%)=Sign%
  If Sign%=0
    Sum%(Digit%)=Sum%(Digit%-1)
    Term%(Digit%)=10*Term%(Digit%-1)+Sgn(Term%(Digit%-1))*Digit%
  Else
    Sum%(Digit%)=Sum%(Digit%-1)+Term%(Digit%-1)
    Term%(Digit%)=Sign%*Digit%
  Endif
Return
'
Procedure Print_equation
  Local Digit%
  For Digit%=1 To 9
    If Sign%(Digit%)<>0
      If Sign%(Digit%)=1
        Print "+";
      Else
        Print "-";
      Endif
    Endif
    Print Digit%;
  Next Digit%
  Print "=";Sum%(9)+Term%(9)
Return
```

. . . Solved!

and a new challenge.



```
-1+2-3+4+5+6+78+9=100
+12-3-4+5-6+7+89=100
+123-4-5-6-7+8-9=100
+123-45-67+89=100
+123+4-5+67-89=100
+123+45-67+8-9=100
+12+3-4+5+67+8+9=100
+12+3+4+5-6-7+89=100
+1+23-4+56+7+8+9=100
+1+23-4+5+6+78-9=100
+1+2+3-4+5+6+78+9=100
+1+2+34-5+67-8+9=100
```

Total 100 program output.

the group, clocking in at 50 seconds for the uncompiled version and 20 seconds for the compiled version. The author's explanation of the program is as follows.

The program considers the sign of each number (1 through 9) to be either +1, -1, or nil (the latter case is redundant for the first digit). It loops through all 13,122 $(2*3^8)$ possible permutations of these signs. For each digit, a sum and a term are computed by the procedure Sumterm. The sum for a digit is the summation of all prior terms up to but not including the current digit.

If the sign of a digit is either +1 or -1, then the sum for the digit is obtained by adding the sum for the previous digit and the term from the previous digit. In this case, the term for the current digit is just the sign of the digit times its magnitude.

If the sign of the digit is zero, its sum is the same as the sum of the previous digit, and its term is ten times the previous term plus the magnitude of the current digit. By keeping track of these sums and terms, a large amount of repetitive calculation is avoided. [*You said it!— Ed.*]

TOTAL 100

ATARI KEY

- Any Atari ST Computer
- Pascal

Listing 2.

```
program make100 (output);

  var
    Solution: array[1..19] of char;
    LengthSolution: 0..19;

  procedure Search (part1, sign, part2, n: integer);

    var
     i: 1..19;

    begin

      If n < 10 then
        begin     (* Suppose current solution is 'XX' *)
          (* Try solution 'XX+n' *)
          LengthSolution := LengthSolution + 1;
          Solution [LengthSolution] := '+';
          LengthSolution := LengthSolution + 1;
          Solution [LengthSolution] := chr(48+n);
          Search (part1 + sign*part2, 1, n, n+1);

          (* Try solution 'XX-n' *)
          Solution [LengthSolution - 1] := '-';
          Search (part1 + sign*part2, -1, n, n+1);

          (* Try solution 'XXn' *)
          LengthSolution := LengthSolution - 1;
          Solution [LengthSolution] := chr(48+n);
          Search (part1, sign, 10*part2 + n, n+1);

          (* Backtrack to solution 'XX' *)
          LengthSolution := LengthSolution - 1;
        end
      else if part1 + sign*part2 = 100 then
      (* Current solution is successful so we print it *)
        begin
          for i := 1 to LengthSolution do write (Solution[i]);
          writeln (' = 100');
        end;
    end;

    begin (* main program *)
      (* Initialize solution to '1' *)
      Solution [1] := '1'; LengthSolution := 1;
      Search (0,1,1,2);
    end.
```

When the final sum is found to be 100, the procedure Print_equation is called to show the solution. The print step requires the signs of all the digits in the solution to be known. For this reason, they are stored in the S% array as part of the Sumterm procedure.

The program is easily adapted to look for a sum other than 100. Only the first line must be modified.

Our thanks and a tip of the hat—along with a one-year subscription—go to Joseph Wrobel.

The Pascal Solution

Our other winning entry, the Pascal program in Listing 2, is explained by its author, Pierre Beauchemin.

The idea of this program is to represent all possible calculations using +, -, and the digits 1 to 9 (every digit appearing once) by strings such as 1 # 2 # 3 # 4 # 5 # 6 # 7 @ # 8 # 9, where # can be +, -, or empty. We start with the string 1 and explore all the possible ways to complete the string: 1+2, 1-2, and 12. Then we go on exploring recursively, starting from each of these strings.

A string represents a possible solution only when it is complete, ending with a 9. It would be very inefficient to evaluate each string when it is completed. So as we build a string, we keep track of three values. The first is the partial sum up to the last operator symbol, the second is the value of the number currently being constructed, and the third is the sign (+1 or -1) between the first two values.

If the string is 1+23-4-56, for example, then the first value is 1+23-4=20, the second is 56, and the sign is -1.

If we add 7 to the string (the string then becomes 1+23-4-567), these values become 1+23-4=20, 567, and -1, but if we add +7 (the string becomes 1+23-4-56+7), they become 1+23-4-56=-36, 7, and +1.

The program is based on the recursive procedure Search, which receives the three values we just discussed and the value of the next character digit to be added to the string. The string containing the solution is a global variable (an array of characters).

Pierre notes that for the sake of portability he avoided the use of a predefined string type, which is implemented in some but not all Pascal dialects. A string variable would make the program smaller and faster; nevertheless, its execution time of three seconds on an ST is quite spectacular.

Honorable mention in our competition goes to Jerry Bridgeman of Madison, WI, for the fastest and most elegant solution for Atari 8-bit computers. Jerry's 41-line program finds all 12 solutions in a respectable 90 seconds thanks to three statements that screen out number combinations that can't possibly lead to the right total. Specifically, the program eliminates number combinations that sum to the wrong parity (for example, odd when the total sought must be even), sum to a total smaller than required, and are so much larger than the remaining digits that even if everything else were subtracted, the result would still be too large

We should also give mention to the very readable—if a bit lengthy—Pascal program submitted by Takyiu Liu of the University of Santa Barbara and to Hubert Roop for his Basic program which, at 25 lines, was the shortest submitted.

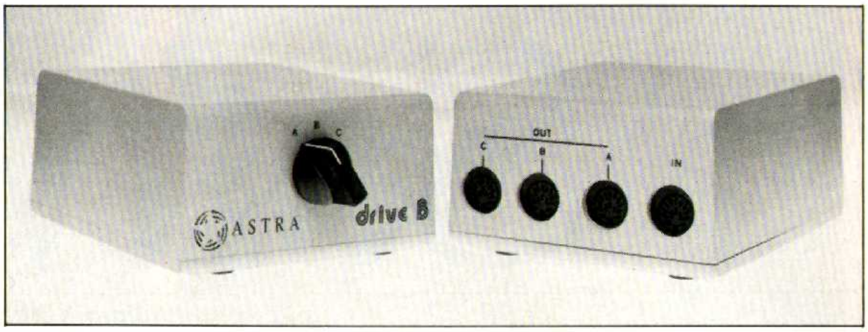
Thanks, too, to all the other readers who entered.

A New Problem

One of our winners, Pierre Bauchemin, has proposed the following problem for readers to solve.

Starting with the number 1, you can multiply by 2 and divide (with truncation) by 3. Using any computer language, find the shortest way to generate the numbers 2 to 25 or, for a more decisive challenge, 2 to 100. Your answers should be represented as in this example, 3=1*****// (means multiply by 2 five times to get 32, then 32/3=10 and 10/3=3).

Be sure to analyze your approach before you commit this to electrons and silicon. Send you answers to Multiply/Divide Problem, Atari Explorer, 7 Hilltop Rd., Mendham, NJ 07945. A one-year subscription goes to the best solution received before May 1, 1989. ■



Disk Drive Switches

Two convenient ways to eliminate the annoyance of drive swapping

Drive Master

System: Atari ST

Summary: Disk drive switch for two drives

Price: $49.95

Manufacturer:
Practical Solutions
1930 E. Grant Rd.
Tucson, AZ 85719
(602) 884-9612

Drive B

System: Atari ST

Summary: Disk drive switch for up to three drives

Price: $79.95

Manufacturer:
Astra Systems
2500-L S. Fairview
Santa Ana, CA 92704
(714) 549-2141

If you have added a 5¼″ disk drive to your ST system, you know how inconvenient it is to plug and unplug drives when you want to switch from the 5¼″ drive to the 3½″ drive or vice versa.

I used to try to minimize drive swaps by saving up my IBM PC tasks and doing them all at once, but it always seemed that I would need to copy a 3½″ disk right in the middle of my work session.

But now, cable switching is a thing of the past thanks to the recent introduction of drive switches by Astra Systems and Practical Solutions.

The Astra Systems device, known as Drive B, measures a largish 6″×5½″×3″ and connects up to three floppy disk drives to the drive B output on the computer. To switch drives, you merely turn a rotary knob on the front to the marking A, B, or C.

Hooking up the device is utter simplicity; you insert a cable (not included) between the drive B output jack on your 1040ST or Mega (or OUT port of Drive A on a 520ST) and the IN jack on the switch, and plug your external drives into the jacks marked A, B, and C. That's it.

Drive B sells for $79.95.

Drive Master

Drive Master from Practical Solutions is different from the Astra switch in several small ways. First, it switches two external drives instead of three. Second, it uses a push button switch rather than a rotary one.

Third, the device is smaller, measuring a compact 4¼″×3¼″×1¾″. And fourth, Drive Master comes with the cable that connects the computer and the switch. Hooking up is exactly the same as with the Astra switch. Drive Master is priced at $49.95.

Which one to buy? Both are well-constructed and, based on the reputations of the manufacturers, should be reliable. The Astra Drive B switches three drives with a rotary switch, which lets you see easily which drive is selected. But it costs more, particularly considering the added expense of the switch-to-computer cable.

Practical Solutions' Drive Master is smaller and includes a 3′ switch-to-computer cable, but limits you to two drives.

I think you can safely make your choice based on the current or projected complexity of your system. Both boxes offer value and convenience, and either one will be a welcome change from the plugging and unplugging routine we have had to live with for so long. ■

By DAVID H. AHL

Software Survey

Lots of new software packages to fill the shelves of your library

SkyChase

🟥🟥🟥🟥⬜⬜⬜⬜⬜⬜
EASE OF LEARNING

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
CHALLENGE

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
GRAPHICS

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
DOCUMENTATION

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
OVERALL RATING

System: Atari ST

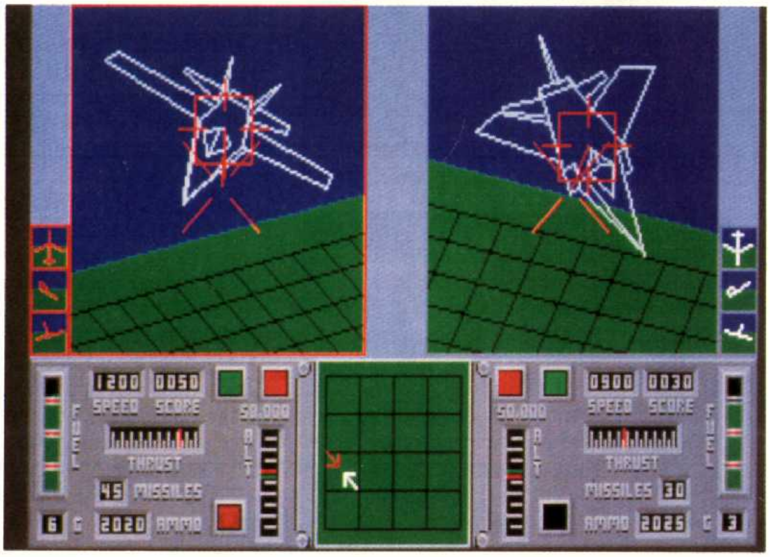
Required equipment: Color monitor, joystick(s)

Copy protection: None

Summary: Spectacular 3-D jet fighter simulation

Price: $39.95

Manufacturer:
Maxis Software
953 Mountain View Dr.
Lafayette, CA 94549
(415) 376-6434



There I was, cruising along in my Grumman F-14 Tomcat, when I suddenly got the feeling I wasn't alone in the sky. Sure enough . . . there was a bogy on the radar, inbound at bearing 55 degrees, coming on too fast, even for a MIG-21. I snapped off a high-G Yo-Yo, and went at him, head to head. As he whipped by my starboard wing, I caught a glimpse of his silhouette: variable geometry, no turbines . . . No question about it, I was being attacked by a paper airplane. For this, I joined the Navy?

If you have always dreamed of flying the heavy iron, you are going to love *SkyChase*. This new offering by Maxis is a 3-D, wire-frame, combat flight simulator that has been optimized to provide maximum performance and aerodynamic realism at the expense of visually complex terrain and a full set of controls.

SkyChase combat is carried out in an enormous cube of virtual space, wherein the ground is represented by a green grid, and the sky by a featureless (not to say long, delirious, and burning) expanse of blue. Play typically begins with a level flyby of your opponent, after which the object of the game is to make that opponent dead. This is very difficult.

Even though the controls of *SkyChase* are limited to joystick and throttle keys, and even though your plane has tools (location radar, ranging, "heads up" display, etc.) to help, finding an enemy aircraft in all that sky, maneuvering him into your sights, and blowing him away (while avoiding having the same done to you) is what might be called very, very challenging.

In the movies, they make jet combat look easy and aggressive, like a video game. Forget it . . . if *SkyChase* is a faithful simulation of reality (and I have reason to believe it is), the pilot's predominant feelings are of almost constant confusion ("Where the hell is he!?") and heart-pounding fear ("Missile Lock! He's got me in missile lock! I'm bugging out!").

You won't believe the graphics—smooth and phenomenally fast. Each pilot has his own cockpit window—one on the left, one on the right side of the screen—so you can enjoy the experience of combat from both perspectives. When you are playing against the computer, combat is too ruthlessly initiated to give you time to enjoy the view. But I'm looking forward to flying some pursuit formations with a cooperative and not-too-bloodthirsty human opponent, just so I can marvel at the way the planes move through the sky.

SkyChase supports a rich range of options for making combat more interesting. A wide variety of plane types—American, Russian, and the aforementioned Paper Airplane—plus variable fuel, ammunition, and missile payloads; missile-lock ratios; G-force tolerances (yes, you *can* black out); combat area; and computer skill make even the one-player game a continuing challenge.

The documentation, which someone named Karen Jacobs wrote in exceptionally clear and precise English (applause to her), covers the details of the program in acceptable depth and even includes a graphic tutorial on aerial combat strategy. This game is a winner! ***—Harmon Cove***

All Aboard

■■■■■■■■■□
EASE OF LEARNING

■■■■■■■■■□
GRAPHICS

■■■■■■□□□□
DOCUMENTATION

■■■■■■■■■■
PLAY VALUE

System: Atari ST

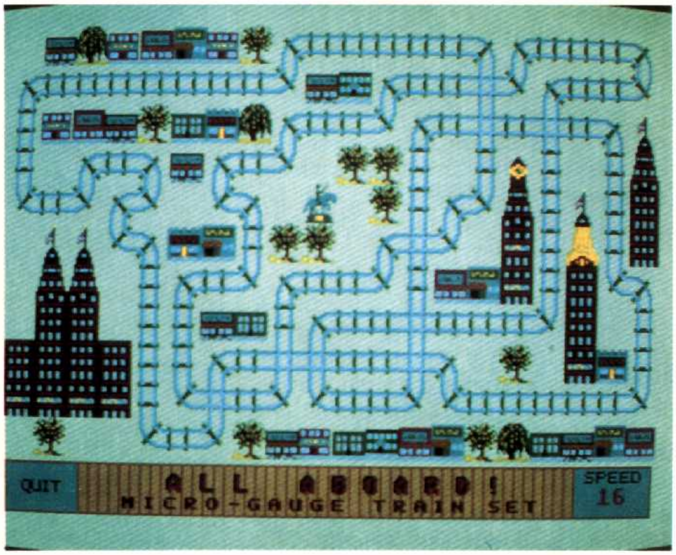
Required equipment: Color monitor; Degas, NeoChrome, or Cyber Paint optional

Copy protection: None

Summary: The ultimate train set for the Pugsley Addams in all of us

Price: $24.95

Distributor:
Terrific Software
544 Second St.
San Francisco, CA 94107
(415) 957-0886



All Aboard is the kind of program I wish I had written myself. Not because it's going to make its authors a zillion dollars (though, in the best of all possible worlds, it really should; they deserve it), but because it's going to put a smile on the face of everyone who boots it up. *All Aboard* is a train set implemented in software—tracks, cars, engines, switches, scenery, the works. Or, rather, it is a train set construction kit, which lets you build train sets, decorate them, and then play with them to your heart's content, unconstrained by lack of space, time, money, manual skill, or any of the other logistical considerations that make life difficult for the conventional hobbyist.

The main screen of the program is laid out in an implicit grid of 16×16 pixel units. Objects (track sections, scenery, etc.) snap to the grid when you move them around, making it easy for even small children to build working layouts. Parts for the layouts (as well as the railroad cars themselves) are taken from a menu that pops up when you move the mouse cursor down to the bottom of the screen—a very elegant set-up that lets you use almost the entire screen for track-building.

Five basic layouts, which can be brought up on the main screen by mouse-click from the menu, are provided with the program. These can be played with as is or modified and decorated in an unlimited variety of ways and saved to disk for future play.

Stoking up a train and getting it going is a matter of selecting the appropriate engines and cars from the parts menu and moving them to a horizontal section of track, where they start chugging along, automatically. Individual cars can run independently, or long trains of them can be linked together with or without engines. The only limitation to train size or configuration, in fact, is the length of horizontal track your layout makes available for train assembly.

It is even possible to run multiple trains on the same layout, though this naturally increases the probability of collision. (Fun! Remember Pugsley Addams? More on collisions later.) The speed of all cars on the layout is controlled by clicking on the mouse buttons, and current speed is indicated on a status line at the bottom of the screen.

You can remove individual cars from the track by clicking on a Clear Last button, but only in the reverse order to that in which they were installed. Alternatively, the track can be cleared of all cars, using Clear All.

The graphics are two-dimensional, and layouts employing the default scenery have something of the look of folk-art samplers. The tiny engines are animated in various simple but amusing ways, and sound effects add to the enjoyment of watching the trains go around.

When you get bored with just watching the trains go around, you can stage a collision or run them off the track in various ways, blowing up cars and track sections with abandon.

Enterprising engineers can even use *Degas*, *NeoChrome*, or a compatible paint program to design their own scenery, cars, and engines, then clip them out and use them in *All Aboard* layouts. Engine designs incorporate two frames, and the program will animate these just as it does its default engines.

Documentation for *All Aboard* comes in the form of a README.DOC file on disk, but is otherwise clear, concise, and easy to understand. Taken as a whole, the program can be considered a real bargain, offering infinite play value in a very small package. Great gift idea, too. ***—Harmon Cove***

Chrono Quest

🟥🟥🟥🟥🟥🟥🟥⬜⬜⬜
EASE OF LEARNING

🟥🟥🟥🟥🟥🟥🟥🟥🟥⬜
CHALLENGE

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
GRAPHICS

🟥🟥🟥🟥🟥🟥⬜⬜⬜⬜
DOCUMENTATION

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
OVERALL RATING

System: Atari ST

Required equipment: Color monitor

Copy protection: Passive. Backups can be made, but original packaging is required to play game

Summary: A uniquely absorbing, massive, and beautifully-executed graphic adventure

Price: $49.95

Distributor:
Computer Software Service
2150 Executive Dr.
Addison, IL 60101
(800) 669-4912

Has anyone else noticed that there seems to be a little French Revolution going on in Atari ST programming these days? And do people agree with me that, even though it almost hurts to admit it, there may be substance in the old stereotype of "French style"? Some of this software—particularly in the graphics area—is really pretty *formidable*.

Chrono Quest, written by Infomedia (France) and distributed in the U.S. by Psygnosis, is a case in point—a superbly-crafted, truly massive graphic adventure game with that certain *je ne sais quoi* and Premium Import written all over it.

The adventure begins in the year 1922, in your father's chateau in the Loire valley. Dad is dead—victim of his unscrupulous manservant, Richard—and you have inherited his money, his property, and his time machine. Now, what with a chateau on the Loire, piles of money, and at least a decade in which to move the property brick-by-brick to Carmel, CA, before the start of the Second World War, any intelligent person might figure he had already won the game before the adventure began.

Unfortunately, there are no commands for settling into the castle, finding a decent cook, and re-stocking the wine cellar. Instead, you are obliged to toddle right off and bring Dad's murderer, the aforementioned Richard, to justice. Thanks to the family time machine and Dad's typically French sense of humor, moreover, this task will take literal eons to complete, and you will spend most of this time dodging arrows in 8th-century Mexico, avoiding lepers in 17th-century India, and, of course, falling down stairs in the dark and breaking your neck.

And you'll love it! The game is played entirely by mouse. An exceptionally well-designed control panel gives access to a range of actions (pick up, put down, push/pull, turn, examine, wait, etc.) that can be performed on objects depicted in the current graphic or stored (iconically) in inventory.

The graphic window takes up approximately three-quarters of the screen, depicting your point of view on the current scene. The graphics themselves are colorful and beautifully drawn, and the game system is capacious enough to incorporate both pop-up magnifications and text descriptions of objects you wish to examine closely.

You won't find many meditative riddles to puzzle over in this game. But you will find lots of drawers to open and close, faucets to turn, baseboards to inspect, and supine lepers to turn over in your search for punch cards and grappling hooks—terrific for the manually-inclined and for lovers of Advent calendars and scavenger hunts.

Moreover, if you get bored with the quest, you can marvel at the technical superiority of the implementation. The game is marvelously responsive—controls respond brightly and scenario shifts occur quickly, even where disk access is required. Although the game is distributed on four disks, things are organized to minimize disk-shifting. And, thanks to the unique copy-protection scheme, the true aficionado can probably copy the game to a hard disk or even a RAMdisk for truly seamless play.

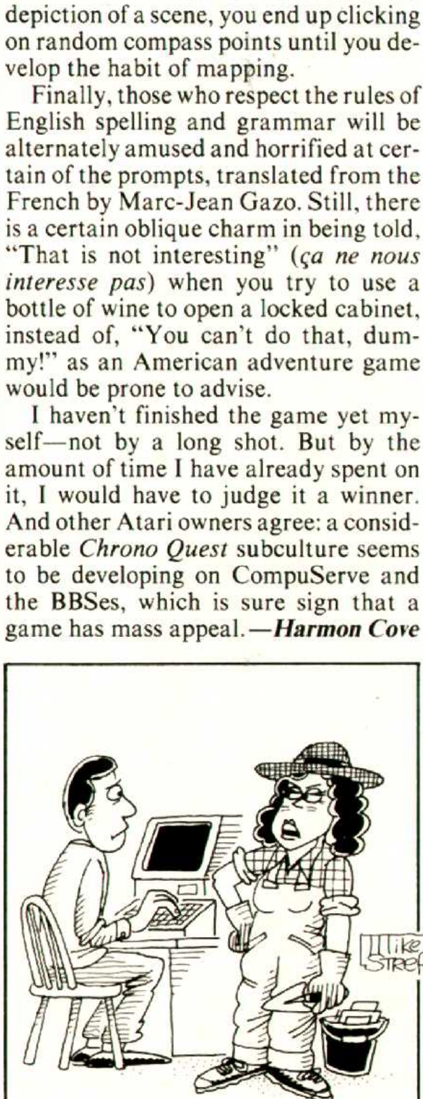
The copy-protection scheme, by the way, works by asking you to enter the predominant color at a particular point in the illustration on the *Chrono Quest* package cover, as referenced by markings on an enclosed transparent grid. Compliance is requested only during the boot process and is not the least bit annoying, provided you don't lose the grid or documentation before you complete the game.

I do have a few complaints, the first being that, although you are permitted to save your position in the game, there seems to be no way to name the file thus created. You are, therefore, forced to use multiple disks if you wish to save multiple positions.

Another minor annoyance is that because most scenes are presented from only one point of view (hardly surprising, considering the amount of disk space it would take to store multiple viewpoints), at least one of the available exits (the one you came through to get there) in any given scene is invisible. Since an exit list is not included in the depiction of a scene, you end up clicking on random compass points until you develop the habit of mapping.

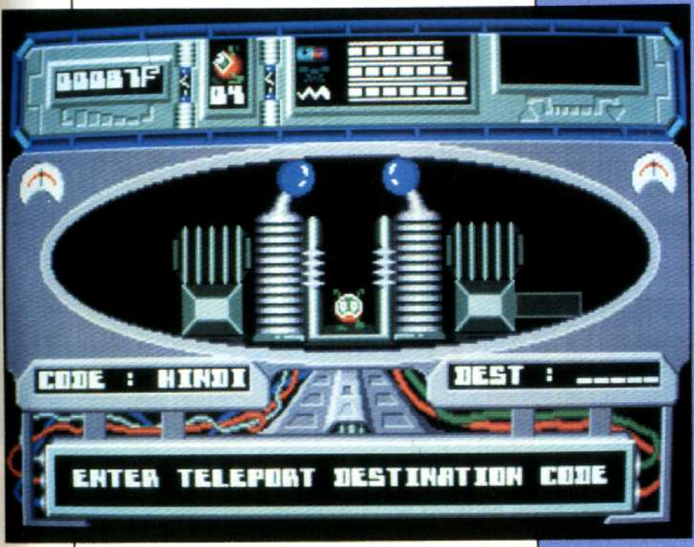
Finally, those who respect the rules of English spelling and grammar will be alternately amused and horrified at certain of the prompts, translated from the French by Marc-Jean Gazo. Still, there is a certain oblique charm in being told, "That is not interesting" (*ça ne nous interesse pas*) when you try to use a bottle of wine to open a locked cabinet, instead of, "You can't do that, dummy!" as an American adventure game would be prone to advise.

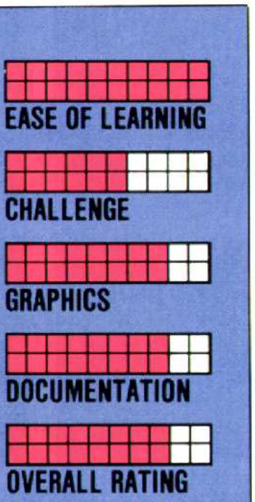
I haven't finished the game yet myself—not by a long shot. But by the amount of time I have already spent on it, I would have to judge it a winner. And other Atari owners agree: a considerable *Chrono Quest* subculture seems to be developing on CompuServe and the BBSes, which is sure sign that a game has mass appeal.—***Harmon Cove***



"I don't need a garden plan or a planting schedule or a seed inventory. I need help weeding."

Star Quake





System: Atari ST

Required equipment: Color monitor, joystick

Copy protection: None

Summary: Arcadesville, man!

Price: $29.95

Distributor:
Antic Software
544 Second St.
San Francisco, CA 94107
(415) 957-0886

One of these days, I'm going to write a book (or a PhD thesis) on video game iconography. Ever notice that there are just some programs that make you want to look for the coin slot in the top of your monitor? *Star Quake*, by Bubble Bus for Terrific Software out of Antic Software (let the IRS worry about it . . .), is that kind of program—a game that starts you hunting in your pocket for spare change. It just looks *so* arcady!

The scenario is this: you have crash-landed (lucky you) on the surface of an unstable synthetic planet and have the unenviable task of finding and re-installing nine elements of the planet's core before it blows up.

Sometimes I wonder why trivial games require background scenarios of such Wagnerian scale, or conversely, why game designers represent such literally earth-shattering themes in such trivial ways, but hey, I guess games are a metaphor for life itself. Anyway, your surrogate on this quest is a BLOB, short for BioLogically Operated Being—a little, round, red guy with green feet, goggles, and a pair of antennae that shoot lightning bolts—a fairly Promethean touch, at that.

Using the joystick, you have to navigate the BLOB about the maze-like interior of the planet (512 screens worth . . . you *have* to make a map), avoiding (or blasting) a multitude of other colorful little creatures before they waste you, picking up core elements and other useful objects, flying around on hover pads, teleporting from place to place, and what-all.

The trick (or so the documentation indicates—I have not conquered this game, myself) is to start by finding your way to the core of the planet, where you can determine which objects are needed to save the planet, then navigate back out again to find the objects and return to install them. The objects change with every play, so once you have successfully completed your mission, you can start right in again and try to improve your time!

What with enemy creatures, traps, and 512 rooms to navigate, your chances of dying from misadventure or energy starvation (not to mention ennui) are fairly high. So high, in fact, that you will probably wish for a save game feature; unfortunately, none is supplied.

Other than that, well, if you like fast-moving arcade action, don't mind being a BLOB, and want to save the world, you couldn't ask for more. The graphics are really cute, the music is poppy, and the joystick, as always, is warm.

—*Harmon Cove*

Wanted: Video Game Players

The editors of *The Atarian*, Atari's new magazine for players of Atari video games, are seeking to purchase well-written playing strategy articles and game playing tips.

Articles and tips must be about a game for one of the three Atari systems (2600, 7800, or XE game system)—no Nintendo or Sega please.

Playing Strategy Article

Length: 200 to 700 words (1 to 3 double-spaced pages).

Start with a brief description of the game and the objective of the player (rescue Melba, destroy Zylons, score points, etc.). Then get into the playing strategy. You can describe it screen by screen (how to elude the baddies or negotiate the mazes, which objects to pick up and which to avoid, etc.) or you can describe a winning plan of attack (for example, the fighting technique to use on goblins, how to position your spaceship in the asteroid belt, or which vegetables to pick up first). Be sure to describe ways to overcome particularly frustrating situations.

If your strategy article needs screen photos (most will), on a separate piece of paper, either describe or sketch the screens needed and indicate the position of the critical objects.

Playing Tips

A single sentence to a paragraph in length, a playing tip is what its name implies, a way to overcome an obstacle, escape from a tricky situation, or defeat a nasty character. Be sure to include a sketch of the screen with the crucial objects marked.

All submissions must be typed (or printed on a word processor) double-spaced. They should be sent to Strategy Article, *The Atarian*, 7 Hilltop Rd., Mendham, NJ 07945. Be sure to include your name and address on all submissions and enclose a self-addressed, stamped envelope for our reply. ■

Offshore Warrior

🟥🟥🟥🟥🟥🟥🟥⬜⬜⬜	EASE OF LEARNING
🟥🟥🟥🟥🟥🟥🟥⬜⬜⬜	CHALLENGE
🟥🟥🟥🟥🟥🟥🟥🟥🟥⬜	GRAPHICS
🟥🟥🟥🟥⬜⬜⬜⬜⬜⬜	DOCUMENTATION
🟥🟥🟥🟥🟥🟥🟥⬜⬜⬜	OVERALL RATING

System: Atari ST

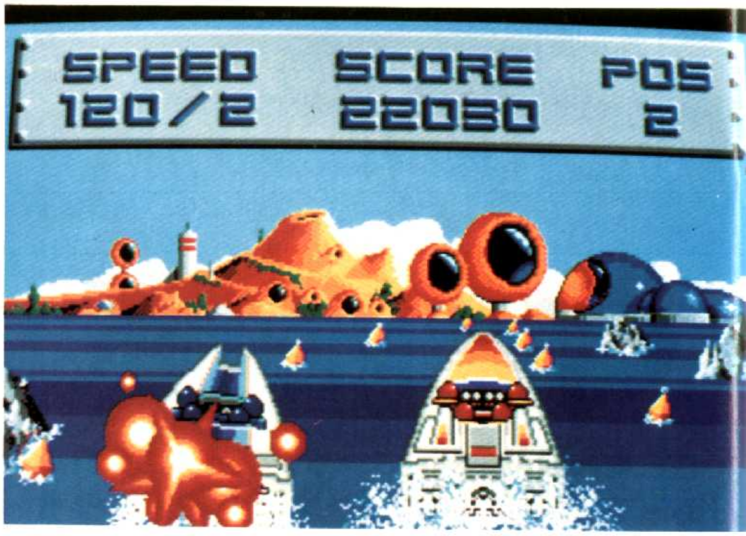
Required equipment: Color monitor; joystick optional

Copy protection: Yes

Summary: Futuristic speed boat race with missiles

Price: $39.95

Manufacturer:
Titus Software
20432 Corisco St.
Chatsworth, CA 91311
(818) 709-3693



The age-old theme of gladiatorial combat is brought to the home computer once again in *Offshore Warrior* by Titus Software. The game is set in the year 2049, several years after an alien race of pacifists has invaded the earth and made war a thing of the past.

Unfortunately for the people of the planet, everything exciting and action-packed has been eliminated. To fill this entertainment gap, the Offshore Warrior movement has arisen, and several large lakes have been set aside for use as arenas of combat. A course, marked by a series of buoys, is laid out on each lake, and bleachers have been placed on shore to accommodate the throngs who want to view the contest.

The object of the contest is to be the first to cross the finish line. Sounds simple, right? But if that were all there were to it, it wouldn't be much of a game. So to make things more interesting, each boat is armed with two missiles with which it can shoot pesky opponents. Add to this the hazard provided by rocks scattered about the course, and you have a challenging race.

The Offshore Warrior annual racing circuit features four different lakes to

Racing game fans have a new treat in store when they boot up one of the latest releases from U.S. Gold. In *Metro Cross*, you must race your way through 24 different sections of a city subway system. Each leg must be completed in a specified amount of time or you are finished, literally. To increase your incentive to complete the races as quickly as possible, bonus time equal to the amount remaining from the previous three races is added after every fourth race.

Many obstacles hinder your progress through the subway system. Slimy tiles cause you to slip and slow down, and rolling barrels, tires, and red cubes attempt to knock you over. In addition, you must leap over hurdles and avoid potholes and air vents along the way. Fat rats also wait for a chance to slow you down by sinking their teeth into your ankles. Finally, tricky walls occasionally rise up to block your path.

Fortunately, everything you encounter is not harmful. You will also find some useful items scattered through the subway system. For example, countdown time is suspended each time you crush one of the many cans that are laying about. And certain of these cans cause an increased adrenaline flow, speeding you toward your goal.

Springboards can be used to propel you through the air like a guided missile. Skateboards are probably the most valuable aid in the game. With these, you can skim right over slime tiles and make great progress toward the goal.

Metro Cross

🟥🟥🟥🟥🟥🟥🟥🟥⬜⬜	EASE OF LEARNING
🟥🟥🟥🟥🟥🟥🟥🟥🟥⬜	CHALLENGE
🟥🟥🟥🟥🟥🟥🟥🟥🟥⬜	GRAPHICS
🟥🟥🟥🟥🟥🟥🟥⬜⬜⬜	DOCUMENTATION
🟥🟥🟥🟥🟥🟥🟥🟥🟥⬜	OVERALL RATING

System: Atari ST

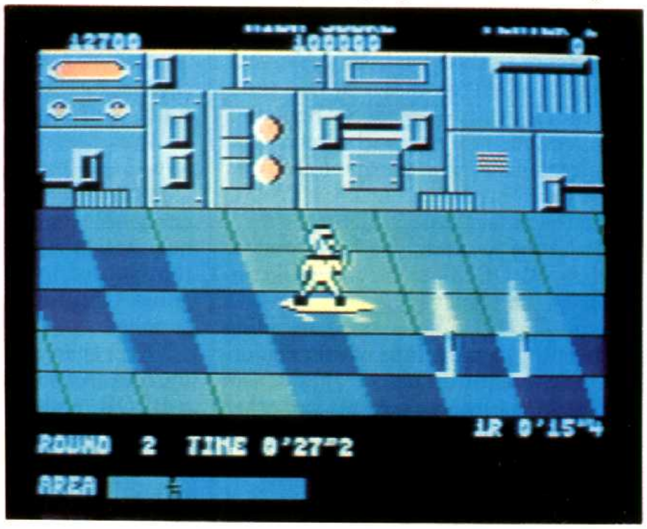
Required equipment: Color monitor, joystick

Copy protection: Yes

Summary: An imaginative and challenging race over an unusual course

Price: $24.95

Distributor:
Epyx
P.O. Box 8020
Redwood City, CA 94063
(415) 368-3200



race on, with each successive race providing more opponents and more missiles to shoot at them. Your boat must finish in either first or second place to be eligible to compete in the next race.

The audio background to the title screen is done with very high quality digitized sound—your first to clue to the excellent quality of the game. The roar of the engines and the boom of the explosions add to the feel of racing a high powered speed boat. The colorful and detailed graphics also serve to draw you into the game.

The game has only two drawbacks. The first is that although the races are challenging, after a while they can seem repetitive and less exciting than they did at first.

Also, firing the missiles accurately can be a daunting task. It is very hard to be sure that your shots are aimed properly, because there is no aiming device. In fact, it is quite possible to miss an opponent dead ahead, because your shot behaves unpredictably.

Despite these minor, shortcomings *Offshore Warrior* should offer many hours of enjoyment to racing game fans. ***—Brad Andrews***

However this speed has a drawback. If you do not know what to expect ahead, you will often be surprised by a hurdle appearing suddenly in front of you. With no time to dodge, you will be thrown from the skateboard and forced to continue the race on foot.

During the race, your secondary goal is to amass as many points as possible. Your score is based on how far along the course you get and the bonus points you earn by kicking cans and other actions. Special point awards are given for successfully kicking the same can several times in a row—a challenging task indeed. Finishing a segment while on a skateboard provides a very nice score increase.

The graphics are very crisp and clear, and the background music adds excitement to each race. Unfortunately game play does slow when too many items are moving on the screen at the same time. But this does not occur often and passes quickly.

U.S Gold has definitely found the proper balance between playability and challenge with *Metro Cross*. It is a game you will often find yourself playing "just one more time" as you strive to better your best score.***—Brad Andrews***

Turbo ST

Turbo ST is a desk accessory that replaces certain of the GEM text and graphics functions with versions that execute more rapidly, letting programs that employ these functions execute at greater speed. It can't improve the performance of programs like *WordPerfect*, which don't employ GEM in screen-handling. Nor can it speed up programs such as spreadsheets, which spend a great deal of time doing calculations. Even word processors show little improvement in speed under *Turbo ST*, at least during most operations.

Turbo ST is not, therefore, a miraculous, across-the-board speed enhancement. Nor is it equivalent to a high-speed graphic processor such as Atari's Blitter. However, if you are a programmer, or if you are a writer, or if you simply wish to page through screen documents very quickly, you should purchase *Turbo ST*.

Full screen text editors, like MicroEMACS, flip through pages three times faster under *Turbo ST*. Desktop windows open and close quicker. Most word processors flip pages faster and demonstrate minor improvements in scrolling speed. Programs written in ST Basic also demonstrate a noticeable acceleration in screen output; a basic listing executes nearly twice as fast when *Turbo ST* is installed (see Figure 1). Also, when using the desktop Show function to scan a document, *Turbo ST* doubles screen display speed.

Installing *Turbo ST* is simple. You just copy TURBOST.ACC from the distribution disk to your boot disk, then boot as usual. In the great majority of cases, the program will function transparently—you will notice its presence only when the speed improvement takes hold. In the few cases in which compatibility problems exist with software, the *Turbo ST* desk accessory lets you turn off the enhancement functions, returning control of graphics to the original GEM code.

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
EASE OF USE

🟥🟥🟥🟥🟥🟥🟥□□□
PERFORMANCE

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
ERROR HANDLING

🟥🟥🟥🟥🟥🟥🟥🟥□□
DOCUMENTATION

🟥🟥🟥🟥🟥🟥🟥🟥🟥□
OVERALL RATING

System: Atari ST

Version reviewed: 1.2

Copy protection: None

Summary: Auto-executing desktop accessory that enhances screen updating for almost all text-oriented programs

Price: $49.95; inexpensive upgrade available to current owners of v. 1.0

Manufacturer:
Softrek
P.O. Box 5257
Winter Park, FL 32793
(407) 657-4611

The *Turbo ST* manual is clear and concise. The first two pages explain all that is necessary to install the program. (I wish all programs could be so easily documented). The remaining ten pages explain how and why *Turbo ST* works and why some programs are not affected by *Turbo ST*.

If you are a programmer, you definitely should have *Turbo ST*. For the end-user, *Turbo ST*, although not likely to demonstrate a great improvement in any one program, will accumulate time savings that will quickly repay the cost of the program. ***—Harmon Cove***

Function	Time in Minutes	
	with Turbo ST	without Turbo ST
Listing a 28-page ST Basic program to the screen	5:30	10:00
Scanning a 70-screen document using the Show function	0:55	1:55

Figure 1. Turbo ST comparative performance data.

Sinbad and the Throne of the Falcon

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
EASE OF LEARNING

🟥🟥🟥🟥🟥🟥🟥🟥🟥□
CHALLENGE

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
GRAPHICS

🟥🟥🟥🟥🟥🟥🟥🟥🟥□
DOCUMENTATION

🟥🟥🟥🟥🟥🟥🟥🟥🟥□
OVERALL RATING

System: Atari ST

Required equipment: Color monitor, joystick; 1Mb RAM recommended

Copy protection: Yes

Summary: An engrossing mixture of arcade action and adventure. Good graphics and music.

Price: $49.95

Manufacturer:
Master Designer Software
5743 Corsa Ave.
Westlake Village, CA 91361
(805) 495-6515

Sinbad and the Throne of the Falcon is another in the series of interactive movie-games from Cinemaware. You play the part of Sinbad the sailor in this, his latest adventure. The Caliph of Damaron has been turned into a falcon by a well-placed spell. Your goal is to find a way to save the Caliph from his feathery fate and protect the city of Damaron, Prince Harun, the rightful heir to the throne, and the lovely Princess Sylphani, who summoned you to help.

The black prince Camaral, one of the Caliph's sons, is trying to take the throne by force. He and other would-be usurpers of the throne are formidable enemies, but they have not reckoned with the strength and courage of Sinbad.

At the start of the game Sinbad stands in front of the city of Damaron. You can choose to explore on foot or set sail. You have a sword-shaped cursor which you use to select your actions and to choose dialogue for Sinbad when he converses with other characters.

At your belt are a green jewel (the Theraniil stone) and a map. Clicking on the jewel provides a close-up view, which reveals an hourglass, which shows the time left in the game, and a soldier. Click on the soldier and a battlefield map appears. Your map shows the area around Damaron, divided into hexes.

While you are traveling and having adventures, the city may be attacked by the black prince's armies, who will try to take it by storm. You must defend Damaron to the last man with the Caliph's loyal armies. Your orders are carried out as the game progresses.

Flashing hexes are supply centers. They allow you to reinforce your depleted armies during the game. The black prince's armies can also occupy these hexes and be reinforced. Obviously, you

Because I didn't like its cover art (amateur airbrush) and found its scenario laughable, devoid of imagination, and badly-written ("The six, legendary Phoenix tablets have been stolen by the evil Tetroids . . . ""), I left *Tetra Quest* at the bottom of my review pile for several days.

Finally, having finished all my other

Tetra Quest

🟥🟥🟥🟥🟥🟥🟥□□□
EASE OF LEARNING

🟥🟥🟥🟥🟥🟥🟥🟥□□
CHALLENGE

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥
GRAPHICS

🟥🟥🟥🟥□□□□□□
DOCUMENTATION

🟥🟥🟥🟥🟥🟥🟥🟥🟥□
OVERALL RATING

System: Atari ST

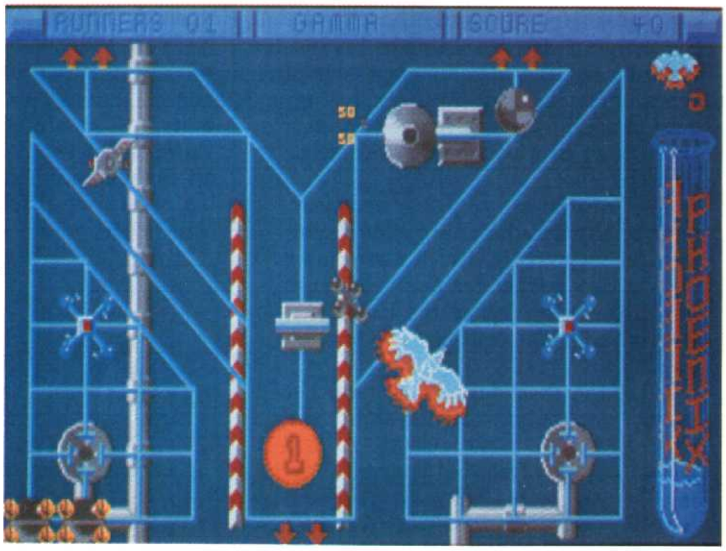
Required equipment: Color monitor, joystick

Copy protection: None

Summary: Super arcade game

Price: $39.95

Distributor:
Microdeal
576 S. Telegraph
Pontiac, MI 48053
(313) 334-8729



should try to keep this from happening. If you defeat the black prince's armies, his threat to the throne is ended. If you forget to keep an eye on the city, the black prince may take the throne by force, and you will lose the game.

Sinbad has a large world, including several continents each with a number of cities to explore. The seas you sail are often perilous, and you will come across many wrecked ships. You can try to pick up surviving crew members and add them to your own crew, but you must be careful not to endanger your own ship.

As you search for a cure to the spell, you meet many strange characters. Three that I have met so far are Libitina, the Gypsy, and the Genie. Libitina usually appears near the beginning of the game. She can be treacherous to deal with, but she can give you extra strength that you may need later in the game.

The Gypsy can provide important information, especially as you get further into the game, so be careful how you talk to her. The Genie can grant you a number of wishes, such as providing more men for your crew, but don't be too greedy.

All the characters in the game are beautifully detailed and animated. The conversations I had with them felt real and added to my feeling of involvement with the story in the same way that good characterization helps the reader get involved in a book.

It is a good idea to explore every city and every continent, so you don't miss some valuable item or important character you will need to finish the game. When you explore, a list of nearby cities is given. If you are at sea, you are given a choice of continents as your destination. The Theraniil stone and a magnifying glass for the map are before you in the ship's cabin.

When you reach your destination, you can take some of your crewmen with you to help you explore and deal with the monsters and other perils you encounter. You battle the monsters, which include a cyclops who carries off members of your crew, a flying demonbird (the Pteranoxos), and a swordsman with six arms, in satisfying arcade sequences. You must hit the cyclops in the eye with a rock and sling, shoot the Pteranoxos with an arrow, and cut down the swordsman.

The black prince himself attacks you from time to time, and an occasional earthquake makes things difficult for the adventurers. The exploration aspect of the game is reminiscent of *Jumpman*, as you jump from ledge to ledge and scramble up and down ropes.

A typical game of *Sinbad and the Throne of the Falcon* takes an hour or two to play. You set sail on your adventure and keep exploring until you win or die. A vertical bar at the left side of the screen shows your health. If it disappears, you die and the game is over. There is no save game option; you must start over from the beginning each time you die.

The graphics in *Sinbad and the Throne of the Falcon* are sharp and pleasing to the eye, giving credence to the Cinemaware claim of movie-like presentation. The animation in the arcade sequences is simple but effective, and the background music sets the mood (Arabian) and changes from scene to scene when appropriate.

Sinbad and the Throne of the Falcon comes on three single-sided disks. Owners of 520 STs will find themselves swapping and flipping disks frequently (especially when Sinbad talks to characters in the story). If you have a megabyte or more of memory, however, most of the game can be loaded in the beginning.

The manual includes a handy map of the world. If you consult it rather than the map on Sinbad's belt, you save one disk flip. The manual is clearly written, helpful, and entertaining.

Playing *Sinbad and the Throne of the Falcon* is engrossing fun. I especially enjoy the theatrical flare of the game and the way it pulls you into the story. It requires a good mix of careful exploration and quick reflexes and can't fail to please computerists with an ounce of adventurous blood in their veins.

—John S. Manor

work, I broke the shrink-wrap on *Tetra Quest*, slotted the disk, and brought the game up. Figure a half-hour of obligatory play time, and . . . The boot-cycle music brought my head around. Absolute dynamite. Danceable, no less! I pressed the joystick button, and . . . well, after that, I don't remember too much. The game started, as I recall, and then . . . suddenly it was morning.

Tetra Quest is nothing short of enthralling. Your character, the Tetra Runner, moves along a jangled geometry of metallic tracks, searching each four-screen quadrant for a like number of coins, which must be retrieved in sequence. Discontinuities in the track system force you to develop and implement a strategy (using teleport pads and switchable "gates") for getting to the coins in proper order.

Nor is coin retrieval your only problem; you must also fight your way through a veritable menagerie of enemy aliens at every level. And, as each alien dies, it creates a new type of discontinuity in the track system. Certain aliens leave oil slicks, which make it difficult or impossible to stop at certain points on the track. Others leave glue spots, which slow you down. Still others emit acid, which burns holes in the tracks and makes whole sections inaccessible.

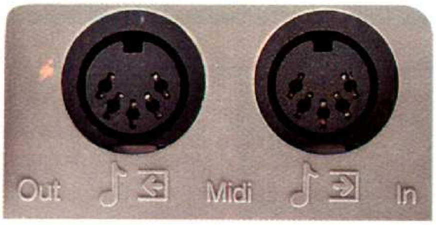
The upshot is that you are faced with the problem of developing a strategy for getting where you want to go in an environment where certain laws are fixed and others keep changing—rather like life itself.

Unlike life (but quite in line with the classical mythology/drama that has infected the scenario—the Tetra Runner is an avatar of Phoebus, the Sun-God), there is a *Deus ex Machina* solution you can employ if things get out of hand. By holding down the joystick button and pressing 0 on the keypad, you can turn into the invincible Phoenix for a time, flying free from section to section of the grid. Of course, you can only do this if you have a Phoenix life left—you have a finite number of these—and if you have enough Phoenix power. Luckily, you can buy and sell Phoenix power in exchange for points (which you gain by shooting aliens), which adds a bizarre economic aspect to the game.

The graphics are terrific, too. Metallic and transparent, jewel-like effects give a lovely, 3-D feeling to the track system, obstacles, and characters. The game is massive; 96 quadrants—384 screens in all—add up to quite a lot of challenge. Far too much, challenge, in fact, to get through in a single sitting. Luckily, the designers have devised a password system that permits you to reenter the game where you left off.

Like *Airball*, also from Microdeal/MichTron, *Tetra Quest* provides a challenge both to the reflexes and to the mind. One hopes that, also like *Airball*, *Tetra Quest* is designed in such a way that a "quadrant construction set" can eventually be made available.

—Harmon Cove



THE DIFFERENCE BETWEEN A COMPUTER THAT MAKES MUSIC, AND ONE THAT MAKES TROUBLE.

Of all the personal computers you can buy to make music, none makes it easier than Atari computers.

That's because, unlike the others, the Atari 512-kilobyte 520ST™, 1-megabyte 1040ST™, and 2- and 4-megabyte MEGA™ computers have more of what you need already built-in.

Here's what we mean.

MIDI. The Key to Electronic Music.

As you may already know, the MIDI interface is the key to electronic music.

If you're unfortunate enough to not be working with an Atari, you'll have to buy an interface separately.

And make sure it's compatible with the rest of your equipment, not to mention your software.

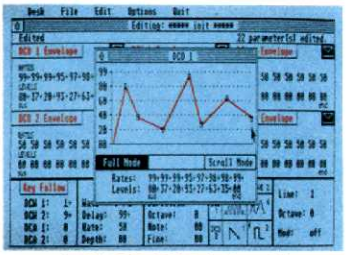
And then you'll have to make sure everything is installed correctly.

What's that like?

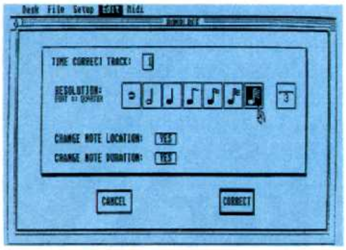
You know the song, "What are you doing for the rest of your life?"

Atari ST™ and MEGA computers, on the other hand, have a MIDI port built right into the back of the computer.

So you can connect all kinds of equipment—synthesizers, samplers, drum machines, SMPTE controllers, pitch-to-MIDI converters—as easily as plugging into an amp.

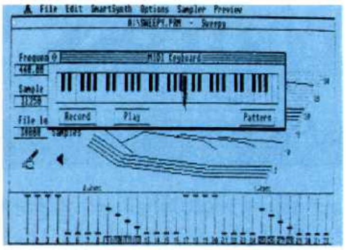


CZ-ANDROID™ Hybrid Arts™



TIME CORRECT TRACK:
RESOLUTION:
CHANGE NOTE LOCATION: YES
CHANGE NOTE DURATION: YES

MIDISOFT STUDIO™ Passport™



SOFTSYNTH™ Digidesign

THE COPYIST™ Dr. T's™ Music Software

A Musician's Music Box.

No other computer company has made the commitment to music and musicians the way Atari has.

That commitment, by the way, doesn't end with our hardware.

We're working in harmony with all the major music software houses to produce the software you want to make music with.

And building a distribution network of music dealers—not computer dealers—who know electronic music well enough to help you, no matter how much you know.

This Should be Music to Your Ears.

The Atari ST and MEGA computers are just parts of a full system. So there are lots of things you can add when you're ready.

Like our MEGA File 20™ 20-megabyte hard disk for storing your magnum opus.

And our SLM804™ laser printer for publishing it.

Plus one of the largest libraries of music software in the industry.

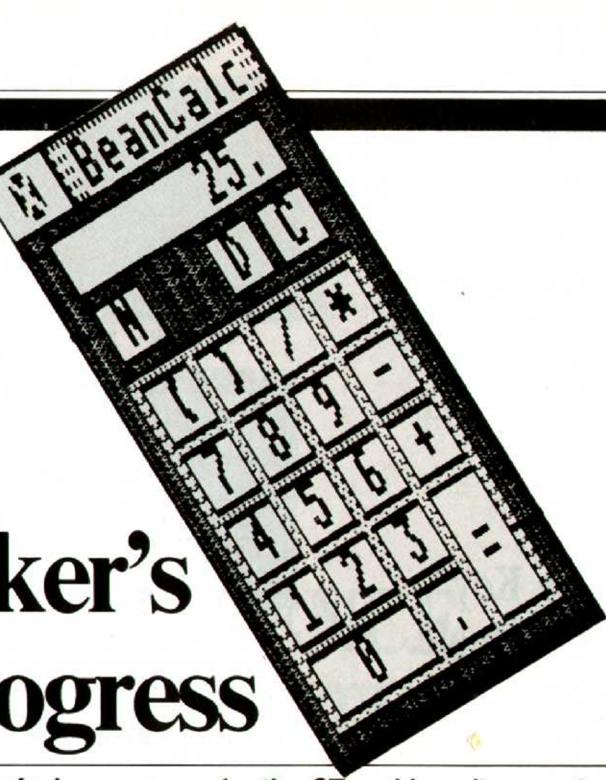
But perhaps the nicest thing about an Atari is how little it costs.

With what you save on an Atari, you could buy yourself a synthesizer. And some software.

Want to learn more? Write or call for the name of the Atari music dealer nearest you.

ATARI®

Atari Corporation • Attention: Music • 1196 Borregas Avenue • Sunnyvale, California 94086 • Tel: (408) 745-2367



Hacker's Progress

A calculator desk accessory for the ST and how it came to be

A few years ago—back when everybody thought computers were going to change the world—I worked as head of "type-in" programming for a big home computer magazine. Night after night, long after all the normal people had gone home, I would be in the lab, hacking away at a dozen or so different brands of microcomputer, drinking gallons of coffee and smoking pack after pack of Gitanes.

Those were the days, I tell ya. The programs I wrote—most of them in Basic, some in altogether weird varieties of assembler . . . hey, they were *classics*, man! You mean to tell me you never saw a copy of "Christmas Tree Construction Set"? What about "Bugs," a simulation that drove painstakingly realistic, life-sized roaches around the screen? Never, huh? Why, where were you?

Anyway, times have changed. Lately, all the programming I have done has been for tutorial demos, and I have begun to worry that too many early nights, insufficient tobacco, and excess of female company are taking away my edge—turning me into the kind of pseudo-technical bean counter that gets off on benchmark results the way an audiophile gets off on stereo test records.

So, three days ago, in a last-ditch effort to forestall premature senility and rediscover my sociopathic side (not to mention generate a few pages for this issue of *Explorer*), I decided to throw myself into a programming project and force it ruthlessly to completion. To push headlong through the forest of algorithms and machine details, ignoring the protests of a mind grown weak with application to the trivialities of normal (i.e., non-programming) existence. To revel in the cruel, masculine ethos of the "sport death" lifestyle. In other words, to smoke a lot, stay up late, ignore my girlfriend, and hack up something nifty for the magazine.

From Concept to Design

Times being what they are, of course, I couldn't be too self-indulgent. The program I built had to have some practical value. It had to be something people would use—at least to get their heads around a programming concept or two.

Now it so happened that just as I was beginning to cast around for an idea, I was asked to review a new product called *Revolver*—a perfectly wonderful program that divides your ST into multiple, virtual machines (see review elsewhere in this issue). I liked the program so much that I made it part of my permanent system installation, with the result that instead of one 1040ST I now have three: one with a half-meg of RAM for production work, and two quarter-meg runabouts for special projects.

I swiftly discovered that a 256K virtual ST doesn't carry enough RAM to support both an application and the 47K scientific calculator desk accessory I normally use. Moreover, I was getting a little tired of that .ACC, just on aesthetic grounds. It is as big as a barn door, for one thing—much too big to

By JOHN JAINSCHIGG

put leave at the side of a word processing document. And ugly! Looks like it draws itself using raw VDI calls. And does it have *features!* Binary, hex, memory, roots, logs, and I don't know what-all.

What I really wanted, I realized, was something cuter, smaller, and less powerful. I wanted a calculator desk accessory that was the equivalent of one of those credit-card-sized premium jobs they give away with subscriptions to *Time* magazine. Four functions, a little color, and a nice, comfortable mouse-or-keypad user interface. So why not write it?

Technically, the project was appealing for a number of reasons. First, there was the question of optimization; I wanted to make the accessory as small as possible, both logically and physically. Then, there was the challenge of figuring out exactly how a calculator works and simulating it to a nicety—something I felt intuitively could best be handled using recursion (magic word!).

Finally, there was the fact that such a project would give me an excuse to dive headlong into the theory of floating point mathematics (something the curriculum in Comparative Literature at Columbia University had somehow failed to cover in adequate depth when I was there as an undergraduate). Was I psyched?

***I wanted a calculator** desk accessory that was the equivalent of one of those credit-card-sized premium jobs they give away with subscriptions to Time magazine.*

The Resource

Cackling wildly, I made for the Resource Construction Set. I don't know about you, but whenever I do a GEM program, I like to get a handle on the resources first—like eating dessert before dinner. Furthermore, my design constraints on this project were pretty stringent: the calculator had to be really small and cute as a bug, yet large enough to handle comfortably with the mouse. Since I planned to wire in the numeric keypad of the ST as an alternative input device, the keypad of the calculator ought to be modeled after it.

Neither, at this point, did I have the least idea what kind of functions I ought to include. Add, subtract, multiply, and divide, of course. And I'll need a Negate key, so that people can enter negative numbers. And a Clear All key so they can clear the machine. What about a Clear Display key to get rid of a bad entry without screwing up a calculation in progress? Sure, why not?

And what about parentheses? Hmmm . . . there they were on the keypad, big as life. Okay, deal—parentheses it is. Exponent entry? Heck, yeah! What about a "financial adding machine mode" key that would force results to be rounded to the nearest cent? Why, suuuuuure!

Strange, the way features seem to multiply when adding one is simply a matter of labeling a new button, eh? Later on, of course, when the time came to write code in support of all these features, I had occasion to reconsider some of my early design decisions. The result: kiss those exponents goodbye, and if you want adding machine functionality, this calculator is not for you.

Building the resource—essentially a complex dialog box—was fairly simple. Because I knew the calculator would be contained in a window, I used a borderless accessory G_BOXrectangle. Another G_BOX, this one with a border, served to outline the main keypad.

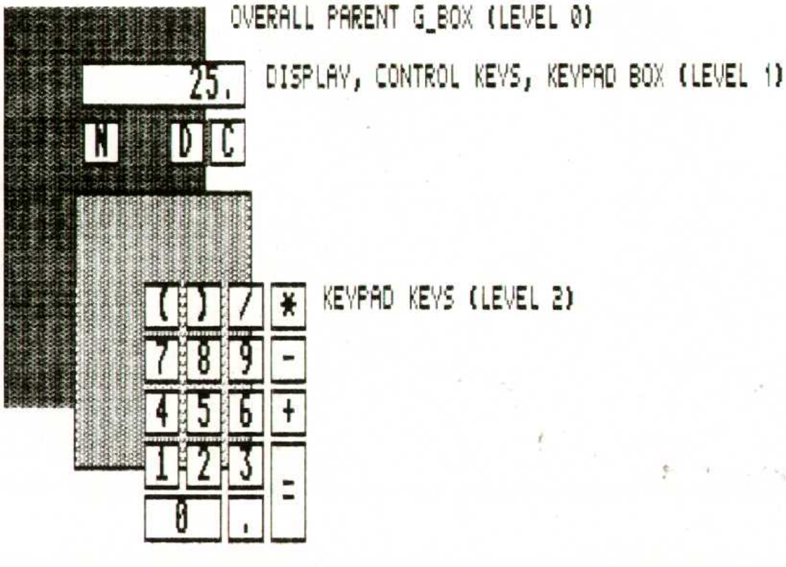
The display went up above, along with the Clear All, Clear Display, and Negate keys. Inside the keypad box went the digit keys, parentheses, and operator keys, in as close an approximation of the geometry of the ST keypad as I could manage. Except for the two boxes, I used G_BUTTON objects for everything. Since boxes and buttons can be represented with OBJECT structures alone (i.e., no TEDINFO or other special secondary structures are required to describe these object types), the resource would be economical of memory and easy to handle when converted to C language format.

Though I realized that I would have to manage the resource at a fairly low level (i.e., using evnt_multi( ), objc_find( ), and objc_draw( )) when it was finally bound into a working desk accessory, I was still careful to set the flags for each object (SELECTABLE, TOUCHEXIT, etc.) appropriately so that I could run the resource under form_do( ). That way, I could build and debug the calculator as a standard GEM application before binding it into the harder-to-debug accessory framework.

Objects at each level of the final resource (see Figure 1) are sorted in ascending order—first vertically, then horizontally. This insures that the resource can be redrawn neatly (GEM draws object trees from the top down). Unfortunately, it also insures that object indices do not relate to object functions in any way.

For most of the objects in the re-

Figure 1.


OVERALL PARENT G_BOX (LEVEL 0)
25.
DISPLAY, CONTROL KEYS, KEYPAD BOX (LEVEL 1)
N
D
C
KEYPAD KEYS (LEVEL 2)
(
)
/
*
7
8
9
-
4
5
6
+
1
2
3
=
0
.

source—the operator keys, for example—this doesn't matter, because the program has to handle each of them as a special case. For the digit keys, however, it matters. I hate the idea of building a lookup table or using a weird switch( ) statement to derive a number from 0 to 9 from whatever random object index ends up being correlated to a digit key.

Of course, the solution was right there in front of me, though it took me a remarkably long time to hit on it. Right, each digit key has a label string, pointed to by the ob_spec field of its OBJECT structure. That structure is stored in index order in the general resource OBJECT structure array. The string contains a digit from 0 to 9. Can you take it from there? Right. If you're sure the returned object index corresponds to a digit key, you can derive the integer value of that key by

```
digit = *calc[obj].ob_spec - '0';
```

As Alf says, "no problem."

The main work of resource design was carried out using the Kuma Resource Construction Set and saved in an .RSC file and an associated header (.H) file. Unlike some RCS utilities, the Kuma system cannot output a resource file in terms of C language data structures. Still, because it is impractical for a desk accessory to load an external resource file, it was imperative that this conversion be made so that I could imbed the resource in the body of my code.

Luckily, since the calculator was made up only of BOX- and BUTTON-type objects, it was easy to write a one-shot program to read the resource file and output the C language equivalent. I keep threatening to write a general purpose utility for this, but never seem to find the time.

The Calculator Algorithm

A calculator is a machine that performs mathematical operations entered in *infix* notation, so called since it predicates that operations be specified *between* the values to which they are applied (A+B), rather than before (*prefix*) or after (*postfix*).

The calculator does not operate by parsing whole expressions in the algebraic sense, applying a system of rules for operator precedence, but simply computes and maintains a running total as operands and operators become available (would that it were that simple—in fact, parentheses screw this up royally, but for the moment, we're talking basics). Thus, the expression

CALCULATOR

ATARI KEY

- Any Atari ST computer
- Mark Williams C Language Development System

Listing 1.

```
/***********************************************
          CALCULATOR DESK ACCESSORY

             By John B. Jainschigg

      (c) 1989, Atari Explorer Magazine
      7 Hilltop Road, Mendham NJ 07945

Written in Mark Williams C for the Atari ST.
Compile under MWC as follows:

>cc -f calc.c -o calc.acc -VGEMACC -lm

Key:
-f              Include floating point printf()
-o              Output executable file named calc.acc (not calc.prg)
-VGEMACC        Employ GEM desk accessory startup routines
-lm             Link in mathematics library

Compiled accessory occupies approximately 16K of RAM.
***********************************************/

#include <aesbind.h>
#include <obdefs.h>
#include <gemdefs.h>
#include <osbind.h>
#include <xbios.h>
#include <math.h>

/* resource set indices for CALC */

#define CALC     0      /* form/dialog */
#define DISPLAY  1      /* BUTTON in tree CALC */
#define NEGATE   2      /* BUTTON in tree CALC */
#define CLEARD   3      /* BUTTON in tree CALC */
#define CLEAR    4      /* BUTTON in tree CALC */
#define CBOX     5      /* IBOX in tree CALC */
#define LPAREN   6      /* BUTTON in tree CALC */
#define RPAREN   7      /* BUTTON in tree CALC */
#define DIV      8      /* BUTTON in tree CALC */
#define MULT     9      /* BUTTON in tree CALC */
#define B7       10     /* BUTTON in tree CALC */
#define B8       11     /* BUTTON in tree CALC */
#define B9       12     /* BUTTON in tree CALC */
#define MINUS    13     /* BUTTON in tree CALC */
#define B4       14     /* BUTTON in tree CALC */
#define B5       15     /* BUTTON in tree CALC */
#define B6       16     /* BUTTON in tree CALC */
#define PLUS     17     /* BUTTON in tree CALC */
#define B1       18     /* BUTTON in tree CALC */
#define B2       19     /* BUTTON in tree CALC */
#define B3       20     /* BUTTON in tree CALC */
#define EQUALS   21     /* BUTTON in tree CALC */
#define B0       22     /* BUTTON in tree CALC */
#define DECIMAL  23     /* BUTTON in tree CALC */

/* display control constants */

#define MAX_DISP        9               /* display width, chars */
#define MAX_DIGS        MAX_DISP - 1    /* above, minus sign char */
#define NXTOLAST        MAX_DISP - 2    /* position left of decimal pt. */
#define VERYLAST        MAX_DISP - 1    /* last char in display */
#define SIGN            0               /* sign char offset */
#define FSTPOS          1               /* first digit position */
#define SNDPOS          2               /* second digit position */

/* ST numeric keypad scancodes */

#define K_CLHM          0x47            /* clear/home key */
#define K_MINUS         0x4a            /* minus key */
#define K_PLUS          0x4e            /* plus key */
#define K_UNDO          0x61            /* UNDO key */
#define K_HELP          0x62            /* HELP key */
#define K_LP            0x63            /* left paren key */
#define K_EQ            0x72            /* enter key */

/* calculator machine states */

#define MNEWENTRY       0               /* start of operand input */
#define MENTRY          1               /* operand input in progress */
#define MOPERATOR       2               /* operator key pressed */
#define MEQUALS         3               /* equals key pressed */

/* various and sundry defines */

#define NOWINDOW        -1                      /* no window open */
#define WTYPE           (NAME | CLOSER | MOVER) /* wind. attribs. */
#define HI_RES          2                       /* mono code */
#define NUM_OBJS        24                      /* objects in resource */
#define NO_OBJ          -1                      /* "no object" code */
#define MAX_PLEVELS     49                      /* paren nesting limit */
```

```
/* Inline functions: push and pop operands, operators, flags, etc */

#define push_opd(op)     (opd_stack[opdsp++] = (op))
#define pop_opd()        (opd_stack[--opdsp]);

#define push_opr(op)     (opr_stack[oprsp++] = (op))
#define pop_opr()        (opr_stack[--oprsp])

#define push_opf(op)     (opf_stack[opfsp++] = (op))
#define pop_opf()        (opf_stack[--opfsp])

#define rdisp()          objc_draw(calc,DISPLAY,MAX_DEPTH,CRECT)
#define display          ((char *) calc[DISPLAY].ob_spec)

/* convenience equates to shorten window-management expressions */

#define CX       calc[0].ob_x
#define CY       calc[0].ob_y
#define CW       calc[0].ob_width
#define CH       calc[0].ob_height
#define CRECT    CX,CY,CW,CH
#define CPRECT   &CX,&CY,&CW,&CH
#define WRECT    w.x,w.y,w.w,w.h
#define WPRECT   &w.x,&w.y,&w.w,&w.h
#define TRECT    t.x,t.y,t.w,t.h
#define TPRECT   &t.x,&t.y,&t.w,&t.h

/* globals */

/* calculator resource data structure -- generated from .RSC file */

OBJECT calc[] = {
0xFFFF,0x1,0x5,0x14,0x0,0x0,0x1173L,0x15F,0x9,0x63,0x53,
0x2,0xFFFF,0xFFFF,0x1A,0x0,0x0,"           0.",0x9,0x2,0x50,0x8,
0x3,0xFFFF,0xFFFF,0x1A,0x41,0x0,"N",0xA,0xD,0x10,0x8,
0x4,0xFFFF,0xFFFF,0x1A,0x41,0x0,"D",0x35,0xD,0xF,0x8,
0x5,0xFFFF,0xFFFF,0x1A,0x41,0x0,"C",0x49,0xD,0x10,0x8,
0x0,0x6,0x17,0x14,0x0,0x0,0x11142L,0x5,0x17,0x59,0x3A,
0x7,0xFFFF,0xFFFF,0x1A,0x41,0x0,"(",0x5,0x3,0x10,0x8,
0x8,0xFFFF,0xFFFF,0x1A,0x41,0x0,")",0x1A,0x3,0x10,0x8,
0x9,0xFFFF,0xFFFF,0x1A,0x41,0x0,"/",0x2F,0x3,0x10,0x8,
0xA,0xFFFF,0xFFFF,0x1A,0x41,0x0,"*",0x44,0x3,0x10,0x8,
0xB,0xFFFF,0xFFFF,0x1A,0x41,0x0,"7",0x5,0xE,0x10,0x8,
0xC,0xFFFF,0xFFFF,0x1A,0x41,0x0,"8",0x1A,0xE,0x10,0x8,
0xD,0xFFFF,0xFFFF,0x1A,0x841,0x0,"9",0x2F,0xE,0x10,0x8,
0xE,0xFFFF,0xFFFF,0x1A,0x41,0x0,"-",0x44,0xE,0x10,0x8,
0xF,0xFFFF,0xFFFF,0x1A,0x41,0x0,"4",0x5,0x19,0x10,0x8,
0x10,0xFFFF,0xFFFF,0x1A,0x41,0x0,"5",0x1A,0x19,0x10,0x8,
0x11,0xFFFF,0xFFFF,0x1A,0x41,0x0,"6",0x2F,0x19,0x10,0x8,
0x12,0xFFFF,0xFFFF,0x1A,0x41,0x0,"+",0x44,0x19,0x10,0x8,
0x13,0xFFFF,0xFFFF,0x1A,0x41,0x0,"1",0x5,0x24,0x10,0x8,
0x14,0xFFFF,0xFFFF,0x1A,0x41,0x0,"2",0x1A,0x24,0x10,0x8,
0x15,0xFFFF,0xFFFF,0x1A,0x41,0x0,"3",0x2F,0x24,0x10,0x8,
0x16,0xFFFF,0xFFFF,0x1A,0x41,0x0,"=",0x44,0x24,0x10,0x13,
0x17,0xFFFF,0xFFFF,0x1A,0x41,0x0,"0",0x5,0x2F,0x25,0x8,
0x5,0xFFFF,0xFFFF,0x1A,0x61,0x0,".",0x2F,0x2F,0x10,0x8
};

char *noblanks();                               /* blank-stripping function */
char opr_stack[50];                             /* operator stack */
double opd_stack[50];                           /* operand stack */
char opf_stack[50];                             /* flag stack */
int oprsp,opdsp,opfsp;                          /* stack pointers */
int levels;                                     /* parenthesis levels */
int op = MNEWENTRY;                             /* calculator state */


/**********************************************
                MAIN PROGRAM
**********************************************/

main()
{
extern int gl_apid;                                  /* our application ID */

appl_init();                                         /* say "hi" to AES */
menu_register(gl_apid,"  Calculator");               /* put our label on menu */
events();                                            /* ... and go for it. */
}

/**********************************************
           ACCESSORY EVENT MANAGER
**********************************************/

events()
{
int ret,mx,my,key,dummy,message[8];                  /* for evnt_multi() */
Mouse mouse = {&mx,&my,&dummy,&dummy};               /* ditto */
Rect w,t;                                            /* window & temp rects */
int win = NOWINDOW;                                  /* window handle */
int obj;                                             /* object number */
int xd,yd;                                           /* window offsets */
int i;                                               /* looping variable */

if (Getrez() == HI_RES)                              /* alter coords per rez */
     for(i = 0;i < NUM_OBJS;i++){
          calc[i].ob_height = calc[i].ob_height * 2;
          calc[i].ob_y = calc[i].ob_y * 2;
          }
```

(Continued on next page)

5+4∗5=, which would produce the result 25 if evaluated algebraically (because multiplication has a higher precedence than addition), produces the result 45 when entered on a calculator.

A simple calculator can be made to function using only an accumulator—a memory register capable of holding a number—an operator storage register, and a flag. The basic logic is as follows:

(Registers and flag are clear)

1. Let the user enter an operand.

2. Terminate operand entry when the user presses an operator key (or the = key).

3. If the flag is clear, this is the first operand in the expression. Store it in the accumulator and store the operator in the operator store. Set the flag and go to 1.

4. If the flag is set, this is the second (or a subsequent) operand. Recover the last stored operator and use it to combine the present operand with the value in the accumulator. Put the result back in the accumulator, put the new operator in the operator store, and go to 1.

You will note, of course, that I am finessing the special issue of what happens when the = key is pressed—you *do* have to handle it differently from an operator. Still, the essentials are there, and it should be clear how this logic can be used to compute both simple (A+B=) and repeated (A+B+C=) expressions.

Occasionally, of course, you will run into some algebra dweeb who wants to evaluate an expression like 5+4∗5 and gets all bent out of shape when the answer 45 comes back. In a case like this, you have two choices: 1) Give him a smack and tell him that if wants to enter fancy-schmancy algebraic expressions, he will have to recast them in descending order of operator precedence. He wants the right answer, let him enter 4∗5+5! What, he's too lazy to do a little parsing? Give him another smack! Or, 2) implement parentheses and tell him what to do with them.

Deferred (Evaluation)

Malheureusement, this parenthesis fetish demands something a little more ambitious than our programmer-friendly little algorithm. But not much more difficult. At first glance, you might think that admitting parentheses would oblige you to store entire expressions and parse them syntactically, but nothing could be further from the truth.

Actually, a parenthetical expression is pre-parsed, in the sense that it is easy to tell the order in which computations must be performed to yield correct results. No, the trick is to figure out some way to defer evaluation of complex subexpressions (up to and including the entire expression) until the simpler subexpressions that comprise them have been evaluated. Does this sound recursive to you?

Recursive it is. To start off, you need a basic subroutine, modeled on the simple accumulator logic described above and capable of accepting entry of operands and operators and of maintaining a running total in its own little accumulator. Then you have to modify the subroutine to do the following things:

1. When it encounters a left parenthesis, it should call itself. When the call returns, it should employ the returned value as an operand.

2. When it encounters a right parenthesis, it should complete its current operation and return the value in its accumulator to its caller.

What happens when you throw an input stream like 5+(4*5)= at this subroutine? The routine accepts the 5 and the +, stores them in the accumulator and operator store, respectively, and sets its flag. Next, it encounters the left parenthesis, so it calls itself, creating a new instantiation with its own accumulator, operator store, and flag. The new instantiation accepts the 4 and the multiplication operator, and stores them. It then accepts the 5 and runs into the right parenthesis. Treating the right parenthesis as a signal to wrap things up, it evaluates the expression it has collected (4*5), and returns a result of 20 to its caller. Treating the returned value as an operand, the top level routine waits for the equal sign, then evaluates the expression 5+20. Result: 25. Just the an-

Trivia Quiz Answers

Questions are on page 14. 1. a; 2. c; 3. d; 4. c; 5. f (NCIC is the acronym for National Crime Information Center; the system described is Magnetic Ink Character Recognition, or MCIR for short); 6. f (By 1950, fewer than 50 computers of all types had been built); 7. t; 8. t; 9. f (In binary 1 + 1 = 11, in boolean 1 AND 1 = 1); 10. t; 11. d; 12. d; 13. d or e (Both came out in 1974; people are still arguing which was first); 14. a; 15. b; 16. a, c, and d; 17. a, b, and c; 18. a; 19. c; 20. c; 21. a; 22. b; 23. b; 24. d; 25. b.

Listing 1. (Continued)

```
form_center(calc,CPRECT);                                  /* center calculator */
wind_calc(0,WTYPE,CRECT,WPRECT);                           /* get window coords. */
xd = CX - w.x; yd = CY - w.y;                              /* set differences */

while(1){                                                  /* endless loop ... */

/* wait for a keyboard, mouse, or message event (or combination) */

     ret = evnt_multi(MU_KEYBD | MU_BUTTON | MU_MESAG,1,1,1,
           0,0,0,0,0,0,0,0,0,0,message,0,0,mouse,&key,&dummy);

     if (ret & MU_MESAG) switch(message[0]){ /* if message ... */

/* if someone wants the accessory open, either create a window or, if
   we already have one, top it */

          case(AC_OPEN):
               if (win == NOWINDOW){
                    win = wind_create(WTYPE,WRECT);
                    wind_set(win,WF_NAME,"BeanCalc",0,0);
                    wind_open(win,WRECT);
                    }
               else wind_set(win,WF_TOP,0,0,0,0);
               break;

/* if the system is about to take the window away, give it up */

          case(AC_CLOSE):
               win = NOWINDOW;
               break;

/* if someone wants the window shut down, close and delete it */

          case(WM_CLOSED):
               wind_close(win);
               wind_delete(win);
               win = NOWINDOW;
               break;

/* if someone moved it, update coordinates and reset it */

          case(WM_MOVED):
               w.x = message[4]; w.y = message[5];
               CX = w.x + xd; CY = w.y + yd;
               wind_set(win,WF_CURRXYWH,WRECT);
               break;

/* if system wants it on top, top it */

          case(WM_TOPPED):
          case(WM_NEWTOP):
               wind_set(win,WF_TOP,0,0,0,0);
               break;

/* if system wants it redrawn, chase rectangle list to redraw it */

          case(WM_REDRAW):
               wind_update(BEG_UPDATE);
               graf_mouse(M_OFF,0L);
               wind_get(win,WF_FIRSTXYWH,TPRECT);
               while(t.w && t.h){
                    objc_draw(calc,0,MAX_DEPTH,TRECT);
                    wind_get(win,WF_NEXTXYWH,TPRECT);
                    }
               graf_mouse(M_ON,0L);
               wind_update(END_UPDATE);

          }

/* if we get a keyboard message, convert the key to an appropriate
   ASCII value and send it to the calculator manager */

     if (ret & MU_KEYBD)
          if (obj = keyconv(key >> 8))
               manage_calc(obj);


/* if we get a mouse button message, determine which object the user
   clicked, and manage the calculator */

     if (ret & MU_BUTTON){
          obj =  objc_find(calc,CALC,MAX_DEPTH,mx,my);
          if (obj != NO_OBJ && obj != CALC && obj != CBOX && obj != DISPLAY)
               manage_calc(obj);
          }
     }
}

/************************************************
     CONVERT KEYCODE TO OBJECT NUMBER

Most of this can be handled by using the scancode
as an index into an array. Otherwise handle special
cases via switch.
************************************************/

keyconv(key)
unsigned int key;
{
```

```
static int keypad[] = {LPAREN,RPAREN,DIV,MULT,
                       B7,B8,B9,
                       B4,B5,B6,
                       B1,B2,B3,
                       B0,DECIMAL,EQUALS};

if (key >= K_LP && key <= K_EQ) return(keypad[key - K_LP]);

switch(key){
     case(K_HELP):  return(NEGATE);
     case(K_UNDO):  return(CLEAR);
     case(K_CLHM):  return(CLEARD);
     case(K_MINUS): return(MINUS);
     case(K_PLUS):  return(PLUS);
     }

return(0);
}

/**************************************************
                MANAGE CALCULATOR
**************************************************/

manage_calc(obj)
int obj;
{
static opf = 0;                                     /* operand flag */

objc_change(calc,obj,0,CRECT,SELECTED,1);           /* select object */

switch(obj){                                        /* dispatch */


/* If the clear key has been pressed, zero all stack pointers and
   the level count, clear the display, and set for new entry */

     case(CLEAR):   opf = opdsp = oprsp = levels = 0;
                    clear_disp();
                    op = MNEWENTRY;
                    break;

/* If the clear display key has been pressed and the machine is in
   the process of operand entry, clear the display and set as if
   operand entry were beginning */

     case(CLEARD):  if (op == MENTRY){
                         clear_disp();
                         op = MNEWENTRY;
                         }
                    break;

/* If an operator key has been pressed and the machine is awaiting
   operator entry, process the operator. Start by pushing the operand
   on the display, and switch the operand count flag. Call
   evaluate() if the operand count is > 0. Then push the operator. */

     case(PLUS):
     case(MINUS):
     case(MULT):
     case(DIV):     if (op == MENTRY || op == MEQUALS){
                         push_opd(atof(noblanks()));
                         opf++;
                         if (opf == 2){
                                evaluate();
                                opf = 1;
                                }
                         push_opr(obj);
                         op = MNEWENTRY;
                         }
                    break;

/* If the equals key has been pressed and the machine is capable of
   performing a pending evaluation, push the operand on the display
   and do it. */

     case(EQUALS):  if (op == MENTRY && opf == 1){
                         push_opd(atof(noblanks()));
                         evaluate();
                         pop_opd();
                         opf = 0;
                         op = MEQUALS;
                         }
                    break;

/* If the negate key has been pressed invert the sign of the display */

     case(NEGATE):  display[SIGN] = (display[SIGN] == '-')? ' ' : '-';
                    rdisp();
                    break;


/* If the left paren key has been pressed and the machine is awaiting
   operand entry, bump up a paren level */

     case(LPAREN):  if (op == MNEWENTRY && levels < MAX_PLEVELS){
                         push_opf(opf);
                         opdsp++;
                         oprsp++;
                         levels++;
                         opf = 0;
                         }
                    break;
```

(Continued on next page)

swer your dweeb was looking for.

Because C is a fully recursive language, I knew it would be possible to implement the above procedure exactly as described, developing a basic routine for expression entry that calls itself as required to solve more complex expressions. However, there are many things you have to watch out for when writing literally recursive code.

For one thing, you have to watch out for stack overflow. For another, you have to figure out escape routes back to top level for use in emergency situations—for example, when somebody presses the Clear All key after entering half of a deeply nested parenthetical expression. All in all, I decided that a pseudo-recursive approach would be easier to manage.

The method I settled on employs two FIFO stacks—one for operands, one for operators. As operands and operators are accepted, they are pushed onto the appropriate stacks, and the stack pointers are incremented. Each time two operators mount up at a given level, a function called evaluate( ) pops them off (decrementing the operand stack pointer), pops an operator (decrementing the operator stack pointer), performs the math, and pushes the result back on the operand stack (incrementing the operand stack pointer). Figure 2 shows what happens when you hand the stack system a simple expression such as 5+4=.

To implement the recursion necessary to handle parentheses, you simply reset your first/second operand flag when you encounter a left parenthesis in the input stream. The system thus fails to call evaluate( ) when an operand is subsequently entered, permitting the stack to increase in depth by one level.

A right parenthesis causes evaluation to take place, reducing the depth of the stack by one level. Figure 3 shows how the stack system works when handling the expression 5+(4*5)=. *Quod erat demonstrandum* . . . all else is syntax checking.

The State Machine

Now syntax checking ain't (as my grandmother used to say) chopped liver, either. The stack system as implemented above is very frail—it needs to be protected against spurious input. The way I decided to do this was to begin by building a low-level set of basic routines, each of which would handle a certain type of input in all situations.

There is a routine to handle operator

entry, a routine to handle operand entry, a routine to handle the equals key, etc. Whenever any one of these routines executes, it changes the state of the machine in a definitive manner. The current state is recorded in a global variable, op.

When one of these routines is called by the keypad dispatcher, it begins by examining the machine state variable and deciding whether (and if so, how) it should execute. This is usually a simple yes/no decision, though in the case of certain routines, such as the operand entry module, the state of internal variables must be consulted. Then it does (or doesn't do) its little thing, whatever that is. Finally, it resets the machine state variable to reflect whatever it has accomplished.

The result is a system that doesn't have to depend on complex, high-level logic to dispatch and execute functions. Instead, the responsibility for deciding when and how to execute is distributed at a fairly low level.

The Math System

This part didn't make me feel so good. I decided early on that all operands would be stored and handled as double-precision floating point numbers for maximum accuracy. A double, in Mark Williams C DECVAX format, is an eight-byte data unit that consists of one sign bit, eight bits of binary exponent, and 55 bits of binary fraction. That means it can represent, roughly speaking, numbers between 10^{38} and 10^{-37} to around 16 decimal digits of accuracy. This is not too shabby.

Unfortunately, even with this broad and capacious format, rounding and truncation errors inevitably creep into computations. The result is that occasionally you will perform what you think is a fairly plain-vanilla operation series, and the machine will come back at you with a result that looks wrong. Something like 10/3*3—the answer is 10, right? Then why does the machine keep insisting that it is 9.9999999999?

Standard calculators have two ways of getting around this little problemette. Either they use Binary-Coded Decimal (BCD) math to handle midrange value computations, or they incorporate heuristics to actually modify certain kinds of irrational-looking (in every sense of the term) results. Thus, when I perform the 10/3*3 test on my very expensive TI SR-40 scientific calculator, I get 3.333333333 as the intermediate result and 10 as the terminal result. However,

Listing 1. (Continued)

```
/* If the right paren key has been pressed and the machine is capable
   of performing a pending evaluation, do it and jump down a paren
   level */

     case(RPAREN):  if (op == MENTRY && opf && levels){
                         push_opd(atof(noblanks()));
                         evaluate();
                         pop_opd();
                         levels--;
                         opdsp--;
                         oprsp--;
                         opf = pop_opf();
                         }
                    break;

/* Otherwise it must be a number key or something similar, so pass
   it on to the operand entry manager */

     default:       do_entry(obj);
     }

objc_change(calc,obj,0,CRECT,NORMAL,1);               /* deselect object */
}

/*********************************************
           CLEAR PHYSICAL DISPLAY
*********************************************/

clear_disp()
{
strcpy(display,"       0.");
rdisp();
}


/*********************************************
MAKE ENTRY: ENTER DIGIT OR DECIMAL, CHANGE SIGN
*********************************************/

do_entry(v)
int v;
{
static int dp = 0,nd = 0;

if (op == MNEWENTRY){
     clear_disp();
     dp = nd = 0;
     if (v == B0) return;
     }

if (v == DECIMAL){
     dp = 1;
     if (op == MNEWENTRY) nd = 1;
     }

else {
     if (op ==  MNEWENTRY || (display[SIGN] == '-' && nd == 0))
          display[NXTOLAST] = ' ';
     if ((nd + 1) < MAX_DIGS){
          strcpy(display + FSTPOS,display + SNDPOS);
          display[NXTOLAST + dp] =
               * ((char *) calc[v].ob_spec);
          if (!dp) display[VERYLAST] = '.';
          nd++;
          }
     rdisp();
     }
op = MENTRY;
}

/**********************************************
          EVALUATE A SUBEXPRESSION
**********************************************/

evaluate()
{
float opd1,opd2;
char oper;

opd2 = pop_opd();          /* pop first operand (note reverse order) */
opd1 = pop_opd();          /* pop second operand */
oper = pop_opr();          /* pop operator */


switch(oper){
     case(PLUS):    push_opd(opd1 + opd2);
                    break;
     case(MINUS):   push_opd(opd1 - opd2);
                    break;
/* handle "divide by 0" errors here */
     case(DIV):     if (opd2 == 0) push_opd((double) 0);
                    else push_opd(opd1 / opd2);
                    break;
     case(MULT):    push_opd(opd1 * opd2);
                    break;
     }
frame(opd_stack[opdsp - 1]);               /* display result */
rdisp();
}
```

```
/**********************************************
           FORMAT DOUBLE FOR DISPLAY
***********************************************/

frame(f)
double f;                                          /* display value */
{
char buf[60];                                  /* 1.e+37 + 11 precision = 49 */

sprintf(buf,"%8.6f",f);                        /* convert to ASCII float format */

if (ezdp(buf)){                                /* If <= 12 chars, minus tr. 0's ... */
      strcpy(display,"            "); /* Clear display */
      display[SIGN] = (buf[SIGN] == '-')? '-' : ' ';  /* Move sign */
      strcpy(display + MAX_DISP - strlen(buf) + (f < 0),buf + (f < 0));
      }
else sprintf(display,"%.1e",f);
}

/**********************************************
        ELIMINATE TRAILING ZEROS IN BUFFER
***********************************************/

ezdp(s)
char *s;
{
int i = strlen(s) - 1;
while(s[i] == '0') s[i--] = '\0';
return(strlen(s) <= MAX_DISP);
}



/**********************************************
      ELIMINATE BLANKS FROM BUFFER FOR ATOF()
***********************************************/

char *noblanks()
{
static char buf[MAX_DISP];
int i = 0,j = 0;

for(;j < MAX_DISP;j++) if (display[j] != ' ') buf[i++] = display[j];
buf[i] = '\0';
return(buf);
}
```

	Stacks:		Stack Pointers:	
User enters	Operand	/Operator	Operand	/Operator
5	Empty	Empty	0	0
+	5	+	1	1
4	5	+	1	1
=	5,4	+	2	1
	. . . evaluate( ) is called . . .			
	9	Empty	1	0

Figure 2. Simple expression-handling by the stack system.

	Stacks:		Stack Pointers:		
User enters	Operand	/Operator	Operand	/Operator	Flag
5	Empty	Empty	0	0	0
+	5	+	1	1	1
(	5	+	1	1	0
	. . . flag is reset . . .				
4	5	+	1	1	0
*	5,4	+,*	2	2	1
	. . . thus evaluate( ) is NOT called . . .				
5	5,4	+,*	2	2	1
)	5,4,5	+,*	3	2	1
	. . . evaluate( ) is called . . .				
	5,20	+	2	1	1
=	5,20	+	2	1	1
	. . . evaluate( ) is called . . .				
	25	Empty	1	0	1

Figure 3. Complex expression-handling by the stack system.

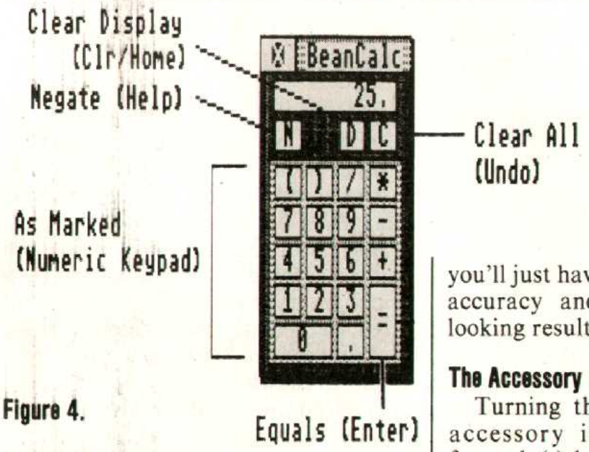


Figure 4.

when I manually enter 3.33333333 and multiply it by 3, I get 9.99999999. Neat, huh?

Well, don't look for that kind of sophistication, here. This is where I stop. Maybe on version 2.0—until that time, you'll just have to be satisfied with high accuracy and the occasional weird-looking result.

The Accessory Frame

Turning the calculator into a desk accessory involved replacing the form_do( )-based application dispatcher with a lower-level dispatcher built around evnt_multi( ), then building a simple window-management system around it.

Evnt_multi( ) waits for a mouse, keyboard, or system-dispatched message event, and returns with data corresponding to the event in hand. A switch( ) statement then dispatches activity according to conditions. If the accessory is currently on-screen and active (topped), mouse and keyboard events are converted to object numbers corresponding to calculator keys, and passed into the calculator dispatcher. Message events tell the accessory when to appear, disappear, redraw itself, or reset its coordinates following a move.

Free Software

What you see is what you get. Figure 4 designates the on-screen controls of the calculator and the corresponding keypad and keyboard keys. Listing 1 contains the calculator code itself—a monument to the Mark Williams C Language development system.

If you own said system, you can actually type the program into a file called calc.c and compile your own working version using the command:

```
cc calc.c -o calc.acc -VGEMACC -lm
```

which means "compile the source file calc.c, link it to the AES and VDI libraries, the math library, and the desk accessory runtime startup routine, and call the output file calc.acc" in compiler talk.

If you wish to have absolutely nothing to do with compilers at all, you can download the calculator source and a compiled accessory from the *Atari Explorer* section in the Genie ST Roundtable or from the *Explorer* section on the CompuServe Atari Productivity Forum. ■

C Life

An introduction to the resource files

By JOHN JAINSCHIGG

Over the past several issues, we have mastered some tricks for managing GEM resources in C. In the process, we have learned to build simple dialog boxes by hand, representing them as data structures bound into C language source code. Knowing how to do this is useful, at least in principle. (Situations *do* arise in which resources should be initialized this way.) But there is no getting around the fact that hand-design is a tedious and painful business for all but the simplest graphic objects.

Resource data structures are, after all, complex—packed densely with parametric information vital to the behavior of the resource under AES management and even (in certain circumstances) to the integrity of the AES itself. Resources are, moreover, the most visible elements in a typical ST program, so it is important to have them "look right."

With all this at stake, production GEM programmers rarely design resources by hand. Instead they employ a utility called a Resource Construction Set—a GEM-based, point-and-click application that lets them create resource sets on-screen in (now get this) simple, intuitive fashion. Resources can then be saved to disk either in .RSC format (for runtime access via the AES function rsrc_load( )) or even as C language source code (not all RCS systems provide this latter feature).

Several RCS systems are currently available. The most popular is probably the Atari RCS, which is included with the ST developer's package. Others include the Mark Williams RCS, distributed with versions 3.0 and higher of the Mark Williams C Language Development System; the Megamax RCS, included with Laser C version 1.0; Kuma's K-Resource; and several public domain offerings. Each has its own strengths, weaknesses, endearing quirks, and not-so-endearing bugs; all are more-or-less similar in appearance and basic function.

Resource Files

Before undertaking to describe how an RCS is used, I want to take a moment to examine the files it is employed to create and the way in which these fit into the overall scheme of GEM. RCS programs are usually employed to create .RSC files—files containing from one to several resource trees, packed in the format shown in Figure 1.

The AES function rsrc_load( ) is used to load an .RSC file into the memory of the ST. The function works by allocating RAM for the resource, reading its image into this space, and doing a linker-like "fixup" on absolute pointers in the file (for example, the ob_spec field of a button object contains an absolute character pointer to its label string) to correspond to the absolute location of the resource. Because memory for resources is allocated dynamically, these addresses cannot be known prior to runtime. It returns zero if an error occurs during the load process and a number greater than zero if no error occurs. Only one resource file can reside in memory at any one time.

Once a resource has been loaded, an application can determine the address of the OBJECT structure for each of its root objects using the AES function rsrc_gaddr( ), called as follows:

```
OBJECT *p;

rsrc_gaddr(type,index,&p);
```

where "type" is an integer between 0 and 16 denoting the type of object sought (there are bugs in the function that make it useless for object types other than trees: type 0), "index" is the number of the tree whose address you wish to find (the first tree in a resource has index 0, the second index 1, etc.), and "&p" denotes the address of the pointer in which you want the address of the OBJECT stored. The function returns a zero on error and a number greater than zero on success.

This sounds complicated, but is really very straightforward in practice. For example, suppose you have used an RCS to construct the pair of resources shown in Figure 2.

The RCS has magically saved the resources in a file called SAMPLE.RSC. Your program might use the code in Listing 1 to load SAMPLE.RSC and prepare to exploit the resources it contains.

Once the above function has been executed, you can proceed to display or otherwise manipulate the resources on an

1. Number of trees in resource file.
2. List containing the integer offset of each root object in subsequent OBJECT array.
3. Array of OBJECT structures.
4. Other structures (TEDINFO, etc.) required by the objects in the array.
5. Strings, bit-images, and other data.

Figure 1. Resource file structure

```
#include <aesbind.h>      /* AES binding declarations */
#include <obdefs.h>       /* object definitions */

#define FORM1        0          /* index of first dialog root */
#define FORM2        1          /* index of second dialog root */

#define TREE         0          /* tree object type for
                                   rsrc_gaddr() */

OBJECT *firstdial,*secondial; /* Declared as globals since other
                                   functions will want to use them
                                   once they are initialized. */

load_rsc()     /* load resource file and initialize pointers */
{
if (!rsrc_load("SAMPLE.RSC"))  /* i.e., if the resource can't be
                                    found, or is corrupted ... */
     return(-1);               /* ... return an error code. */

/* Now get the addresses of the OBJECT structures for the root
   objects of each tree. Return -1 on error. */

if (!rsrc_gaddr(TREE,FORM1,&firstdial)) return(-1);
if (!rsrc_gaddr(TREE,FORM2,&secondial)) return(-1);

return(0);     /* Return "okay" */
}
```

Listing 1. Code used to load and prepare to use the resources contained in SAMPLE.RSC.

individual basis, using their root object pointers. For example, you might center and display the first dialog box with the following two function calls:

```
form_center(firstdial,&firstdial[0].ob_x,
                      &firstdial[0].ob_y,
                      &firstdial[0].ob_width,
                      &firstdial[0].ob_height);

objc_draw(firstdial,0,MAX_DEPTH,firstdial[0].ob_x,
                      firstdial[0].ob_y,
                      firstdial[0].ob_width,
                      firstdial[0].ob_height);
```

Resource Deallocation

Once you have finished using the resources in a file, you deallocate the memory used to store the resource, using the AES function rsrc_free( ), called as follows:

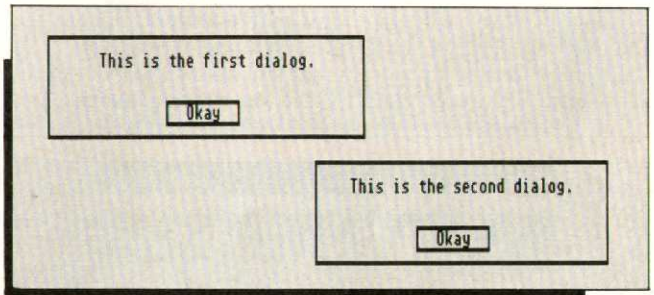
```
rsrc_free();    /* Deallocate memory used to
                   store resource file */
```

Naturally, rsrc_free( ) cannot take an argument, because GEM permits only one resource file to reside in memory at a time. This imposes limitations on the use of resource files by applications.

For example, it is impractical to create separate resource files for menus, dialogs, and other object types, because loading any one of them would force deallocation of memory for the rest (rsrc_load( ) calls rsrc_free( ) automatically, before loading a resource file). Another example: it is impractical for desk accessories to make use of external resource files, because loading them would conflict with resources used by main applications.

.H Files

In using the RCS to design a resource, you have the opportunity to give a name to each tree and object you create. The RCS maintains a database of these names, correlating them with tree and object indices. When you save your resources to disk as an .RSC file, the RCS also produces a C language "header" file (.H extension) in ASCII format, containing



This is the first dialog.

This is the second dialog.

Figure 2. Dialog boxes contained in the file SAMPLE.RSC.

```
#define   FORM1      0   /* form/dialog */
#define   DIAL1      0   /* BOX in tree FORM1 */
#define   BUTTON1    1   /* BUTTON in tree FORM1 */

#define   FORM2      1   /* form/dialog */
#define   DIAL2      0   /* BOX in tree FORM2 */
#define   BUTTON2    1   /* BUTTON in tree FORM2 */
```

Figure 3. The header file SAMPLE.H.

#defines equating your object names to their indices in the resource OBJECT structure array.

Figure 3. shows the header file SAMPLE.H that might be produced by a typical RCS in association with the resource file SAMPLE.RSC.

The header file is, in effect, a map of the structure of the resource set. Each group of #defines begins with the name of a tree in the resource file, equated to its tree index. Thus the first dialog, FORM1, has index 0, and the second, FORM2, index 1. Grouped beneath each tree index are indices for each object in that tree, denoting the offset of OBJECT structures in the overall OBJECT array of that tree.

Using such equates can simplify code used to manage a resource, largely because it clarifies the array notation used to refer to individual objects. For example, rsrc_gaddr( ) might be used as follows to initialize an OBJECT structure pointer to the root of the first tree in SAMPLE.RSC:

```
#include "sample.h"
  .
  .
  .
OBJECT *first;
  .
  .
  .
rsrc_gaddr(0,FORM1,&first);    /* initialize
                                  pointer to tree */
```

Thereafter, OBJECT structure elements for individual objects in that tree can be referred to by a combination of array and structure notation, using equates from the header file. For example, the notation first[BUTTON1].ob_x may be used to retrieve the X-offset of the button from its parent, the BOX object that forms the outline of the dialog.

Using the RCS

Now that we have a basic understanding of RCS output files and how they are used, let's walk through a sample session with the RCS itself. In the process, we'll learn a little more GEM resource arcana—just enough to make the work interesting.

Imagine, if you will (like most native writers, I adore the Rod Serling effect of the phrase "Imagine, if you will . . . ") that we are writing a program that will mimic the front panel of a digital FM stereo tuner. At the top center of our display, we want a digital readout, capable of representing a decimal number between 87.5 and 108.0 (you don't have to check, I'm looking at my stereo right now).

Underneath, we want a pair of scanning buttons, one to move us up through the frequency band, the other, down. Beneath that, we want a group of, say, eight "preset" buttons, each of which will move us instantly to a given frequency. Finally, to make things really interesting, we want to be able to initialize the presets from the panel. To do this, we will need a "memory" button—press it, then a preset, and the frequency currently displayed on the readout will be assigned to that preset.

Designing the Panel

Booting our trusty generic RCS, we perform the machinations and magic passes required to set up a new resource file named FM.RSC. An appropriately labeled icon appears on the work screen. Double-clicking on this icon causes it to expand into a working window.

The resource we are about to create is essentially a complex dialog box or gem "form" (as distinct from a menu, an alert, an icon, or other type of resource tree) (see sidebar "Anatomy of a Resource"). To establish a form tree, we select TREE from the NEW menu, naming the resulting tree icon PANEL and selecting FORM from the list of types offered to us by the resulting dialog. A new work window appears, containing the outline of a dialog box.

Grabbing the lower right-hand corner of the dialog, we

INDEX TO ADVERTISERS

Alpha Systems	49
American TV	43
Astra Systems	25
Atarian	Cover 4
Atari Corp.	62-63
Atari Explorer	Cover 3
Avant-Garde Systems	13
BRE	42
Computer Cache	42
Computer Games +	2
Dealer Directory	78-79
Future Systems	15, 24, 38
Gribnif Software	38
IB Computers	33
Intersect Software	33
Joker Software	Cover 2
Micromiser	52
Migraph	21
Navarone	11
Park Avenue Video	52
Practical Solutions	1
Prospero Software	33
Scan-Tech	24

Puzzles & Problems

ANSWERS

Problems are on page 10.

Six Beneficiaries

Thomas and May Hughes inherited $284,000 and $142,000 respectively; Albert and Jean Smith, $198,000 and $132,000; and Henry and Kate March each received $122,000.

Sum of Combinations

The sum of all 12 possible combinations, ab, ba, ac, . . . , yields the equation 99a+99b+99c+99d= 000a+100b+10c+d. Using some logic to solve the equation, we find the number is 1782.

Simple Division by Three

428,571 when divided by 3 equals 142,857; also 857,142 when divided by 3 equals 285,714.

Simple Division by Four

The numbers are 307,692; 615,384; and 923,076.

Two Towers

32 and 18 rods, respectively.

Cryptic Peace

9338+932=10270.

Six Authors

Allen writes novels, Baker is a poet, Chapman is a playright, Davis writes humor, Egmont is a historian, and Frazer writes short stories.

The Escalator

100 steps.

All Numerals From 1 to 9

The only other such pair is 18×297=5346 and 27×198=5346.

Anatomy Of A Resource

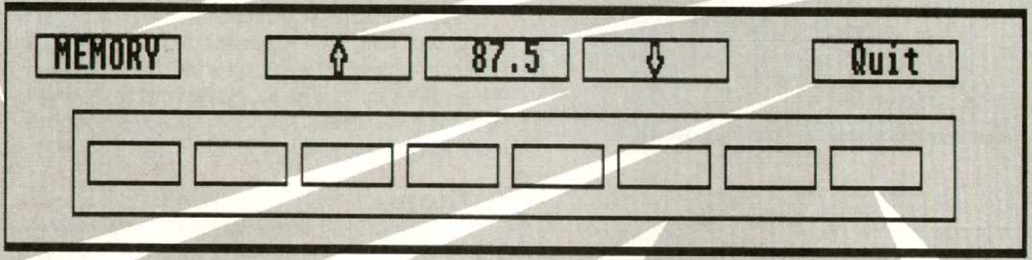
FM.RSC, the resource used by FM.C

Object: BOX
Type: BOX
Flags: None
Children: MEMORY, FREQUP, FREQDN, DISPLAY, INVISBOX

Object: MEMORY
Type: BUTTON
Flags: SELECTABLE/TOUCHEXIT
Children: None

Object: FREQUP
Type: BUTTON
Flags: TOUCHEXIT
Children: None

Object: DISPLAY
Type: BOXTEXT
Flags: None
Children: None



Object: FREQDN
Type: BUTTON
Flags: TOUCHEXIT
Children: None

Object: QUIT
Type: BUTTON
Flags: TOUCHEXIT
Children: None

Object: INVISBOX
Type: IBOX
Flags: None
Children: B1-B8

Objects: B1-B8
Type: BUTTON
Flags: SELECTABLE/TOUCHEXIT/RBUTTON
Children: None

stretch it out into the kind of horizontal rectangle we require for our control panel. Double-clicking in the interior of the rectangle causes an attribute panel to appear. It contains controls that let us name the object (we name it BOX), adjust border thickness, color, and fill pattern, and set object attributes such as SELECTABLE and TOUCHEXIT. Since the box itself will not be used as a control in our program, we make sure that no special attributes are set.

Next, we build ourselves a digital display. Going back to the NEW menu, we select OBJECT, choosing a BOXTEXT object (a normal-size text string surrounded by a box) and dragging it to the approximate center of our dialog box, above the midline.

By putting this object down inside the containing rectangle, we cause it to be considered the child of the dialog box outline. This "visual hierarchy" rule is always observed by the RCS: putting one object on top of or within another makes the first the child of the second.

Double-clicking on the BOXTEXT object, we use the resulting dialog to set color, fill, and border attributes. Since this object will not be used as a control, we set neutral GEM attributes. Finally, we name the object DISPLAY and enter its GEM attributes to neutral.

At the same time, we name the object DISPLAY and enter an initial text string in the following format: " 87.5". Our program will alter the displayed frequency on-screen by modifying the string and redrawing the DISPLAY object; we leave the leading space so that we are sure to have enough room to store a four-digit number with decimal point.

Next, we make the frequency scan controllers, selecting a pair of BUTTON objects and dragging them to either side of the display. Double-clicking on each in turn, we name them FREQUP and FREQDN (for Frequency Up and Down), modify their labels to contain the words Up and Down respectively (or, better yet, employ the special GEM up-arrow and down-arrow characters as labels!), and make sure each is identified as TOUCHEXIT. This will insure that when a user clicks on either one of them when our form is being managed by GEM, control will return to our program so that we can increment or decrement the frequency, appropriately. Finally, we resize the buttons to match one another.

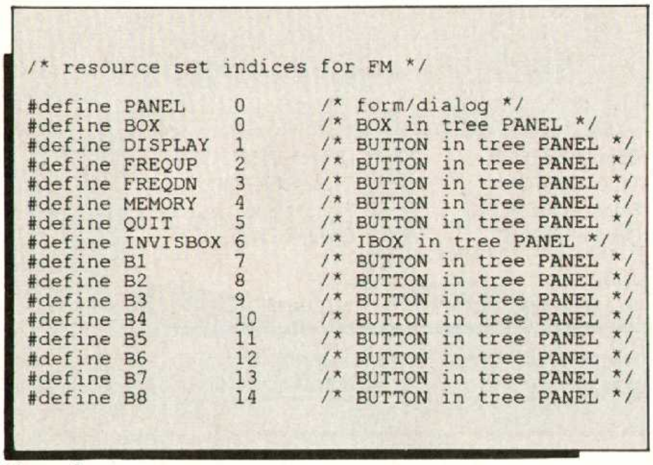
Now we make the memory button, selecting a BUTTON object, dragging it to the right side of the display, naming it MEMORY, and labeling it likewise. GEM attributes are set to SELECTABLE/TOUCHEXIT, telling GEM to darken the button when it a user clicks on it and to return control to our program at this point.

Finally, so that our user can exit from the program, we create a QUIT button, naming it Quit, labeling it likewise, identifying it as TOUCHEXIT, and placing it in the lower right-hand corner of our control panel.

Radio Buttons

Finally, we are ready to make our brace of eight preset buttons. Ideally, we would like these to work like the channel selectors on a car radio: when one is clicked on, it should darken, and any other darkened button should "pop out" (that is, be redrawn in normal black-on-white). Surely, if we were to identify each of the buttons as SELECTABLE and TOUCHEXIT, we could write C code to manage their mutual exclusivity. But it's easier to let GEM do it.

This can be done by identifying each button with the GEM attribute RBUT (short for radio button). Sibling objects flagged in this manner are handled in mutually exclusive fashion by GEM: if one is selected, the others are automatically de-selected. The only complication is in grouping radio button objects together in such a way that GEM doesn't get confused by the presence of other, non-radio-button siblings



```
/* resource set indices for FM */

#define PANEL     0     /* form/dialog */
#define BOX       0     /* BOX in tree PANEL */
#define DISPLAY   1     /* BUTTON in tree PANEL */
#define FREQUP    2     /* BUTTON in tree PANEL */
#define FREQDN    3     /* BUTTON in tree PANEL */
#define MEMORY    4     /* BUTTON in tree PANEL */
#define QUIT      5     /* BUTTON in tree PANEL */
#define INVISBOX  6     /* IBOX in tree PANEL */
#define B1        7     /* BUTTON in tree PANEL */
#define B2        8     /* BUTTON in tree PANEL */
#define B3        9     /* BUTTON in tree PANEL */
#define B4        10    /* BUTTON in tree PANEL */
#define B5        11    /* BUTTON in tree PANEL */
#define B6        12    /* BUTTON in tree PANEL */
#define B7        13    /* BUTTON in tree PANEL */
#define B8        14    /* BUTTON in tree PANEL */
```

Figure 4.

at the same level in the object tree.

A simple trick is used to get around this problem. By assembling radio button objects within the confines of a BOX-type object, the BOX becomes their parent according to the visual hierarchy rule, grouping them together as "sole siblings." Nor are we obliged to reveal this trick to the user of our program; an invisible box (IBOX) object will work just as well as a BOX in this case.

Selecting a BOX object from the NEW menu, we move it into position on our dialog background and resize it until it is big enough to contain eight buttons. We don't have to be too precise at this point, since the RCS will let us modify the size of the box later on. As noted above, we might also have used an IBOX, though invisible objects are difficult to manage on-screen. Instead, we'll use this regular box until everything is in order, then finally render it invisible by setting its border to zero.

Next, we get a BUTTON object from the NEW menu and stick it in the parent box at the left-hand side. This will be the first of our preset buttons. We give it the name B1 and remove its text entirely, leaving it blank. SELECTABLE, TOUCHEXIT, and RBUT attributes are selected. Now, by pressing the Shift key and clicking on the button with the mouse, we pick up a duplicate of it (most RCS systems will let you duplicate objects this way), moving right and clicking to deposit seven perfect copies of our button object at successive positions within the enclosing box. Double-clicking on each of them, in turn, we name them B2 through B7, noting as we go that each duplicate button is identified as SELECTABLE/TOUCHEXIT/RBUT, like the first.

Now, let's think for a moment about how our program will work. Each of the frequency scan controls performs a unique action, as does the memory button, so it is not important what indices they return when they get clicked on, so long as we can identify them. The radio buttons, on the other hand, all do the same thing—set a frequency—and it would be nice to have them return successive indices. That way, our program could use the button indices to reference elements of an array used to store the frequency to which each button corresponds.

RCS convention implies that sibling objects will have successive indices corresponding to the order in which they are added to a resource. Unfortunately, we may have mixed up the order of our radio buttons in the process of moving them around. Luckily, RCS systems offer a nifty function that can sort objects at any level of a resource tree by X-coordinate (sort horizontally), Y-coordinate (sort vertically), or both (sort horizontally, then vertically).

This sort operation does not alter the position of objects on-screen. Rather, it re-orders the resource OBJECT array so that objects are represented there in the same order as they appear on-screen. By sorting the radio buttons by X-coordinate, we insure that they have successive indices corresponding to their individual screen positions. Then, by sorting all objects in the dialog box by X-, then by Y-coordinate, we insure that GEM will draw the dialog in a clean, orderly fashion (GEM draws sibling objects in a resource in index order).

As a final step, we double-click on the radio button containment box and set its border to zero, rendering it invisible. Done. We close all working windows and select SAVE from the RCS FILE menu. The RCS responds by generating a resource file, FM.RSC, and a header file containing our object names, FM.H (see Figure 4).

The Program

FM.C, our digital-tuner demonstration program (less music and talk shows, of course), is shown in Listing 2. As expected, the resource file is loaded with rsrc_load( ), and pointers to it are initialized with rsrc_gaddr( ). The form is then drawn on the screen, and control of it passed to GEM with form_do( ). Form_do( ) manages the form until the user clicks on an exit object (TOUCHEXIT or EXIT). It then passes control back to our program, returning the index of the object used to close the dialog.

A "switch" statement is used to dispatch operations, according to the index of the control selected. Clicking on a frequency scan control causes the current frequency to be incremented or decremented by tenths (note that this is done with integer math—more efficient than floating point), causes the resulting figure to be converted to ASCII and inserted in the label string of the display object, then, finally, causes that object to be redrawn in situ.

Clicking on the MEMORY button causes it to be selected or deselected by form_do( ). The state of the MEMORY button serves as a flag for the code that manages the presets. If the MEMORY button is selected when a preset is pushed, we assume the user wishes to memorize a frequency. We place the current frequency in the slot of the preset in the memory array, normalize the state of the MEMORY button and the preset, and redraw both objects, "clearing" the memory request. If, however, a preset is pushed when the state of the MEMORY button is normal, we assume the user wishes to implement the frequency associated with that preset. Thus we reinitialize the current frequency from the memory array and redraw the display.

Note that the memory array is accessed by subtracting the index of button B1 from the returned button value to produce a range of indices from 0 to 7. This exploits the fact that our radio buttons return sequential indices from left to right. Note also that when one button is clicked, it darkens, and any other darkened button is inverted to normal, courtesy of the GEM Radio Button feature.

Clicking the QUIT button breaks the loop, deallocates the resource, and terminates the application.

In our next issue, we will conclude our discussion of dialog boxes by using the RCS to design some more interesting controls—slider switches, locator grids, etc. Then, we will break out into new resource territory, and learn something about GEM menus. ■

FM.C

ATARI KEY

- Any Atari ST computer
- Any Resource Construction Utility
- Mark Williams C Language Development System

Listing 2. FM.C, FM digital tuner demonstration program. Requires that the user employ an RCS to create the resource file FM.RSC and the associated header file, FM.H, described in the text.

```
#include <aesbind.h>                    /* AES bindings */
#include <obdefs.h>                     /* object definitions */
#include "fm.h"                         /* place in same folder as source */

#define TREE            0               /* for rsrc_gaddr() */
#define ROOT            0               /* for objc_draw() */
#define REDRAW          1               /* for objc_change() */
#define ARROW           0               /* to avoid #including gemdefs.h */
#define MIN_FREQ        875             /* minimum frequency (x 10) */
#define MAX_FREQ        1080            /* maximum frequency (x 10) */

/* When a user double-clicks on a TOUCHEXIT object under the control
   of form_do(), the integer index of that object is returned with its
   high bit (bit 15) set. This is an "undocumented but useful" feature.
   This mask is used to force that high bit to zero, in self-defense. */

#define DCL_MASK        0x7fff

OBJECT *panel;                          /* pointer to dialog */

main()
{
static int memory[] =           /* memory array for presets */
        {875,875,875,875,875,875,875,875};
int cur_freq = 875;             /* current frequency on display */
int ret;                        /* return value */

appl_init();                            /* Init AES application */

graf_mouse(ARROW,0L);                   /* Change mouse pointer to arrow */

if(!rsrc_load("fm.rsc") ||              /* Load resource and set pointer */
   !rsrc_gaddr(TREE,BIGBOX,&panel)){    /* .. Note "short-circuit" */
        appl_exit();                    /* .. evaluation of error returns */
        exit(-1);
        }

form_center(panel,&panel[BIGBOX].ob_x,  /* Alter coords. to center form */
                  &panel[BIGBOX].ob_y,
                  &panel[BIGBOX].ob_width,
                  &panel[BIGBOX].ob_height);

objc_draw(panel,ROOT,MAX_DEPTH,         /* Draw the control panel */
                  panel[BIGBOX].ob_x,
                  panel[BIGBOX].ob_y,
                  panel[BIGBOX].ob_width,
                  panel[BIGBOX].ob_height);

do {                                    /* Management loop. Forever .. */

   switch(ret = form_do(panel,0) & DCL_MASK){   /* GEM manages form */

        case(QUIT):     break;          /* If QUIT, exit */

        case(FREQUP):                   /* If UP button pressed ... */
             if (cur_freq < MAX_FREQ){
                 cur_freq++;            /* ... increment frequency */
                 mod_display(cur_freq); /* ... and change display */
                 }
             break;
```

(Continued on page 80)

ATARI DEALER DIRECTORY

The following is a list of dealers who carry Atari products. Please patronize them and mention *Atari Explorer* when you do. Dealers: List your store here. Call Barbara Edwards at (201) 543-6007 for details.

METROPOLITAN NEW YORK

Park Avenue Video Center
260 Park Ave. S. (21st St.)
New York, NY 10010
(212) 505-0930
Authorized sales and service of Mega, ST, and 8-bit Atari computers. We also carry other major brands of peripherals. Best price, best service.

Island Software and Computer Service
35 Middle Country Rd.
Coram, NY 11727
(516) 736-1001

Authorized Atari Business Computer Center featuring the Mega line. We are the oldest authorized Atari dealer and service center on Long Island. Over 500 software titles in stock. Full line of accessories and peripherals.

Software Station
129 Rockaway Townsquare Mall
Rockaway, NJ 07866
(201) 328-8338
and
7 Headquarters Plaza Mall
Morristown, NJ 07960
(201) 455-7858
and
Princeton Forrestal Village
Princeton, NJ 08540
(609) 520-1212
Atari lovers, Atari owners, Atari software, Atari hardware. We go together at Software Station. The Atari lover's store. 8-bit and ST. 25% off software with this ad.

Video Home Center
336 Rte. 9 North
Manalapan, NJ 07726
(201) 431-7636
Atari 520, 1040, and Mega authorized sales and service. Full line of printers and other peripherals. Your one-stop Atari shop!

NEW ENGLAND

Soft Ware Haus
49 Beale St.
Quincy, MA 02170
(617) 770-3899
Full service Atari center since 1983. Atari 8-bit and ST hardware and software, accessories, and peripherals. Layaways available. Major credit cards accepted. 25% off list on software if you mention this ad.

HANDS ON COMPUTERS

99-B LARRABEE PLACE
WESTBROOK, ME. 04092

VOICE 207-854-1155
BBS 207-854-2607

Sales & Service Specialist for ATARI & COMMODORE COMPUTERS

Software for ALL MAJOR COMPUTERS *Supported*

ATARI

Computers Etc.
425 Kings Highway East (Route 1)
(½ mi. east of I-95, exit 24)
Fairfield, CT 06430
(203) 336-3100
Largest Atari ST dealer in the state of Connecticut. Voted one of "The Best of Fairfield County" in the *Fairfield County Advocate*'s annual reader poll. All ST software at least 20% off all the time.

The Computer Bug
113 Russell St.
Hadley, MA 01035
(413) 584-7722
(413) 586-1430 (BBS)
The Atari ST experts of Western Mass. Sales, service, and support. We offer solutions . . . not problems. Authorized on the complete Atari ST line.

Computer Cache
CC
273 Willard St.
Quincy, MA 02169
(617) 472-1502
Full line of Atari ST hardware and software. We also carry all other peripherals. 20% off software with this ad.

MIDATLANTIC STATES and SOUTHEAST

Elden Computers
1 Ramu Rd. at Toporock
Box 3201
Charleston, WV 25332
(304) 344-2335
Near C&P Telephone off MacCorkle.

Applied Computer Associates
Gaithersburg, MD
(301) 948-0256
and
Chantilly, VA
(703) 631-5377
aca
Atari ST sales and service. Major credit cards accepted.

McDonald's Computer Center
4921 E. Colonial Dr.
Orlando, FL 32803
(305) 894-2270

Central Florida's No. 1 Atari Business Computer Center. Authorized Atari Service. Complete line of software and accessories for Mega ST, 1040/520ST, and 8-bit.

The Home Computer Store
12008 Bustleton Ave.
Philadelphia, PA 19116
(215) 934-6999
Authorized Atari repair dealer. Full line of software and hardware for the 8-bit and ST. Visa MasterCard, and American Express accepted. Open Mon-Fri 12:30-9, Sat 10-6.

CompuVision Computer Centers, Inc.
Route 3, Old Mill Rd.
Northway Shopping Center
Millersville, MD 21108
(301) 987-0998
Full-service authorized Atari dealer specializing in the ST. Over 1000 software titles in stock.

Home Computers Co.
1055 Taylor Ave.
Towson (Baltimore), MD 21204
(301) 337-2733
SALES AND SERVICE
Atari 520ST and 1040ST systems • 130XE computers • Full line of disk drives, printers, etc. • Huge software library • Major credit cards.

MIDWEST

Software Plus
731 West Dundee Rd.
Wheeling, IL 60090
(312) 520-1717
and
2001 West Irving Park Rd.
Hanover Park, IL 60103
(312) 837-6900
and
6212 N. Western Ave.
Chicago, IL 60659
(312) 338-6100
1040, 520ST specialist. Authorized sales and service.

DigitalWorld Co. Inc.
711 Army Trail Rd.
Addison, IL 60101
(312) 543-9000

DigitalWorld

DigitalWorld has been a full-service authorized Atari dealer since 1981. We provide a total sales, education, and repair (in store) service environment for the Atari customer. Atari Mega ST dealer.

United Computer, Inc.
41818 Ford Rd.
Canton, MI 48187
(313) 981-1150
(313) 981-1680 (BBS)
We handle the full line of Atari hardware, software, and accessories. Authorized Atari and Panasonic repair center.

Mars Merchandising
1041B E. Saint Charles Rd.
Lombard, IL 60148
(312) 627-7462
(312) MARS-INC
and
Rockford, IL
(815) 633-8300

Blast off with our ST European imports. Manufacturer of 3rd drive and monitor extension cables. Astronomic 8-bit, 2600, and 7800 selection. Fabulous 2000+ 8-bit PD library. ST PD music library of over 4000 titles.
Visit our new store and try before you buy. Music demos, classes and repairs.

Fairborn Home Computer
2602 Col. Glenn Hwy.
Fairborn, OH 45324
Sales - (513) 429-3116
Service - (513) 429-8897
Full sales and service for Atari, Atari ST, and Mega systems.

Video Express
6016 E. Main St.
Columbus, OH
43213
(614) 866-2685

Computer Express
800 Bethel Rd.
Columbus, OH
43214
(614) 451-1440

Full range of Ataris: 2600, 7800, new Atari XE game machine, 8-bit, and ST. Software and hardware.

B and G Electronics Inc.
15729 Madison Ave.
Lakewood (Cleveland), OH 44107
(216) 228-PEEK (7335) (24-hr. BBS)
(216) 521-2855
Authorized Atari Business Computer Center. Full line of 8-bit and Mega ST software and hardware. Open Mon/Wed/Fri 10-6; Tues/Thurs 10-7, Sat 10-5. Major credit cards.

SOUTHWEST

The Floppy Wizard
217 Memorial City Mall
Houston, TX 77024
(713) 461-8660
Full-line authorized Atari dealer and repair station. In business since April 1983. Atari is our number one selling line in the store.

Computers to Grow
7046 Bissonnet
Houston, TX 77074
(713) 777-1673
Authorized Atari sales and service. Open 1-9 Mon-Fri, 1-6 on Sat, and 10-6 on Sun. Now in our fourth year of saving you up to 60% over inflated mall prices.

Computer Discoveries
12801 Midway Rd., Ste. 109
Dallas, TX 75244
(214) 484-9104
The premier Atari dealer for all of Texas. We specialize in the ST computer line and are now the ST headquarters for MIDI musicians. We also carry 8-bit hardware and software. Major credit cards accepted.

Info 1 Computers
N.W. 50th & Portland
Oklahoma City, OK 73112
(405) 942-7768
Authorized Atari ST and Mega Computer Sales and Service Center. We carry the full line of Atari hardware, software, and peripherals for the ST and Atari 8-bit line. Over 500 titles of domestic and imported software for the ST! We also carry Panasonic printers including the Laser Partner. Open weekdays 10-6, Sat 10-5.

Plaza Computers
San Mateo and Montgomery Plaza
Albuquerque, NM 87109
(505) 888-4885
New Mexico's leading Atari dealer. Authorized sales and service. Discount prices. Over 300 ST titles in stock.

Computer Works

4337 W. Bethany Home Rd.
Phoenix, AZ 85301
(602) 246-6364
Authorized Mega ST Business Computer Center. Sales - Service - Support. Large software selection. Low prices and quality service. Largest Atari dealer in Arizona.

Wedgwood Rental
5316 Woodway Dr.
Ft. Worth, TX 76133
(800) 433-2938
Software Rental: Atari, ST, IBM, C64, Apple. Call for free list. Thousands of titles. Major credit cards accepted.

Computer Skills
#66 Wilshire Village Shopping Center
Euless, TX 76040
(817) 267-5151
Your full blown Atari ST dealership! Always 400+ software titles in stock. 30% Discount Club. Software Rental Club. Hackers Club on Saturdays. Monthly raffles. User group discount always 25%! Authorized Mega sales and service. We install memory and 16MHz upgrades.

CALIFORNIA

Mid-Cities Comp/Soft
9406 Flower St.
Bellflower, CA 90706
(213) 867-0626/8994
Authorized Mega dealer.
Full Atari ST specialty store with over 500 titles in stock, including professional MIDI software. Also a wide range of 8-bit software and peripherals.

B&C Computervisions
3257 Kifer Rd.
Santa Clara, CA 95051
(408) 749-1003
Atari specialty store. Full line of software and hardware for both 8-bit and ST computers. Large public domain software library. Write or call for free catalog. Open 10-6, Tues-Fri; 10-5 Sat; closed Sun-Mon. Authorized Atari service center.

LTC COMPUTERS
2431 North Tustin Ave.
Santa Ana, CA 92705
(714) 667-1575

The complete Atari computer store. Authorized Atari ST and Mega computer sales and service. Southern California's largest and oldest Atari software and peripheral dealer. Open Mon-Fri 10-6, Sat 10-5.

COMPUTER·OUTLET
5945 Mission Gorge Rd.
San Diego, CA 92120
and
344 East H St.
Chula Vista, CA 92010
(619) 585-8100
Send for FREE! newsletter, the OUTLET. Always first on the West Coast with new ST products! Authorized Atari sales and service center.

Logical Choice for Computing
6116 Lankershim Blvd.
N. Hollywood, CA 91606
(800) 992-LCFC (toll-free)
(818) 760-0738 (in CA)
(818) 760-1018 (repairs)
(818) 760-0943 (BBS)
•Discount software & hardware •Laser printing services •Public domain software •Used hardware and software •Authorized Atari Service Center •Authorized Atari Business Center. "Your One Stop Atari Shop." Open Mon-Sat 11-7.

PACIFIC NORTHWEST

IB Computers
1519 S.W. Marlow Ave.
Portland, OR 97225
(503) 297-8425
Atari Business Computer Center. Full hardware & software selection for all Atari computers. Call for free price list.

XANTH
14100 N.E. 20th
Ste. 105
Bellevue, WA 98007
(206) 643-9697
(206) 562-0128 (BBS)
Full-service authorized dealer—Atari only.

Listing 2. (Continued)

```
        case(FREQDN):                    /* If DOWN button pressed ... */
             if (cur_freq > MIN_FREQ){
                 cur_freq--;             /* ... decrement frequency */
                 mod_display(cur_freq); /* ... and change display */
                 }
             break;

        case(MEMORY):                    /* If MEMORY button pressed ... */
             break;                      /* ... do nothing. */

        default:                         /* Otherwise, must be a preset, so ..*/
                                         /* If MEMORY button is SELECTED ... */
             if(panel[MEMORY].ob_state & SELECTED){
                  memory[ret - B1] = cur_freq;  /* change button's memory */

                        /* Deselect and redraw preset and memory buttons */

                  panel[ret].ob_state = NORMAL;
                  panel[MEMORY].ob_state = NORMAL;

                  objc_draw(panel,ret,0,
                                panel[BIGBOX].ob_x,
                                panel[BIGBOX].ob_y,
                                panel[BIGBOX].ob_width,
                                panel[BIGBOX].ob_height);

                  objc_draw(panel,MEMORY,0,
                                panel[BIGBOX].ob_x,
                                panel[BIGBOX].ob_y,
                                panel[BIGBOX].ob_width,
                                panel[BIGBOX].ob_height);
                  }

             else{                      /* Otherwise ... */
                  cur_freq = memory[ret - B1];   /* .. reset frequency */
                  mod_display(cur_freq);         /* and change display */
                  }
         }
   } while(ret != QUIT);

rsrc_free();                            /* Free resource memory */
appl_exit();                            /* Detach application */
}


/************************************************
           FUNCTION TO MODIFY DISPLAY
************************************************/

mod_display(value)
int value;
{
/* Sometimes stdio functions are convenient. In this case, we're using
   sprintf(), a variant of printf() that "prints" directly into memory
   to alter the display label string, pointed to by the ob_spec field
   belonging to the label object. The expression in quotes is a format
   specification that is interpreted: "Print the first argument as a
   right-justified decimal number of up to three digits, padded on the
   left with spaces, as necessary; then print the second argument as a
   one-digit decimal number." The first argument expression divides the
   frequency value by 10, an operation that, when performed on an int,
   lops off any decimal remainder (i.e., the rightmost digit), neatly
   framing the remaining digits as a 2 or 3-digit integer between 87 and
   108. The second argument expression rederives the remainder (i.e.,
   the rightmost digit) using the modulo function. */

sprintf(panel[DISPLAY].ob_spec,"%3d.%1d",value / 10,value % 10);
objc_draw(panel,DISPLAY,0,panel[BIGBOX].ob_x,   /* redraw display */
                               panel[BIGBOX].ob_y,
                               panel[BIGBOX].ob_width,
                               panel[BIGBOX].ob_height);
}
```

ATARI EXPLORER
THE OFFICIAL ATARI JOURNAL
To The Future:
WHAT'S AHEAD FOR ATARI?
Soul Of A New ST—Designing the Mega Machines

Atari Explorer

The User-Friendly Computer Magazine

Why did you originally buy an Atari computer? To do word processing? To compose music? To manage your business? To play games? Chances are, whatever your initial reason for buying an Atari, you've discovered that it has many additional capabilities and potential applications.

The flip side of the coin is that you've probably experienced some frustration as well. Maybe your word processing package won't do subscripts or underlining. Perhaps your database won't sort on as many fields as you need. Or, it could be that when you write a program, your whole system acts user-hostile.

Depending upon the balance between your satisfaction and your frustration, you may continue to use your computer or set it aside. But there is a better way: **Atari Explorer.**

As the premier magazine for Atari computer owners, it is our responsibility to make sure that you get the most out of your computer. To us, that means making sure your Atari does more than you bought it to do, more than friends' and associates' computers do, and more than you ever imagined a computer could do.

To make sure that you get the most out of your computer, **Atari Explorer** brings you objective, in-depth reviews of hardware and software; up-to-date information about new products; practical tutorials; stimulating columns; thought-provoking articles; and valuable inside information.

Hard-Hitting Evaluations

At **Atari Explorer**, we obtain new peripherals and software packages as soon as they are released. We put them through their paces in our on-site laboratory and also in the environment for which they are intended—home, office, lab, or school.

Our evaluations are unbiased and accurate. We are not afraid to call a spade a spade or a lemon a lemon. Our first obligation is to you, our readers, and editorial excellence and integrity are our highest goals.

Practical and Stimulating

We know that some of our readers are beginners and others are experts. Thus it is our responsibility to make what we publish both comprehensible to newcomers and interesting to veterans. That does not necessarily mean that the material is simple; we know you like to be challenged. What it does mean is that we provide the inexperienced user with every possible means to seize the subject matter and make it his own.

However, we don't want the experts to be bored, so although articles are accessible to beginners, they are theoretically non-trivial, cover topics in depth, and present information on more than one level.

At **Atari Explorer**, we are intensely interested in all aspects of computing. Ours is the magazine of pragmatic applications, communicative graphics, stunning animation, mind-expanding games, and realistic simulations. We take our business seriously, but we have fun too. We are convinced that you will, too, when you go exploring with the **Atari Explorer** family.

A Great Deal! Guaranteed!

To order your subscription to **Atari Explorer**, simply fill out the coupon and mail it to the address below. There is no risk; if at any time your are not completely satisfied, we will refund the unfilled portion of your subscription—no questions asked.

By subscribing now, you'll be on the inside track to Atari—the company, the people, and the products—you won't miss a single information-packed issue, *and* you'll save 16% to 25% over the newsstand price. Act today!

Send to: Atari Explorer
CN961
Netcong, NJ 07857

Yes! Please enter my subscription to **Atari Explorer** at the money-saving rate I have checked below.

	USA	Canada*	Foreign*
6 issues	□$14.95	□$19.95	□$24.95
18 issues	□$39.95	□$54.95	□$69.95

*Checks must be in US funds and drawn on a US bank.

My computer is: □ST □8-bit □Game System

Name ________________

Address ________________

City ____________ State ____ Zip ____

□ Payment enclosed □ Visa □ MasterCard

Credit card number ________________

Expiration date ______ Signature ____________

Send to: Atari Explorer, CN961, Netcong, NJ 07857-0961.

Introducing The Atarian Club. Only $15. For Atari® Owners.

Here's your chance to become a *charter member* of the only club that's just for Atari owners. Join the Atarian Club and we'll send you an Atari T-shirt, a custom designed sew-on patch, and an Atarian membership card.

Charter members also get a one year subscription (6 issues) to the Atarian Magazine, featuring stories on what's new, what's hot, previews of upcoming games, tips on how to max your scores, interviews with game designers, puzzles, jokes, and a whole lot more.

Atarians also qualify for *special discounts* on Atari merchandise, a special video game preview tape, and the opportunity to get FREE GAMES.

With Your $15 Membership You Get:

A T-shirt worth	$10
A Sew-On Patch worth	$ 2
A Laminated ID Card worth	$ 1
Six Issues of the Official Atarian Magazine worth	$12
Plus Discounts on Games, Contests, Prizes, Fun, Fun, Fun.	
Total Membership Value	**$25**

ATARIAN CLUB, 7 Hilltop Rd., Mendham, NJ 07945

Yes! I want to join the Atarian Club. Please rush my T-shirt, patch, and membership card, and enter my subscription to Atarian magazine.

Name ____________________

Address ____________________

City ____________ State ____ Zip ________

□ Enclosed please find my check or money order for $15.

□ My T-shirt size is (circle one): S M L XL

□ My game system is (circle one): 2600 7800 XE

My favorite games: 1. ____________

2. ____________ 3. ____________