SEASONS · GREETINGS!

ST LOG

THE ATARI ST MONTHLY MAG

FDC 50076

DECEMBER 1988

ISSUE 26

USA $3.95
CANADA $4.75

DUNGEON MASTER
PLAYER'S GUIDE

ST DATE PLANNER

INSIDE THE
ST XFORMER II

Reviews:
First ADD
Pala n
Le atf rneck
B bb Ghost

0 74369 50076 1

Some Call It A Refreshing Change

We Call It The FA-ST Hard Drive

We named our drive after its swift and aggressive behavior. But it's really not fair to limit this incredible peripheral to just one name.

Call it cool. Cool, calm and collected with its whisper-quiet fan to prevent heated situations.

Call it high-class. With refined style, its sleek design complements your Atari computer system. Quite simply, functional elegance under your monitor that's designed to adjust to your system and lift your sights for easy viewing.

Call it friendly. Our FA•ST Hard Drive welcomes a host of features like dual DMA ports which invite new devices. Our SCSI expansion is ready when you are. And inside, our drive can handle a partner like no others can. Have the time? The FA•ST drive does . . . the right time, everytime.

Call it durable. Unwavering dependability from a winning design. Only the best components are found inside our FA•ST Hard Drive. A full one year warranty and ICD's uncompromised reputation for quality should say it all.

Now, don't let the abundance of features scare you . . . FA•ST Hard Drives are available in all sizes *and* at prices you can afford. So, to be quite honest, we really don't care what you call our hard drive – as long as you call for it today. And get ready for the best thing that ever happened to your Atari ST.

Call or write for our free catalog today.

1220 Rock Street • Rockford, Illinois 61101 • (815) 968-2228 • MODEM: (815) 968-2229 • FAX: (815) 968-6888

ICD

CIRCLE #101 ON READER SERVICE CARD.

EDITORIAL

BY CLAYTON WALNUM

Cold. A bitter, crystalline cold that whirls with the wind, changing the air to glass, the sea to rock. Snow swirls across a plain of white, bringing with it delicate frost figurines that dance, spin, leap, then fall back to be lifted again into new, undulating shapes. And there is the song. The wind's song. It wails its lament into the arctic afternoon, alone, a single voice offered up from the storm's glacial throat.

A tiny stream of warmth trickles from the frost-painted glass of a window. The heat tumbles away, into the storm, into the gale, into the cold. There is a face in the window. Pink nose and cheeks, pointed ears and sad, aquamarine eyes. The elf rubs away some of the window's frost. He is the watcher. Every morning he posts himself by the window and spends the day gazing out over the sea, waiting, hoping, looking for that which has yet to come. In his heart there is doubt, a glowing coal of despair. Where are they? Why have they not come? Christmas is fast approaching!

Behind him there is a long table wrought from oak and trimmed with black strips of steel. One hundred creatures sit on benches—99 elves and one man. There is one empty place. It belongs to the watcher.

This is a feast: ten roasted turkeys, four buckets of stuffing, 150 sweet potatoes, 15 gallons of wine and two dozen pumpkin pies. But the eating has not yet begun. All wait for the man to make the toast, to officially begin the holiday season. Eyes flicker hopefully to the watcher, then back to the man. It would not do to start the holidays without that which has not come. In a few moments, the man will have to begin the feast anyway, and none can bear the thought of such a dark omen.

Where are they? When will they come?

The man glances behind him. The watcher turns and shakes his head. The man sighs, stands, lifts a glass of wine. The time has come; he can wait no longer. He forces a merry smile through a forest of snow-white whiskers. He laughs as he raises his glass above all their heads, his belly bouncing and quaking (it really is a great deal like a bowl of jelly). It's a wholly unconvincing laugh, even though he is doing his best to brighten the event.

"My friends," he begins, and there is no need to hush them since none has spoken since gaining his seat, "I once again take great pleasure—."

Pleasure, thinks the watcher. The empty words are little more than a background rumble, belying the man's effort at jollity. What pleasure can be left now? He blinks and rubs the window, trying to remove the speck of dirt interfering with his view.

But the rubbing does no good. The window is clean.

"Wait!" he cries, and the man stops the toast. All turn to look at the watcher. "I think—I think—."

The speck grows larger, and as it approaches, he can see a thin stream of smoke climbing into the sky.

"Yes, it's definitely a boat. This could be it!"

No one moves as the ship drifts into dock. No one speaks as the gangways are affixed. No one breathes as the cargo bays are opened. Finally, pallets of boxes begin moving toward the warehouse, toward the place where the man's still half-empty gift sacks await filling.

The watcher squints, leans forward. The forklifts roll closer.

"Well?" the man says. "*Well?*"

"I can't quite read the labels yet," the watcher replies. "Just a little closer—a little closer.—"

And then the waiting is over. Even through the snow he can see that the first pallet is marked "Atari Corp.," the second "ST-Log."

"Yes!" The watcher spins from the window, performs three backflips and plops neatly into his position at the table. "They're here!"

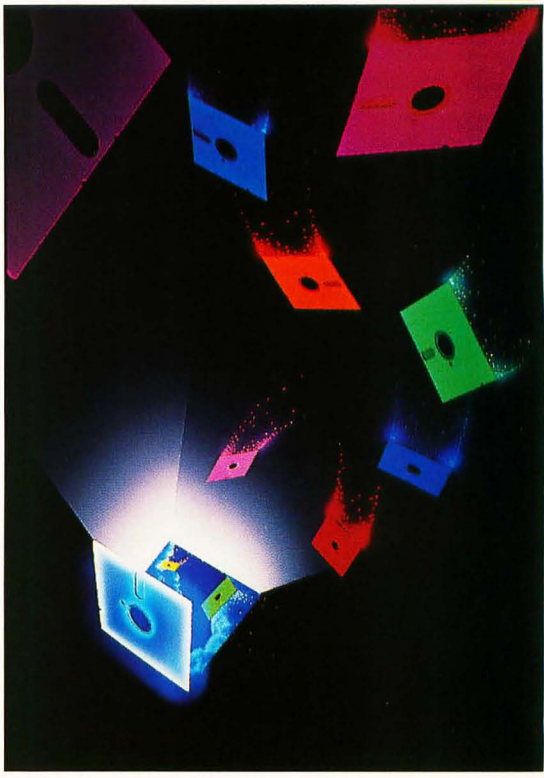
Cheers and pointed caps fill the air. The man raises his glass. "My friends, I once again take great pleasure in welcoming the new holiday season. Peace to you all!"

"Peace to us all!" 100 voices echo.

Outside, the storm abates, and the forklifts—bearing thousands of exciting gifts for well-behaved souls all over the world—trundle into the warehouse.

IN THIS ISSUE

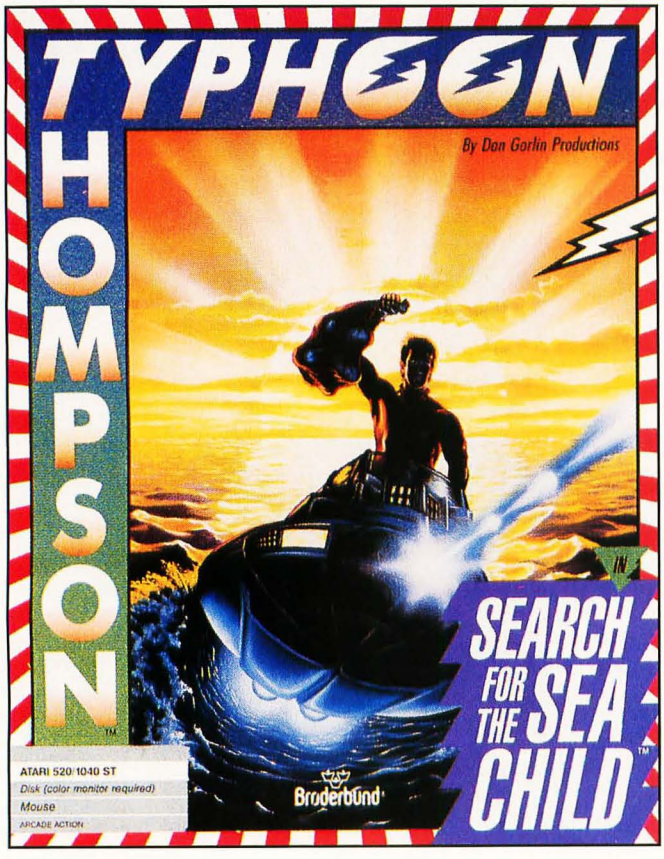
INSIDE ST Xformer II 18



FEATURES

REVIEWS

ST · USER 34



COLUMNS

DEPARTMENTS

PROGRAM · LISTING · GUIDE

ST-Log (ISSN 0890-9601) is published monthly by L.F.P., Inc., 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210.

U.S. subscription $28 for one year (12 issues), $52 for two years (24 issues), $76 for three years (36 issues). Foreign subscriptions: add $7 per subscription per year. Single copy $3.50 (add $1 for postage). Change of address: six weeks advance notice, and both old and new addresses are needed. POSTMASTER, send change of address to: **ST-Log** Magazine, P.O. Box 16928, North Hollywood, CA 91615. Second-class postage paid at Beverly Hills, CA, and additional mailing offices.

ST LOG
THE ATARI ST MONTHLY MAGAZINE

DECEMBER 1988
ISSUE 26

STAFF

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

ADVERTISING SALES

Correspondence, letters and press releases should be sent to: Editor, **ST-Log**, 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210.

Correspondence regarding subscriptions, including problems and changes of address, should be sent to: **ST-Log,** P.O. Box 16928, North Hollywood, CA 91615; or call (818) 760-8983.

Correspondence concerning a regular column should be sent to our editorial address (see Authors below), with the name of the column included in the address. We cannot reply to all letters in these pages; so if you would like an answer, please enclose a self-addressed, stamped envelope.

J.E. Publishers Representatives — Los Angeles: (213) 467-2266.
San Francisco: (415) 864-3252. Chicago: (312) 445-2489.
Denver: (303) 595-4331.
6855 Santa Monica Blvd., Suite 200, Los Angeles, CA 90038.
New York: (212) 724-7767.

Address all advertising materials to: Paula Thornton — Advertising Director, **ST-Log**, 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210.

PERMISSIONS

SUBSCRIPTIONS

ST-Log, P.O. Box 16928, North Hollywood, CA 91615; or call (818) 760-8983. Payable in U.S. funds only. U.S.: $28.00-1 year; $52.00-2 years; $76.00-3 years. Foreign: add $7.00 per year per subscription. For disk subscriptions, see the cards at the back of this issue.

COVER ILLUSTRATION
• BY FRANK HOWA

INTERIOR PHOTOGRAPHY
• STEVEN HUNT
• DON CARROLL
• ORION PRESS

AUTHORS

When submitting articles and programs, both program listings and text should be provided in printed *and* magnetic form, if possible. Typed or printed text copy is mandatory and should be in upper- and lowercase, with double spacing. If a submission is to be returned, please send a self-addressed, stamped envelope with your manuscript to **ST-Log**, P.O. Box 1413-MO, Manchester, CT 06040-1413.

ST GOSSIP FROM Hollywood USA

BY T.G.

I don't ever want to hear any of you question whether ST-Log is willing to spend big money to get a story. As a partial Christmas present to the rest of the office staff (I tend to hang around the office a lot during the holidays looking for invites to parties) and partially as an effort to pursue "the story," ST-Log recently sent me on a fact-finding mission to Paris. Now after seven days of grueling research at the corner tables of every bistro on the West Bank, I can report on some of the latest in ST hardware. Please note: The fact this stuff exists in Europe indicates that it may well be available in the U.S. at some time in the very near future.

. . . . . .

Now after seven days of grueling research at the corner tables of every bistro on the West Bank, I can report on some of the latest in ST hardware.

The big picture

Tired of looking at the world through a 12-inch picture tube? Well, *big* screen monitors are now on sale in Paris. I finally got to try one this week. It was set up in the window of a little computer shop just across the street from my hotel. When I asked the manager how it was selling, he shrugged (all Frenchmen shrug) and told me all the "Atari stuff" sold well. Most of the buyers were using the monitor for CAD work although some of his customers were graphic artists. I was given about a half hour with the unit during a quiet time in the early morning.

The 19-inch screen, with an awesome resolution of 1024 x 1024 pixels, was up and running on a Mega 4. This adaption to the ST consists of a connector from the Mega's internal expansion port which runs to the monitor. That means it will not work with Atari's new arithmetic coprocessor but. . . .

The driver for this screen is loaded as a desk accessory and was written by a local (French) company called Megavision. One of the authors is Dominique Laurent, the guy who wrote *TurboDOS*. The driver still has a few rough edges—but very few. For all you Techie types here are a few specs.

Screen type: 19-inch, paper white phosphor
Resolution: Up to 1024 x 1024 pixels
Scanning: Horizontal 48@ kHz
Vertical 60 Hz Non-Interlaced
Input: Video: TTL Positif
DB-9 pin (DIN)
Other Information: Software delivered as a Desk-Accessory; can be used as a "Switcher"; choice between pull-down or drop-down menus; menus can be called from anywhere on the screen; laser compatible for screen dumps.

This beauty will work with all GEM programs if they are written following the DRI rules. I can tell you from personal experience that it works with *Timeworks Desktop Publisher ST* and *Superbase Professional*, and I am told that the monitor is being used regularly with some of the better CAD packages in Europe now. There are a few programs that it does seem to have problems with, however. The only one you might have heard of in the U.S. is *1ST Word Plus*. It seems like the graphics mode of this popular word processor gets the screen a little messed up—usable, but messed up.

This monitor offers more than just large, sharp pictures. After installing the monitor software, you can use the large screen for your main work area and use the smaller Atari SM124 monochrome monitor to display a "Zoom Mode" of small sections of the large monitor's screen. This unit is clearly designed as a professional tool and is priced as such: $2,600, U.S. That's a week's stay at the deluxe hotel I've been hanging out in here in Paris! The value of the U.S. dollar is really working against the import of some of the more exciting products I've seen. Among the other things you may see in the coming months is. . .

Handy Scanner

You've seen one form of these advertised in some of the business magazines. It's a small device that looks a bit like an old style electric razor and functions as a personal copier. You simply run it across a page and a photocopy (on a thin strip of paper) of whatever the four-inch head passed over is produced.

Now, adapted to the ST and connected via the cartridge port, you can pass the head of the unit across an image, and the image is stored in your ST's memory. You can then save the picture in any of several formats for later use in desktop publishing or any other graphics application that will benefit from scanned input. The image quality is better than that which you get from *Image Scan*, and you don't need to run your original through a printer with a scanner head attached to achieve that quality. Once again the cost is high for the home user and quite reasonable for the office environment: $400 U.S.

Meanwhile, back at the ranch. . .

Back on the home front, word is that Atari is making good progress on its own full-page monitor, and it should see the dealer's shelves within the next 30 to 60 days. This unit is an Atari product and will, of course, sell for much less than the big-screen job we talked about above.

In addition, in an effort to build its DTP market position, Atari has been guiding the rewrite of GDOS into a potential powerhouse. The specs for the new GDOS (under development for over six months now) call for it to be able to: 1) load fonts dynamically; 2) use outline-defined fonts rather than bit-mapped fonts; 3) require only one size (definition of font) to be loaded for each style to be used, with the ST doing all the rescaling internally at the time of display or printing; and 4) provide 100% compatibility with all programs using GDOS.

If these four objectives are met, GDOS will move from a memory-hungry patch to a lean, mean DTP tool. To take advantage of the new GDOS, Atari has licensed over 40 new fonts, to be released with the new version of GDOS. When? Well, this is the big question at Sunnyvale. Lately, Atari has shown a willingness to take the time necessary to develop a bug-free version of a product before shipping, rather than rushing something to market and letting the public do its debugging. It's a more professional attitude, but it does mean that all we can tell you is that you'll see it when it's ready and not before.

Unix, MS-DOS emulation, GEM and a 68030 all in one box—tune in next month for details. . . .

Wine, women, song

After spending seven days in Paris, I can honestly say the wine in California is a better buy and in many cases just plain better. Speaking of Hollywood, unless memory fails me, Warner Bros. still has a financial interest in the success of Atari Corp. Has anyone from Atari's ad agency suggested to the Warners people that it would not hurt to show STs in a film now and then? We all see movies and TV shows with computers in the background every day. How about using some Ataris for props? The people who own one will be the only ones who recognize it, but the unit is unique looking, and the image will impress itself on the mind of the potential computer buyers who might just get a positive feeling from the ST the next time they visit a computer store displaying the unit.

TG can often be found skulking the turf around Hollywood and Vine. However, he won't be going to Paris again anytime in the near future, mostly due to the fact that the Parisian police have made it very clear that they don't want him back—something to do with the police chief's daughter. Heard any good Atari rumors? Write it down and stick it with used gum on the underside of the pay phone at the address above. (Don't live in La-la land? Then send TG's mail to: ST-Log, 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210.)

HOWARD MEDICAL COMPUTERS

1690 N. Elston • Chicago, IL 60622 • ORDERS (800) 443-1444 • INQUIRIES AND ORDER STATUS (312) 278-1440

★ 5 STAR FINAL NOV'88 **CLEAR**

HMC SAVES BUDGET

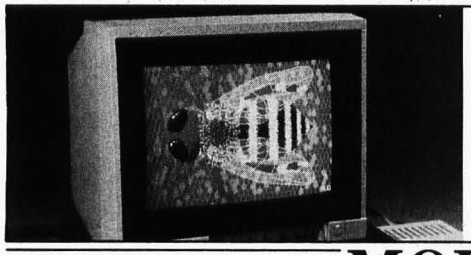
MONITORS

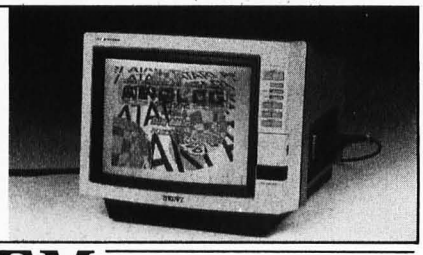
MAGNAVOX color **8CM515** (14 ship) **$269** 3 inputs offer maximum versatility for this 13" color monitor; RGB Analog for the ST; composite for the 800/XL/XE or VCR and RGB TTL for IBM and Clones. The 640 × 240 resolution with .42 MM dot gives vivid colors and a green screen override for 80 column text. ST RGB cable $19.95 with monitor or $29.50 separate.

MAGNAVOX 8CM 505 $220
13" Color Monitor 390 × 240 resolution

SONY color **KV1311CR** (15 ship) **$499** The Ultimate 13" color monitor includes a trinitron TV with remote control; 640 × 240 resolution with a .37 MM dot and a Micro Black screen that is vertically flat for distortion free viewing. Four inputs includes all of those from the Magnavox 515 plus a digital RGB. The colors are incredibly brilliant.

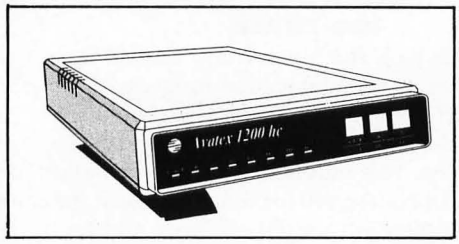
ZENITH green **123A** (7 ship) **$67.50** This close out from Zenith features a 12" non-glare composite Amber screen with 640 × 240 resolution. 90-day warranty valid at our 1200 locations. cable **$19.95**





MODEM

AVATEX 1200 HC (2 ship) **$119.95** This Avatex modem is fully Hayes compatible and operates at 300 or 1200 baud. We include express 3.0, a public domain communications program free with each modem.



DRIVE A+ ($178.45)

Howards Drive A+ features our DB-1 drive box and our DD-3 MPI double sided double density, 40 track full height drive with case and power supply for full 360K storage.

DB-1 DRIVE BOX ($85.00)

Howards drive box hooks up to a ST 1040 to give 2 external drivers. A 3 pole switch will configure an 80 track DSDD (double sided double density) drive or a 40 track DSDD drive or the new 3½ drive. Use with PC Ditto for IBM software compatability.

SIO ($64.45)

IDC's PR. Connector gives 2 RS 232 ports for a modem or serial printer and 1 parallel port for a parallel printer like EPSON or STAR. Ask for our special price on cables.

SPARTA DOS COMBO ($50.00)

This combo gives you IDC's SPARTA DOS OPERATING SYSTEM and the 2 chips ROM set that allows double sided access for the 1050's.

HMC's Guarantee— A Promise you can take to the Bank.

Howard Medical's 30-day guarantee is meant to eliminate the uncertainty of dealing with a company through the mail. Once you receive our hardware, try it out; test it for compatibility. If you're not happy with it for any reason, return it in 30 days and we'll give you your money back (less shipping.) Shipping charges are for 48 states. APO, Canada and Puerto Rico orders are higher.

hotline number

DON'T MISS OUT, ORDER TODAY!
800 / 443-1444
WE ACCEPT VISA • MASTERCARD • AMERICAN EXPRESS • C.O.D. OR CHECKS DISCOVER • SCHOOL P.O.

CIRCLE #102 ON READER SERVICE CARD.

ST NEWS

TOS 1.4 Release

Atari's registered developers have been offered an update kit to the ST Developer Kit. The package of revisions and additions includes a floppy disk version of TOS 1.4, Atari's latest edition of the low-level ST operating system. The update kit also includes a new object-level debugger. Bug fixes for Atari's assembler, *MadMac*, are also included.

Developers were asked to pay $20 to cover postage for the update kit, which includes several new documents. One of the documents describes the new SFP004 Mega ST floating point math coprocessor board for the Mega ST expansion slot.

With the developer release of TOS 1.4, the ST user community is that much closer to being able to buy the new operating system.

"Atari's registered developers have been offered an update kit to the ST Developer Kit."

Atari Corp.
1196 Borregas Avenue
Sunnyvale, CA 94086
(408) 745-2000

CIRCLE #103 ON READER SERVICE CARD.

WordPerfect $155

The most expensive word processor for the ST, *WordPerfect*, is now being offered to members of Atari user groups in the United States for only $155, a savings of $174 from the normal list price of $329. Although the offer is only available for a limited time, the lower price is supposed to make *WordPerfect* affordable to the dedicated Atari user who has not been able to purchase the mammoth word-processing package since its release a year ago.

To qualify for the special price, a user-group purchase agreement must be submitted to WordPerfect Corp. Information about obtaining the necessary paperwork can be found by contacting WordPerfect Corp.

WordPerfect Corp.
1555 N. Technology Way
Orem, UT 84057
(800) 321-4566

CIRCLE #104 ON READER SERVICE CARD.

Miniature ST golf

Family fun comes to the ST in a new miniature-golf game from Digitek Software, the company that brought *Vampire's Empire* to the ST. *Hole-In-One Miniature Golf* comes with four complete miniature-golf courses on two ST diskettes. From the classic windmill to the inside of a pinball machine, Hole-In-One provides a multi-level golf game that is suitable for game players of all ages.

Hole-In-One Miniature Golf has a list price of $29.95 and is now available.

"Family fun comes to the ST."

MINIATURE ST GOLF

DigiTek, Inc.
10415 N. Florida Ave.
Suite 410
Tampa, FL 33612
(813) 933-8023

CIRCLE #105 ON READER SERVICE CARD.

The Magic Sac saga continues

David Small is working on a new version of the *Magic Sac*, the Macintosh emulation system for the Atari ST. The new system, *Spectre 128*, has two major differences from the old *Magic Sac*: it works with the newer 128K Macintosh Operating System ROMS, and it is not being marketed by Data Pacific.

In 1986, David Small teamed up with Joel Rosenblum, an old friend who was also a computer programmer, to start Data Pacific, the company that would handle marketing and sales of the Magic Sac. Small and Rosenblum parted last April, with Small announcing the commencement of the Spectre 128 project and Rosenblum continuing to market the Magic Sac.

Spectre 128, expected to be out in the first quarter of 1989, will open up a huge library of Macintosh software to ST users. Since the release of the Macintosh Plus in 1987, most Macintosh software developers have taken advantage of the new operating system functions built into the 128K Macintosh ROM. *Hypercard* relies heavily on the new ROM set, as do most of the other commercially available products.

Gadgets By Small, Inc.
40 Littleton Blvd #210-211
Litteton, CO 80210
(303) 791-6098

CIRCLE #106 ON READER SERVICE CARD.

Speed Your FAT

The File Allocation Table (FAT) is a list of free disk space on your ST's hard-disk drive. While most people aren't aware that the FAT even exists, once you fill your hard disk with more than a few files, you will find the performance of your hard disk to be greatly impaired. The problem lies in the method TOS uses to find free space to store information on your hard disk.

Fat Speed is a public-domain utility program that patches TOS to use a more efficient method of reading the FAT. The small utility program is added to your auto folder and, when your ST is powered on Fat Speed, invisibly modifies the method TOS uses to search through the FAT.

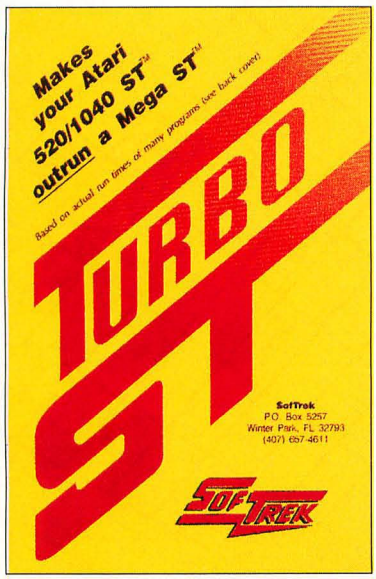
Fat Speed works with most versions of TOS ROM chips, but the author makes a clear disclaimer that use of Fat Speed can sometimes permanently erase the FAT, making it impossible to access any of the files on your hard disk.

You can find Fat Speed on DELPHI, GEnie and CompuServe. A small document is included that gives some particulars of usage and an address where the author may be reached.

Turbo ST Version 1.2

Softrek has introduced an update to its popular *Turbo ST* utility program for the ST. Turbo ST patches the GEM operating system to provide quick methods of drawing text to your ST's screen. The result is a noticeable speed improvement in GEM programs that rely heavily on GEM to draw text. For example, Timework's *Word Writer ST* scrolls text up to 50% faster when *Turbo ST* is in use.

Older versions of Turbo ST had compatibility problems when using certain programs. The new version 1.2 solves 90 percent of the compatibility issues and has been tested with *Flash*, *VIP Professional*, *Zoomracks II*, Beckemeyer's *MT C Shell*, and *Interlink*.



The new version also includes a fast VDI filled-rectangle routine. Almost every function of GEM relies on filled rectangles. For example, drawing windows, menus, scroll bars, and other objects require rectangles to be plotted to your ST screen. Turbo ST 1.2 greatly improves GEM's speed, serving as a software blitter chip.

Registered users can obtain an update to *Turbo ST 1.2* by returning the original program diskette plus one dollar to Softrek. A new manual is included in the upgrade fee.

Softrek
P.O. Box 5257
Winterpark, FL 32793
(407) 657-4611

CIRCLE #107 ON READER SERVICE CARD.

Software directory '89

If you have ever attended a computer trade show when Atari Corp. is in attendance, there is a good chance that you will have heard Atari's estimation of how many software products exist for the ST computer. The number seems to change from person to person, with estimates ranging from a few hundred to many thousands. Although Atari Corp. frequently advises software developers to send library copies of new products to Cindy Clavern, Atari's librarian and developer relations coordinator, it is really anyone's best guess to determine the number of products available to ST users.

For the past three years, Andy Nicola has maintained a list of ST software products. The latest release of the software list includes 2,000 entries of commercially and publicly available programs for the ST. The software list is updated as often as new products are brought to the attention of Nicola, sometimes as often as twice a week. The software list is purposefully kept off of CompuServe, DELPHI, GEnie and BIX, as Nicola does not believe

"The list includes prices, availability dates, descriptions and review publication dates, and can be organized by category, publisher, or title."

the information should be sold. Instead, distribution of the list is limited mostly to public bulletin-board services.

The list includes prices, availability dates, descriptions and review publication dates, and can be organized by category, publisher, or title. For information on obtaining a copy of the software list, Nicola can be contacted directly.

Andy Nicola
5143 Devon Drive
N. Olmstead, OH 44070
(216) 777-2532.

CIRCLE #108 ON READER SERVICE CARD.

New GEM file selector

The *Universal Item Selector* replaces the klunky file selector that appears when GEM-based programs need to select disk files. ST users that are familiar with the GEM Desktop are often plagued by the limited nature of the normal GEM file selector. For example, when attempting to load a file

"ST users that are familiar with the GEM Desktop are often plagued by the limited nature of the normal GEM file sector."

from a different disk drive or folder, the usual GEM item selector forces the user to enter a number of clumsy keystrokes and mouse clicks before the correct file directory is displayed. The Universal Item Selector sports many icons to easily move between disk drives and folders, and supplies many other functions not available from the standard GEM file selector.

Originally introduced in 1987, a major upgrade has just been released. *Universal II* supports many new functions, yet it occupies less than 23K of your ST's memory. The new version allows groups of files to be copied, deleted and locked. New dialog warnings have been added to functions that could remove or change files. The new system also supports the *Twister* disk format—a system of initializing a floppy diskette that also speeds disk access—on Mega ST computers. Universal II also comes with a printed manual. (The older version included the manual recorded on the program diskette.)

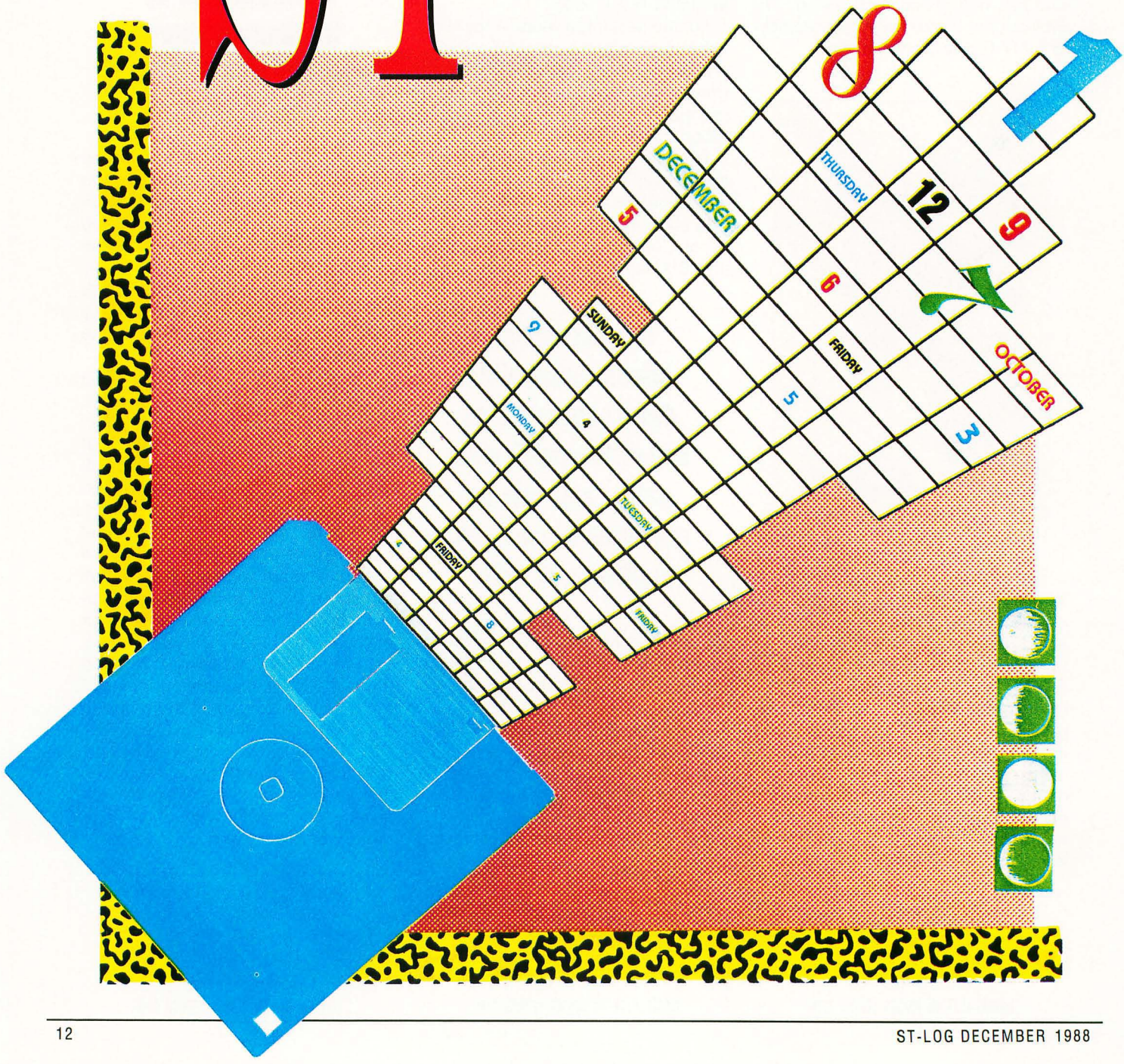
Universal II has a list price of only $19.95 and is available now. Upgrades to the new system are available to existing users for only $4 when the original diskette is returned.

Application & Design Software
226 NW "F" Street
Grants Pass, OR 97526
(503) 476-0071
12N-5PM PST Weekdays

CIRCLE #109 ON READER SERVICE CARD.

THE ST DATE PLANNER

by David Plotkin



The *ST Date Planner* will generate a calendar for any month and year, print that calendar out and allow you to enter appointments for any day of the month. It is completely GEM-driven and easy to use.

Listing 1 is the GFA BASIC source code for ST Date Planner. If you do not have GFA BASIC, both the source code and the GFA BASIC run-only program (a program that allows people who don't own GFA BASIC to run programs written in that language) are available on this month's disk version or in the databases of the ST Users' Group on DELPHI.

Using ST Date Planner

When you first start the program, the calendar for the current system month and year will be presented on the screen. This will be November 1985 for those without clock cartridges. To set any date, drop down the Selection menu and click on Select Date. A window will open in which you can type in the year you want. If you just press Return, the currently selected year will be used. Next, click on the month you want. The calendar for that year and month will appear.

To enter appointments, click on any day of the on-screen calendar. A window for the day you choose will open, providing you a form on which to enter your appointments. Each half-hour time slot from 7:00 a.m. to 5:00 p.m. has a line, and you may type in just about anything you choose. The cursor arrows will move the cursor around the window, as will the mouse pointer. The backspace key moves the cursor to the left, erasing each character it passes over. The return key moves to the beginning of the next line.

Editing is done in "replace" mode; so just type over any mistakes you make. The delete and insert keys also work. Pressing the "Clr" key will clear the line on which the cursor rests. There are also three "buttons" in the window. The OK button returns you to the calendar. (You may also press the escape key to exit to the calendar.) Click on the same day again, and you will find that your entries are still there. However, you must save the file before selecting another month and year (and generating another calendar), or you will lose your entries. The Print button will print the current day's appointments on the printer.

(Programming note: Notice the strings *Bon$* and *Boff$* in the procedure *Initialize*. These are the printer codes for Epson and Epson-compatible printers that turn on and off bold printing. Substitute the codes from your own printer here if they are different.)

The Clear button will clear all the appointments entered for the current day. You are not asked to verify this selection, so be careful about choosing it! After you return to the main calendar, you will notice that any day where you have made at least one entry will be marked with an asterisk.

The File menu is for saving and loading calendars. Save File is important, since all data is lost whenever you select a new month or year. If you have chosen Select Date and realize that you haven't saved your file, just choose the same month and year you were working on, and the data will still be there. When you save a file, the program will prompt you for a filename, using the standard GEM file selector box. It will even construct a filename for you to use if you wish, consisting of the month and year.

Load File will load a file of previously saved data, and the calendar for the month and year of the loaded file will be presented after the file is loaded. Again, remember that you must save your data before you load a new file or you will lose the old data. Save Screen will save the current screen in DEGAS format. This is so you can embellish the screen or use DEGAS' printouts to get better results. Quit will take you back to GFA BASIC.

The Print menu allows you to Print File (dump all of the days for the currently selected month to the printer) and Print Screen, which will send a copy of the calendar to the printer. To allow for different types of printers and for daisy-wheel printers without graphics, I have kept the printing simple and used only widely available characters. You can replace the characters, if you like, with something that looks better on your printer.

The Program

GFA BASIC is a remarkably easy language to program in, both due to its power and because it is interpreted. If you just want to use ST Date Planner, you can stop here; but if you would like to learn some GFA BASIC programming tricks, read on.

Windows, Dialog Boxes and the Mouse

GFA BASIC has two windowing systems, and the beginning of the program demonstrates the more versatile of them. The built-in system allows you to set the center-point on the screen, then open up to four windows—one in each corner of the screen, with the center-point determining the size and shape of the four windows. This system also allows you to use such commands as FULLW, which calls up a full-screen window.

The second windowing system uses a table of window properties. The table starts at WINDTAB (a predefined variable). The first ten positions of the table (WINDTAB

TABLE 1

Window 1	Window 2	Window 3	Window 4	Property
WINDTAB	WINDTAB+12	WINDTAB+24	WINDTAB+36	Handle
WINDTAB+2	WINDTAB+14	WINDTAB+26	WINDTAB+38	Attributes
WINDTAB+4	WINDTAB+16	WINDTAB+28	WINDTAB+40	X-pos
WINDTAB+6	WINDTAB+18	WINDTAB+30	WINDTAB+42	Y-pos
WINDTAB+8	WINDTAB+20	WINDTAB+32	WINDTAB+44	Width
WINDTAB+10	WINDTAB+22	WINDTAB+34	WINDTAB+46	Height

to WINDTAB + 10) define window 1, the next ten (WINDTAB + 12 to WINDTAB + 22) define window 2, and so on. The table is constructed as follows:

Each item in the table consists of two bytes. The "handle" is the GEM identification for the window, and you would not normally mess with this (unless you like system crashes). Attributes set window properties such as Full Box, Title Box, Size Box, Sliders and such. The new version of GFA's documentation shows which numbers correspond to which attributes. The X-pos and Y-pos refer to the upper left corner of the screen, and the width and height determine the size of the window in pixels.

The two windowing systems do not get along well. If you use the built-in system, you may not use the WINDTAB system because you will get a system crash. So if you need to overlap windows, use the WINDTAB system, as I have in this program.

GFA BASIC has no easy way to use Dialog Boxes. By using a small window, you can simulate a Dialog Box, enabling the user to type in information and click on buttons. When you open a window, GFA automatically makes such commands as *Print at* work relative to the upper left corner of that open window. Further, the mouse information commands *Mousex* and *Mousey* return coordinates which are relative to the upper left corner of the window, not the screen. This is convenient when you don't know exactly where the window will be, such as when you allow the user to move the window. Since opening one window on top of another will erase what is underneath the new window, you can temporarily store a picture of the hidden portion by using the GET command; then PUT it back after you close up the top window.

The construct DO ON MENU LOOP is one of the most powerful in GFA. To use it, you would previously write procedures which define what you want to do when the user clicks the mouse button, moves the mouse in or out of a predefined rectangle on the screen, selects a menu item, presses a keyboard key or takes some action which causes GEM to send a message. These messages are such things as clicking on the CLOSE box, moving a slider or resizing the window. The new GFA documentation tells you how to retrieve the appropriate information for each type of message. After writing the procedures, you would have a series of lines such as:

```
ON MENU GOSUB RESPONDMENU
ON MENU KEY GOSUB RESPONDKEY
ON MENU BUTTON 1,1,1 GOSUB RESPONDBUTTON
```

These lines tell the program that you want it to branch to the routine when the proper action is taken. Then you just put the program in the DO LOOP shown above and wait for the user to do something. This is exactly how ST Appointment Calendar works (check the listing).

There is one difficulty with this, however. Once you have turned these commands on, there is no way to turn them off. For example, let's say that if the user presses the mouse button, and the program branches to RESPONDBUTTON, you temporarily want the program to ignore keyboard input even after the program returned to the DO ON MENU LOOP. The way to accomplish this is provide an empty procedure and reroute the keyboard command to this procedure (which does nothing except return):

```
ON MENU KEY GOSUB DUMMY

PROCEDURE DUMMY RETURN
```

Nothing more elaborate is needed.

When a mouse button is pressed, the program needs to respond. The procedure *Menubutton* reads the system variables *Menu(10)* and *Menu(11)* to get the location of the mouse pointer. It then converts the coordinates of the mouse pointer to figure out which day was selected, and then brings up the appointments for that day. This gets a bit complicated because the mouse pointer coordinates are relative to the upper left corner of the screen. (I could have used Mousex and Mousey, which are relative to the upper left corner of the window, but I didn't know about that at the time.) The conversion depends on whether you are running the program in high or medium resolution.

What's on the menu?

The main action of this program is its response to menu selections. This is handled by the procedure RESPONDMENU. It tests to see which menu item was selected, then takes the appropriate action. The menu choice is returned as a number (*Menu(0)*). This number is an index into the string array that holds the menu items, and can thus be compared with each item in the menu:

```
IF Strip$(Menu(0))=
" Select Date " THEN....
```

Notice, though, that you need to match exactly the elements of the string, including any leading or trailing spaces.

One of the more interesting menu selections is the one that saves the calendar as a DEGAS format file. These files have 34 bytes at the front, followed by 32000 bytes of picture data. Saving the picture data is easy, you can do it in one command:

```
BPUT #1,Xbios(3),32000
```

You must have previously opened channel 1 for output. *Xbios(3)* returns the address of the screen, and 32000 is the number of bytes to save. You can read the picture data just as easily:

```
BGET #1,Xbios(3),32000
```

The 34 bytes mentioned earlier are arranged as follows: The first two bytes are the resolution. The resolution is returned by Xbios(4) and is 0 for low resolution, 1 for medium and 2 for high. To put this information into the first two bytes of a string which will be written to disk, you must first make sure the string is empty and available:

```
DEGAS$=STRING$(34," ")
```

Then, you put the bytes in:

```
MID$(DEGAS$,1,1)=CHR$(0)
MID$(DEGAS$,2,1)=CHR$(Res)
```

The other 32 bytes are the color palette. You start with element 3 of your string (DEGAS$ in this example). You can then DPEEK the hardware registers that hold the palette data. These registers run from &HFF8240 to &HFF8260. Of each two-

byte number, only the lower three nybbles are valid—the highest four bits contain garbage. Thus, you need to mask the high four bits:

```
Dumn=(DPEEK(&HFF8420) AND &HFFF)
```

Then you need to turn the result into two one-byte numbers:

```
Dumhi=(Dumn AND &HFF00)/256
Dumlo=(Dumn AND &HFF)
MID$(DEGAS$,3,1)=CHR$(Dumhi)
MID$(DEGAS$,4,1)=CHR$(Dumlo)
```

Once you have loaded all the palette colors into DEGAS, you can just write it out to the disk:

```
BPUT #1,VARPTR(DEGAS$),34
```

and retrieve it just as easily:

```
BGET #1,VARPTR(DEGAS$),34
```

One of the things we don't do in this program, but you might want to do, is to reload the palette data and picture. I've shown you how to put the picture data back on the screen, but you can't just DPOKE the palette data back into the memory; it won't work. (They are hardware registers, which can be read, but not written to.) The first thing is to pull the character data out of DEGAS$ and back into numbers:

```
Dd$=MID$(DEGAS$,3,1)
Dumhi=ASC(Dd$)
Dd$=MID$(DEGAS$,4,1)
Dumlo=ASC(Dd$)
Dumn=Dumhi*256+Dumlo
```

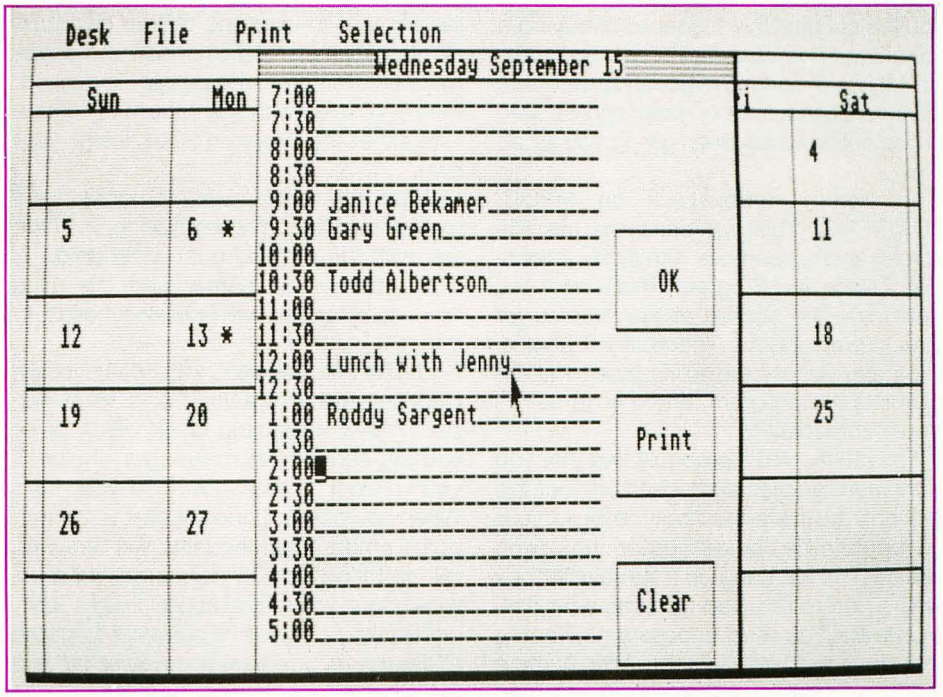
Dumn now contains all the color data for one color register, with the red, green and blue data in each group of four bits (the high four bits are empty). This is exactly the format for one of the forms of the SETCOLOR command; so you can now set the color:

```
SETCOLOR 0,Dumn
```

Faking the cursor

When you open an appointment window on the screen, you need to do several things. First of all, of course, you need to allow the user to enter information from the keyboard. But you must also respond to clicks of the mouse button, so that the user can indicate a cursor position and click on one of the three buttons. This presents something of a problem. The normal mechanism for getting user input from the keyboard would be with something like the INPUT command, which provides a cursor and full-editing capabilities on the chosen line. However, while the program is waiting for INPUT, it won't respond to mouse clicks. Using INPUT also precludes moving up and down the window with the arrow keys.

"The scan code is how the ST distinguishes between such keys as the numbers at the top of the keyboard and the numbers on the numeric keypad."



The way to accomplish fully flexible input is to once again use DO ON MENU LOOP and respond to mouse clicks and keystrokes. When a key is pressed, you can see what key it was by looking at *Menu(14)*. *Menu(14)* is actually a two-byte number with the ASCII code of the selected key in the low byte and what is called the scan code in the high byte. The scan code is how the ST distinguishes between such keys as the numbers at the top of the keyboard and the numbers on the numeric keypad. You also need to look at the scan code for such keys as the arrow keys.

Once the program has figured out which key was pressed, it can act on that information. For example, if a letter key was pressed, then the letter can be printed on the screen. If an arrow key was pressed, you can move the position of the cursor. Just one little problem with that—there is no cursor! The program can easily keep track of the "cursor" position and print information in the correct place, but the cursor itself really doesn't exist and is not visible. This makes it very hard for the user to keep track of things.

The answer lies in the ability of GEM to print letters on the screen in a mode called "reverse transparent," which looks like nothing so much as a cursor. Say, for example, the left arrow key was pressed. What you do is print the proper character in the old position of the cursor in regular (or "replace") mode, move the cursor one space to the left and print the character in that position in reverse transparent. This is accomplished by the Procedure *Prntit*. GFA has no easy way to switch printing modes; so we resort to generic GEM calls, just like ST BASIC:

```
DPOKE Contrl,32
DPOKE Contrl+2,0
DPOKE Contrl+6,1
DPOKE Intin,4
```

When you are done with reverse transparent mode, you can switch back just by:

```
DPOKE Intin,1
```

There is one final thing you should know about setting an on-screen character to reverse transparent. You must erase the character first (while still in replace mode, just print a space in its place), switch to reverse transparent mode, *then* print the character. Otherwise, instead of a character printed in reverse, you will get only a solid rectangular cursor. ■

by Kirk Stover

If you spend more time working on your computer than watching television, you may be missing out on a new social phenomenon—the *Wheel of Fortune* craze. Well, fear no more! America's favorite game show is now playing on your Atari 520ST. With *Mouse of Fortune* you too can feel the suspense of buying a vowel, and know the thrill of solving a puzzle. Mouse of Fortune may also serve to lure to your side that anti-computer spouse, parent or sibling who's been giving you a bad time.

To begin, double-click on MFORTUNE.PRG. This will bring up the title screen and an alert box with three options: File, Create and Program. This is how you determine the source of the puzzles for your current game. The outlined choice, Program, will use the 100 puzzles that are built into the program, and the game will begin immediately.

For variety, you'll probably find that you want more puzzles to choose from. For this reason, you have the ability to build your own puzzle files by clicking on the option Create. This will bring up a file selector box where you're required to name your new puzzle file. You must choose a new name that does not already exist. After naming the file, you will be presented with a screen which will ask for each topic and its corresponding puzzle individually. The topic can contain ten alpha and space characters, and the puzzle itself can contain 26 alpha and space characters.

Any topics or puzzles you input which do not meet these criteria will not be accepted into the file. You then repeat this process for as many puzzles as you can think of, with a maximum of 100 per file. To end the create mode, type in END when asked for the next topic. Your new puzzle file will be saved to disk, and you'll be returned to the title screen. To load one of your own puzzle files, click on the File option and specify, when prompted, the puzzle file desired. Now you're ready to play Mouse of Fortune with either the program's puzzles or your own.

When the game starts, the screen will display the randomly selected topic at the top, with the blanked out puzzle below it. It's a two-player game, and the blue "mouse of fortune" will indicate whose turn it is.

Players are presented with three choices: Choose Letter, Buy a Vowel and Solve Puzzle. There's a beeping clock which gives you ten seconds to make your choice. If you run out of time, you lose your turn. When you opt for Choose Letter, a random dollar amount that the letter will be worth will be displayed. As well as getting a letter amount, you may also get "Lose a Turn" or "Bankrupt." If you hit bankrupt, you lose all the money you have accrued so far, and the other player gets the turn. As long as you get a dollar amount, a letter board will be displayed for you to choose from. Just click on the letter you want to try for.

If the letter you've chosen is in the puzzle, all occurrences of it will be turned over, and you'll receive the dollar amount it is worth multiplied by how many times it appears. If you choose a letter that is not in the puzzle, you lose your turn. Watch the blue mouse-of-fortune. It always knows whose turn it is. Once a letter has been chosen, it will be blanked out on the letter board.

In order to buy a vowel, you must have at least $250. Once you click on that option all available vowels will be displayed and $250 will be deducted from your account. If you buy a vowel that's not in the puzzle, you lose your turn.

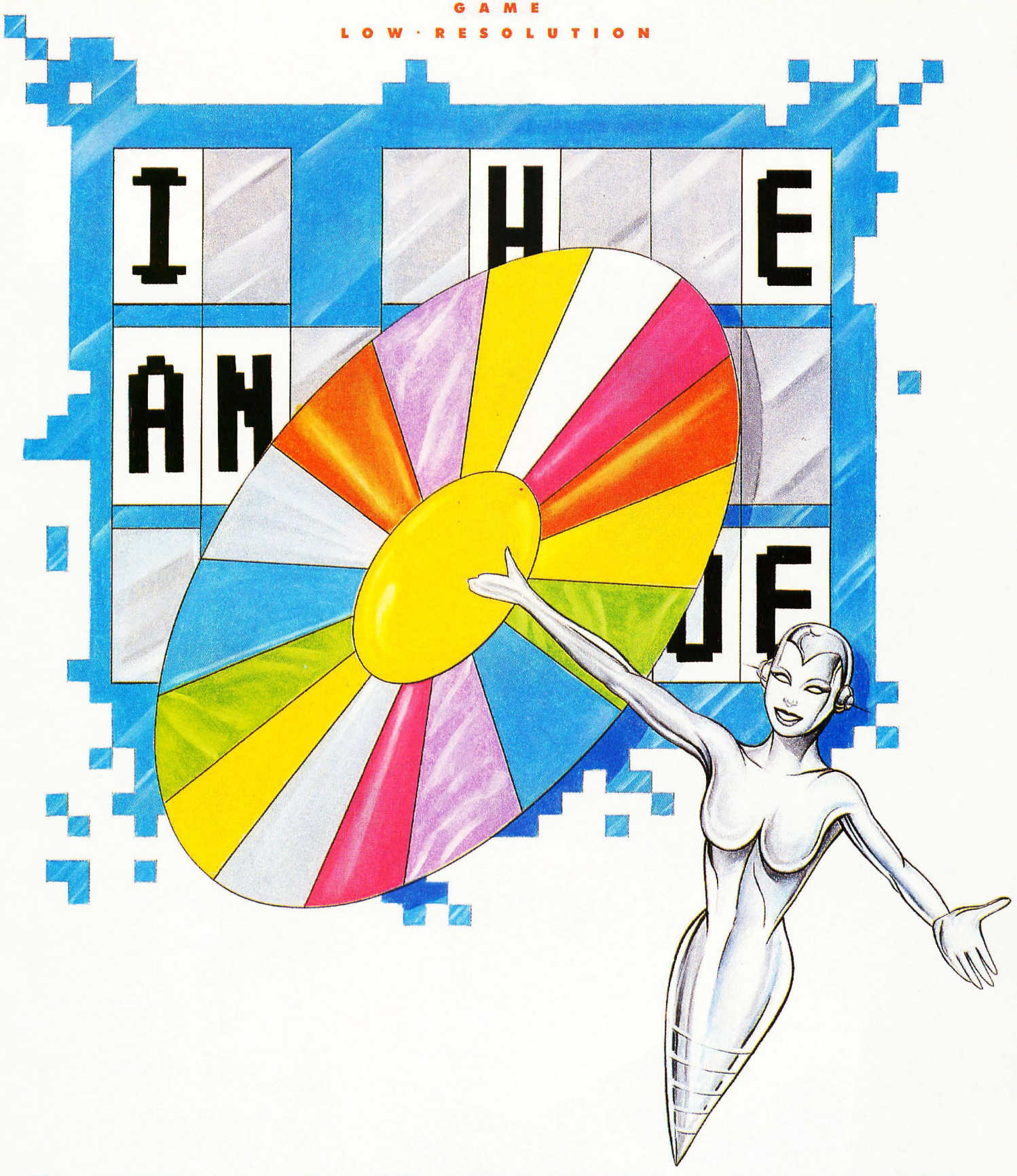
Players continue to choose letters and buy Vowels until someone thinks they have the puzzle solved. When you feel you know, click on Solve Puzzle. This will give a prompt at which you type in what you think the solution is. It must match exactly, or you'll lose your turn. You can pause the game at any time by pressing any key, and resume play in the same manner. You can end the game at any time by pausing the game and pressing the escape key.

Whoever solves the puzzle correctly has their score from that round rolled over into their game total. The losing player receives no money for that round. There are three rounds in a game, and whichever player accumulates the largest dollar amount over the three rounds is the winner.

Have fun playing Mouse of Fortune, but don't expect to see Vanna White. There's only so much an Atari can do! *(to page 68)*

Kirk Stover lives in Minnesota, where he is a systems analyst for an insurance company. He has worked with computers for about six years, yet remains intrigued by them.

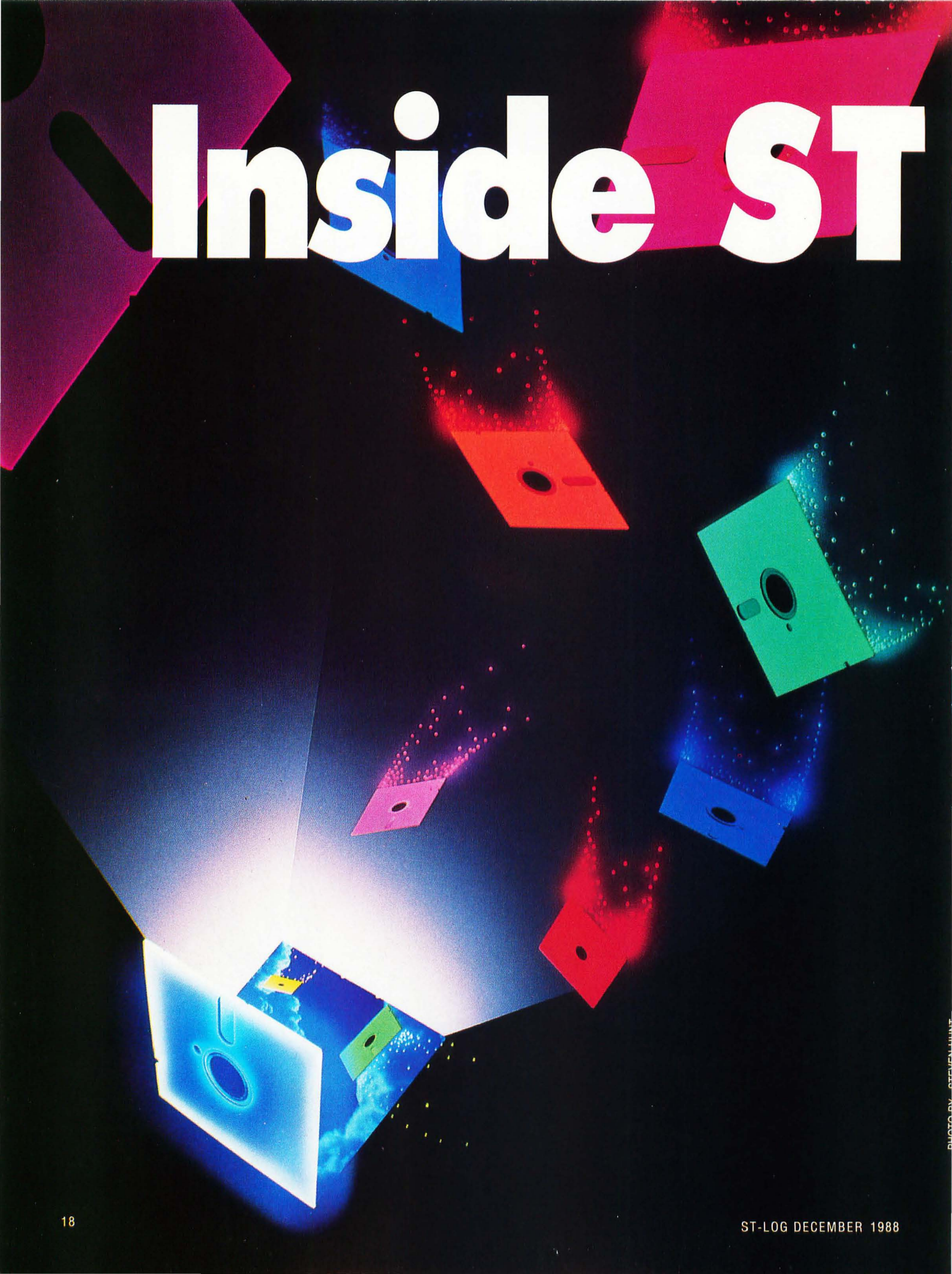
MOUSE



OF FORTUNE

ILLUSTRATION BY BRIAN JON HABERLIN

Inside ST



Xformer II

A TUTORIAL ON WRITING FASTER 68000 PROGRAMS

by Darek Mihocka

About a year ago, ST-Log published the *ST Xformer*, an Atari 800 emulator whose inner workings I described in "Inside the ST Xformer" (ST-Log 17). At that time things seemed pretty well complete. The emulator worked—sort of—and I was satisfied that the assembly-language code making up the 6502 interpreter was well optimized—sort of. I knew of some places here and there where a few clock cycles could have been saved, but the savings wouldn't have amounted to much. The emulator ran at about 20% the speed of a real Atari 800, and it seemed that that was about as fast as it was going to go.

A few alert programmers who read the article and looked through my source code noticed a few more odd places where cycles could have been saved, but nothing that would have significantly sped up the emulator. But by now, most of you reading this know that *ST Xformer II* was released over the past summer, and it runs at an amazing 40% of the speed of an Atari 800—100% faster than the previous emulator.

A faster interpreter

How was it done? What happened between the summer of '87 and the summer of '88 was that I learned a lot more about 68000 machine language than I really ever wanted to. As most programmers know, if you want to optimize a high-level program, you rewrite it in assembler. But if you want to optimize an assembler program, you have to find a better algorithm. And if that doesn't work, cheat! That's what I did with the Xformer II.

I learned quite a few good 68000 tricks, and I'll cover them all in this article. One important fact about the 68000 is that because of its orthogonal design, there are usually many ways of getting it to perform the same task, and sometimes the less obvious choices are faster and better. But first, to refresh everyone's memories about the code we're about to get into, I'll briefly describe how the old emulator functioned. Those of you familiar with writing compilers and p-code interpreters will immediately recognize that my 6502 emulator is nothing more than a p-code interpreter itself. Microprocessors, after all, are just hardware implementations of p-code interpreters.

In the case of the 6502, we have a p-code instruction set of 256 possible instructions, of which about two-thirds are actually valid instructions, with the rest usually treated as NOPs (although some of them also have the undesired effect of putting the 6502 to sleep).

The 6502 has a small set of registers. The A register, or accumulator, is an 8-bit-wide general-purpose register. The X and Y registers are also eight bits wide and are used mostly for indexing purposes. There is an 8-bit status register with the usual flags like N (negative), Z (zero), C (carry), V (overflow) and D (binary coded decimal mode). An 8-bit stack pointer and 16-bit program counter complete the set.

The first Xformer did what most p-code interpreters do: it used a jump table to perform a fetch-and-execute sequence over and over again. Most BASIC interpreters, and even some compiled languages, use this method.

An opcode is fetched by the virtual 6502 program counter, and that opcode is used as an offset into a table of addresses which point to various opcode emulation routines. Some of them do nothing,

but still have to be in the table (like the 100 or so NOPs in the 6502 instruction set).

So of course, the obvious code would look something like this:

```
; d0 - temporary register
;      to store opcode and
;      calculate offset
; a0 - virtual 6502 program
;      counter
; a1 - temporary register
;      used to calculate
;      jump address
;
   clr.w   d0
   move.b  (a0)+,D0
   lsl.w   #2,d0
   move.l  jump_table,a1
   move.l  0(a1,d0.w),a1
   jmp     (a1)
```

If you kept your jump table within 128 bytes of this routine (known as a dispatcher), you could get away with:

```
clr.w   d0
move.b  (a0)+,d0
add.w   d0
add.w   d0
move.l  jump_table(PC,d0.w),a1
jmp     (a1)
```

Note the use of two ADD instructions to replace a shift, which is faster. All together, the first method shown takes $4+8+12+20+20+8=72$ clock cycles. The second method takes $4+8+4+4+20+8=48$ cycles which is 50% faster.

I used a similar dispatcher in Xformer which required 88 clock cycles in total. There was some other overhead, but of those 88 cycles, 48 were required for the fetch-and-jump operation.

I recently made some modifications to the old emulator to allow me to monitor exactly which 6502 opcodes get executed and how many times within a given period of time. The results were very interesting.

The speed of the emulator (with the extra tracing code) was about two million instructions per minute, or about 33,000 per second. This gives an average 6502 instruction time, dispatch to dispatch, of about 240 clock cycles. Take off the 72 cycles required by the extra tracing code, and you get 168 cycles, or 22 microseconds. Compare this to the three or four microseconds required by a real 6502, and you see why the emulator was so slow. The dispatcher was eating up more than half the time, 88 cycles.

This one fact disturbed me for months. After all, the dispatcher should be invisible. It simply guides the flow of the program to the proper opcode emulation routines (the 256 routines that I said do something). It gets executed over 30,000 times per second, over and over again, and just eats up precious time. Obviously, if the dispatcher could be eliminated altogether, the average time per opcode would drop from 168 to 80 cycles, doubling the speed of the emulator. This of course would require compiling the 6502 code ahead of time, and I wasn't quite ready for that.

Now, you readers that didn't immediately think of the jump table idea a few paragraphs ago were probably thinking "You fool, don't look up the address; just calculate it." This was suggested to me by several people, even before Xformer first got published, but early attempts to come up with code that took less than 88 cycles failed.

The basic idea is this: keep the 256 routines spaced apart at regular intervals, say 64 bytes. This would allow most of them to fit within their alloted space, and if not, they could always branch somewhere else. Then, when the dispatcher fetches an opcode, it multiplies it by 64, and jumps that many bytes ahead. The multiply by 64 is really a shift of six bits to the left, and the code could look something like this:

```
clr.w   d0
move.b  (a0)+,d0
lsl.w   #6,d0
jmp     0(PC,d0.w)
```

This takes 48 cycles and is not any faster than the table lookup method when the load pointer and jump code is added.

I was about to give up on this idea when the number 256 was suggested, rather than 64. Multiplies by 256 can be accomplished by shifting 8 bits to the left, which is really a shift of an entire byte. And address register can easily be loaded with a pre-shifted byte by using the little known MOVEP instruction, as the following code shows:

```
movep.w 0(a0),d0
addq.l #1,a0
clr.b   d0
jmp     0(PC,d0.w)
```

But this too takes 48 cycles, the same as the multiply by 64 routine. Gee, wasn't I just saying something about the 68000 being capable of doing the same thing in a number of ways?

Fortunately, the 68000 usually has a faster way. Any assembly-language programmers reading this who haven't already whipped out their M68000 Programmers Reference Guide (Motorola, 1986) may wish to do so, and see if they can find a faster way.

Here it is: the trick is, of course, self-modifying code. I can already hear the screams of "Foul!" ringing out, but the plain fact is, the 68000 is not as advanced as the 80386 or 68020 and couldn't care less if you modify its code at runtime, as long as you're not modifying the very next byte. Besides, if I ever port this to a 68020, I can just use the table lookup instructions anyway! So, by using a self-modifying jump instruction, the code becomes:

```
      move.b (a0)+,label+3
label:
      jmp    $0002(PC)
```

This only requires 32 cycles, which is definitely faster than 48. By using an address register to store the value of *label + 3* so that the value does not have to be fetched every time, the cycles required drops to 24. I mentioned before that I was out of address registers. In this case, I could use the register that I had previously used to point to the jump table. (I didn't use the PC relative method I showed above because the Megamax C inline assembler wouldn't let me. *Boo, hiss.*)

I was eventually able to free up more address registers (I'll get to that later) and eliminated the need to load the pointer that points to the dispatcher. A simple JMP through an address register would do, bringing the total dispatch time to 32 cycles. This is much better than 88 cycles. A quick calculation now showed that the emulator's speed could be improved by a factor of $(88+80)/(32+80)=1.5$, or in other words, it could run at 30% the speed of an Atari 800 instead of 20%, just making this relatively simple coding change.

Just in case anyone is lost, here is how the self-modifying dispatcher works. The two instructions, when assembled, generate six bytes (three words) of code. The first word is the MOVE.B instruction. The second word is the JMP instruction, and the third word is the signed offset for that JMP.

The opcode is fetched and then stored in the fifth byte, which starts off as $00. This, of course, modifies the offset. If the opcode was $12, then the new offset is $1202. If it was a $9B, then the offset is $9B02. These offsets are signed, so the dispatcher itself, at offset $0000, is in the very middle of the code, which has to be a 64K block of memory to allow for offsets from $8002 to $7F02.

FIGURE 1

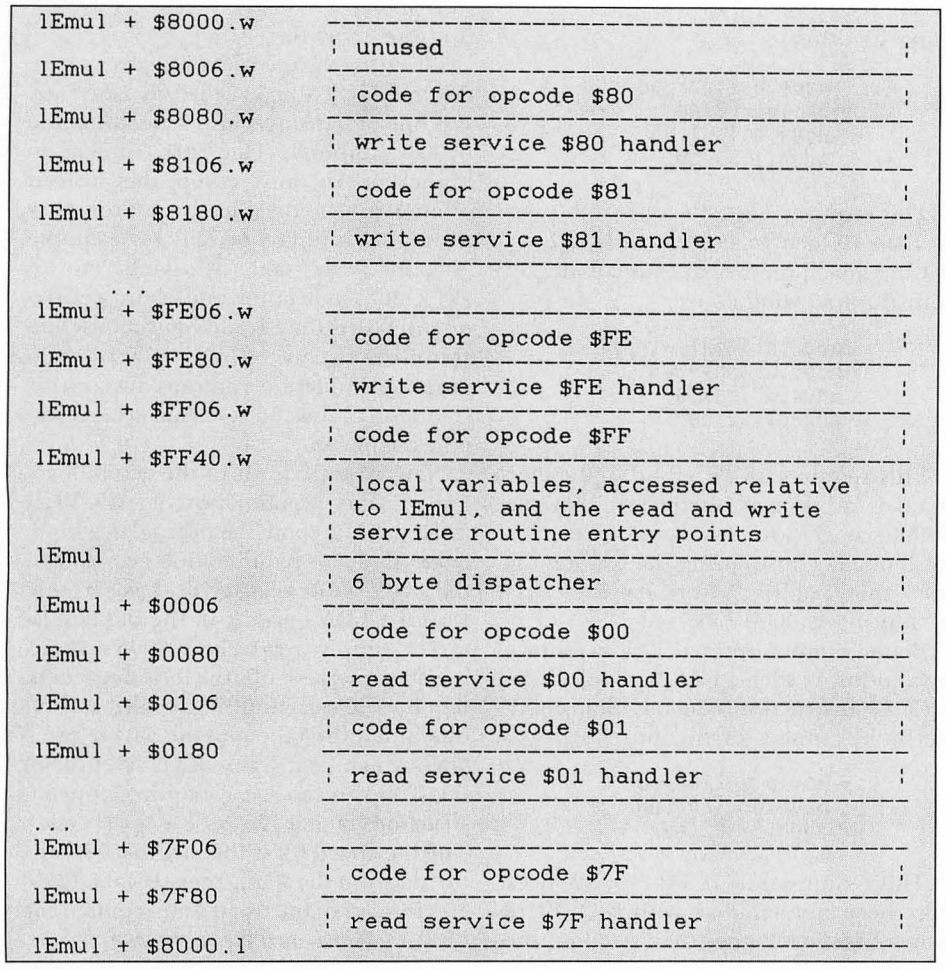


Figure 1 is a diagram of the memory organization of the emulator's code. This memory block can be anywhere in the ST's memory and need not be on a 64K boundary, since everything is relative. The pointer *lEmul* points to the dispatcher, all memory references are relative to this value.

Read and write service handlers are another 256 routines I'll mention soon, but they were able to share memory with the 6502 dispatcher since none of the opcode routines was ever longer than about 80 bytes.

I discovered all this just a few weeks before the ST-Log containing the emulator hit the stands. Too late now, I thought, but it sure will make the next update look better.

Some of you may recall from the first emulator what the read and write service routines were all about. As the 6502 emulator is chugging along, every once in a while it will encounter a 6502 instruction that affects a hardware register or screen memory location. In the first Xformer, I had an extra 64K memory block that mapped each byte of the 6502's address space to a type of memory. $00 indicated it was normal RAM, and values from $01 to $FF were used to indicate various hardware registers. When the emulator came across a memory reference to a location whose type was not $00, it branched to a separate dispatcher which took the type byte ($01 to $FF) and did another jump table dispatch to a handler.

This worked fine to trap all write to screen memory and hardware registers, but reading bytes from screen memory also caused it to jump to this dispatcher. In Xformer II, I fixed this by having two separate 64K arrays: one for memory reads and one for memory writes. The read array contains 62K of zeroes and 2K of non-zero values which are mapped to the 2K of hardware register memory. Thus any reads from the screen memory, BASIC or OS ROMs are treated as normal reads from RAM.

Again, rather than use a jump table, I created two more six-byte dispatchers: one for the read handlers, one for the write handlers. Each one dispatches to a 64K block of memory, with the handlers at 256-byte intervals. To simplify things (and save a lot of memory), the two blocks were overlayed. Read handlers took one 32K chunk, and the write handlers took the other 32K. All this was offset by 128 bytes and overlayed on top of the 64K block of code used by the 6502 emulator. This now completes the diagram.

What made this whole thing tricky was that I was writing this using Megamax C, and then later Laser C. Neither one allows you to write code that spaces routines 256 bytes apart—unless you manually count the bytes yourself. So, using the old jump tables, the emulator constructs the 64K block at runtime. All the opcode handlers and read and write handlers are compiled without any sort of spacing, then at runtime, the routines are copied one by one into this 64K block, at 256-byte intervals. Again, since all the code is relative, there was no need to worry about absolute memory references getting messed up.

Smarter code

Xformer II runs at 40% of the speed of the original 8-bit OS, so some other speed increases must have been found, right? Yes!

The remaining speedups were basically just code optimizations. They required a less significant leap of the imagination than the self-modifying dispatcher but are interesting nevertheless because they once again show that the obvious code is not always the fastest. In fact, as you will see, there are many 68000 instructions that shouldn't even be allowed, because by using them, we all end up writing slower code. A quick look at any part of the ST's ROMs shows this.

One of the most common ones missed by most compiler writers is the method used to clean up the stack after a function call. The 68000 provides a way of doing the clean-up fast with the ADDQ (add quick) instruction, which, in two bytes of code and eight cycles, adds a number from 1 to 8 to an address register. This is fairly fast for most calls, but what happens if there were more than eight bytes on the stack? Flip the page in the Motorola book, and the obvious choice seems to be ADDA (add to address register), which requires an extra two-byte signed offset, and an extra four cycles. What most people miss is that the same thing can be accomplished with no speed loss with the LEA (load effective address) instruction. For example, to increment the stack pointer by 20, do a "lea 20(a7), a7" (eight cycles) instead of a "adda.w 20(a7), a7" (twelve cycles).

Another goodie that most people have caught on to is that to shift a register by one or two bits to the left, it is faster to add it to itself rather than shifting it. For example, use "add.w d0, d0" (four cycles)

instead of "lsl.w #1, d0" (eight cycles). All the flags except overflow are affected the same.

Another common mistake is when clearing registers. To clear a data register, use "moveq #0, d0" (four cycles), instead of "clr.l d0" (eight cycles).

I ran across these mistakes in my own Xformer code, and by making these small changes, the code sped up by about another 10%.

This still leaves a considerable speed increase unaccounted for. A quick look through the Xformer II code would seem to indicate that it had reached its top level of optimization. The self-modifying dispatcher can't be improved upon, and any possible instruction substitutions had been made. Again, I turned to the tracing dispatcher for help.

I set it up to run some program, say Atari BASIC, for about two minutes. Then, two minutes and four million 6502 instructions later, it dumped a list of the number of times each of the 256 opcodes had been executed. Of course, the 100 or so NOPs all had counts of 0.

I found that most programs (like Atari BASIC), the operating system and most binary files only use a set of about 110 to 120 unique instructions out of the 150 or so possible. (The emulator does not handle or look for the "hidden" 6502 instructions). Of these, there is a core of about 30 instructions that are executed almost all the time. The remaining 90 or so are used very infrequently. Topping off the list of most executed instructions were BEQ and BNE, each with about 7% of the total count. Following them were instructions like LDA #, ADC #, LDY #, and STA zero page. As expected, the LDA, LDX, and LDY instructions, with all their addressing modes, were executed about half the time. The remaining half is taken up by store instructions like STA and the remaining branch instructions like JMP, JSR, RTS, BCC, etc.

This now made me focus on the LDA instructions. How could they be sped up? Listing 1 and 2 are examples of the old code that emulated the LDA abs instruction (opcode $AD) and the code that replaced it.

The first optimization deals with how a 16-bit effective address, in low-byte high-byte format, is loaded into a 68000 register. This particular function is required in many places in the emulator and is even required by the ST to read GEMDOS disks, which are in MS-DOS (low-byte hi-byte) format.

The most obvious code would be something like this:

```
move.b 1(a0),d0
lsl.w  #8,d0
move.b (a0),d0
addq.l #2,a0
```

This requires 12 + 24 + 8 + 8 = 52 cycles on the ST. Ouch! In last year's article, I was patting myself on the back for coming up with the following code:

```
move.b 1(a0),-(sp)
move.w (sp)+,d0
move.w (a0),d0
addq.l #2,a0
```

This requires 16 + 8 + 8 + 8 = 40 cycles. It takes advantage of the fact that the 68000's stack pointer always decrements by 2, not 1. Thus putting the byte on the stack then reading a word has the effect of shifting it eight bits.

Better, but not optimal, as I was shown by another reader. I had overlooked the MOVEP instruction, which is almost (but not quite) suited for this purpose:

```
movep.w 1(a0),d0
move.b  (a0),d0
addq.l  #2,a0
```

This requires only 16 + 8 + 8 = 32 cycles. For those not familiar with MOVEP, it works like this: Rather than fetching two consecutive bytes from memory (which must be word aligned), it fetches a byte, skips a byte, and fetches another byte, with no alignment restrictions. It's handy for accessing some of the 8-bit devices mapped on the STs memory, like the video chip, which have registers spaced apart like that. The reason it is useful here is that it loads the high byte of the effective address straight into the high byte of D0.w, thus eliminating any need for shifting.

Going back to the code segment above, we now have the effective 16-bit address of the LDA instruction in the data register labeled *drEA*. In Xformer II, I immediately check that effective address against $C000. If the address is below $C000, it means we are fetching a byte from the 48K of RAM in the Atari 800, or maybe from the BASIC cartridge ROM. In either case, it is safe to do so. Otherwise the status byte is loaded and some service routine is executed.

In the old emulator, I had a single entry point to the service routine, so the flag *isread* was used to tell the service routine whether a read or write was being attempted. Although this is unnecessary in Xformer II which has a separate read and write handler, there is a better way of setting the *isread* flag.

This time the instruction that saves the day is, oddly enough, called ST (set true). It is one of sixteen *set* instructions that the 68000 supports. They are similar to branch instructions, except that instead of branching to the effective address, they simply set it to $00 or $FF. For example "st (a0)" is the same as saying "move.b #$FF, (a0)" but faster and shorter. I frequently used the ST and SF (set false) instructions in the Atari 800 graphics-emulation routines, where it is frequently required to clear or set a certain bit plane.

Now we skip down to the actual loading of the accumulator, a "MOVE.B (arEA), drA" in both the old and new emulator. All that is required now is to update the 6502 status register, and we're done with the LDA opcode. In the old emulator, I simply read the 68000's status register, masked off the bits I needed, namely the zero and negative (Z and N) bits, and moved them to the data register *drST*, which kept track of all seven 6502 flags. As you can see, it required four instructions taking 12 + 8 + 8 + 4 = 32 cycles. The big speed hit is the fact that MOVEs to and from the status register take 12 cycles instead of the usual four required for MOVEs between data registers.

I spent several weeks thinking about this one problem. If some way could be found to eliminate the MOVE from the status register, maybe eight cycles could be saved. Once again, the tracing version of the emulator pointed out the solution.

Since most of the frequent opcodes were of the load variety (LDA, LDX, LDY), only the Z and N flags were getting updated frequently. The other flags, like carry and overflow, were updated after ADC, SBC, and shift instructions. I decided to reserve *drST* only for the N and Z flags, and keep the rest off-chip somewhere in memory where they could be accessed relative to the dispatcher.

This modification allowed me to MOVE the value being loaded directly into *drST*. The branch instructions, like BEQ, BNE, were then updated to simply test *drST* for a zero or non-zero value, rather than testing specific bits. This gave a significant speed increase, bringing the speed of the emulator up to the current 40%.

The actual implementation looked like this: *drST* was divided into three parts: the lower byte, the lower word and the whole register. The topmost bit (bit #31), was used to store the D (decimal) flag of the 6502, since it was rarely set or reset but

checked constantly by the add and subtract instructions. It could be checked with a simple "TST.L drST." Similarly, the N flag was stored in bits 8 through 15, and could be tested for with a "TST.W drST." Finally, the zero flag could be tested with a "TST.B drST."

Each of the TST instructions takes only four cycles, rather than the eight required for a bit test. Fortunately, I did have a spare data register lying around, so I used it to store the carry flag. After some arithmetic operation, it could be set or reset with the SCS instruction, as could the overflow flag.

To see a great example of an instruction that really sped up, Listings 3 and 4 are the old and new code for the ADC# instruction. It assumes the decimal flag is off. Listings 5 and 6 are the old and new code for the ASL A instruction. Note the significant speed increases due to the better dispatcher and the better handling of flags.

A neat trick I use for setting the X (extended carry) flag in the status register without MOVEing to the SR for both the add and shift instructions is to simply add *drC* to itself. Since *drC* is updated with the "SCS drC" instruction, its value is always either $00 (carry flag clear) or $FF (carry flag set). Adding *drC* to itself will either generate $00 with no carry or $FE with a carry. The $FE is thrown out right away and is irrelevant, but it does accomplish the task in four cycles instead of 12.

Other Enhancements

Although the changes to the 6502 interpreter are the most important new features of Xformer II, there are other big improvements. Anyone who has used it will immediately notice the GEM-based interface which replaces the old text interface. Another enhancement is in the way disks are emulated. The old emulator cheated by monitoring the CIO entry point and branching off when disk operations were attempted. It was somewhat compatible at the file level, but did not allow for sector-level I/O and had no support for DOS disks.

Early in the development stages of Xformer II, I decided that a major key to compatibility was to be disk compatible. That would mean that someone could port an 8-bit, 5 1/4-inch disk to the ST (somehow) and run it on the emulator as if it was running on a real 8-bit computer.

Fortunately, the 8-bit disk has a simple format, similar to the format used by the ST itself. There are 40 tracks of 18 sectors, with 128 bytes in each sector. The sectors are numbered 1 through 720 and can be accessed by sending a SIO (serial input/output) command to the disk-drive controller. Because the controller is on the drive and not on the computer, it means that there is no way to bypass the controller (unless you stick in a *Happy* chip, but let's ignore this fact!). Therefore it was very simple to emulate the controller. It has a small number of commands, like format, read sector, write sector, and status. It also returns a very small number of error codes.

What I originally did to emulate the controller was to keep a 90K block of memory which contained the information of the 720 sectors. When a command was sent to the disk drive, the emulator would simply grab 128 bytes in this block and treat it as a sector. Writing a sector meant writing to this block, and reading a sector meant reading from this block.

This has worked so well that this method is still in the current emulator. Of course, the requirement of keeping these 90K blocks of memory (180K for double-density emulation) means that Xformer II can only run on a one-meg Atari ST, as some unhappy 520ST owners found out.

Another improvement is in the way I handle hardware registers. The old emulator handled a few locations, like the sound registers and color registers. This was done by assigning unique status bytes to those registers in the status array. Unfortunately, I left a lot of the registers untouched. Xformer II on the other hand at least does something, anything, when one of the 136 hardware locations (yep, 136, count 'em) in the ANTIC, POKEY, PIA and GTIA chips is accessed. Xformer II also supports multiple occurrences of hardware registers by masking unneeded bits from the effective address.

Player/Missile graphics are now finally supported, to a degree. After unsuccessfully trying to come up with fast pixel-blitting routines to do software sprites (in much the same way that the ST's mouse cursor is done), I decided to cheat and take advantage of bit planes. Since the emulator runs in low resolution, which provides four bit planes, I could get away with using three of the bit planes for the regular graphics display of up to eight colors and then use the fourth bit plane to store the Players and Missiles. This way, when a player moves, its bit image is moved only within the one bit plane, leaving the underlying graphics image intact. These enhancements don't really contribute to the overall speed of the emulator, although they do greatly improve compatibility. The disk emulation and Player/Missile graphics code is written in C, taking advantage of Laser C's ability to use both C and assembly-language source code.

What next?

I'll try not to make any guesses as to what sort of emulators will be around next year. I really put my foot in my mouth last year by saying that the old emulator was about as good as it was going to get. And, unlike last year, I already know that I'll be working on a new version of the emulator which should run at full speed or faster. It will simply compile the 6502 code into 68000 code and eliminate the need for a dispatcher all together. This alone can almost double the speed of the current emulator, but I've got a few other tricks up my sleeve which I'm not going to discuss quite yet.

I know that several other emulators are being written, one a 6809 emulator based on the technology I presented last year, and another 6502 emulator, so things are definitely going to get better yet!

Anyone interested in getting their hands on the source code to Xformer II can find it on DELPHI, Compuserve and GEnie. It's a huge ARC file, and you must have Laser C from Megamax if you want to compile it.

I can be reached by mail at the following addresses.

CompuServe: 73657,2714

DELPHI, GEnie and BIX: DAREKM

I for one am going to apply some of the tricks from the emulator and use them to write some fast replacement routines for TOS, which, as I already mentioned, uses the most non-optimal 68000 code almost everywhere. ■

ST Xformer LISTING 1

ASSEMBLY

```
; LDA abs from Xformer 1.2, simplified for demonstration purposes
; minimum 112 cycles + 88 for dispatcher = 200 cycles
; arPC  - address register which is the virtual 6502 prorgam counter
; drEA  - data register where the effetcive address is calcualted
; arEA  - address register where effetcive address is moved to
; drDATA - data register used for general purpose work
; drST  - data register containing the 6502 status register flags
; drSTAT - pointer to 64K status byte array
; drA - virtual 6502 accumulator
; isread - global flag used by emul_serv
;
opAD:
   move.b 1(arPC),-(SP) ; push high byte on stack
   move.w (SP)+,drEA    ; read as a word
   move.b (arPC),drEA   ; fetch low byte of effective address
   addq.w #2,arPC       ; increment virtual program counter
   move.b 0(drSTAT,drEA.w),IR  ; get status byte
   bpl.s  doLDA         ; if positive, not a read service
   addq.w #1,isread(A4) ; else set read flag
   bsr    emul_serv     ; and go service
;
;at this point the service routine executed or the status byte was 0
;
doLDA:
   move.l drEA,arEA     ; move effective address to addr reg
   move.b (arEA),drA    ; load accumulator
   move.w SR,drDATA     ; get 68000 status register
   and.b  #(BITN|BITZ),drDATA ; isolate N and Z bits
   and.b  #~(BITN|BITZ),drST ; clear old N and Z bits
   or.b  drDATA,drST    ; update 6502 status register
   DISPATCH             ; go to dispatcher
```

ST Xformer LISTING 2

ASSEMBLY

```
; LDA abs from Xformer 2.1, simplified for demonstration purposes
; minimum 72 cycles + 32 for dispatcher = 104 cycles
; same naming convention as above, plus
; arRSTAT - address register pointing to 64K read status array
; arEMUL - address register pointing to dispatcher
;
opAD:
    movep.w 1(arPC),drEA    ; fetch high byte of effective addr
    move.b (arPC),drEA      ; fetch low byte
    addq.w #2,arPC          ; increment virtual program counter
    move.l drEA,arEA        ; move effective address to addr reg
    cmpi.w #0xC000,drEA     ; is EA less than $C000
    bcs.s  .1               ; yep, it can't be a hardware reg
    move.b 0(arRSTAT,drEA),drDATA ; otherwise fetch status byte
    beq.s  .1               ; if zero, probably ROM, so skip
    jsr   lReadDisp(arEMUL) ; else go service it
.1:
    move.b (arEA),drA       ; load accumulator
    move.b drA,drST         ; set Z flag
    ext.w  drST             ; set N flag
    DISPATCH                ; go to disptacher
```

ST Xformer LISTING 3

ASSEMBLY

```
; ADC # from Xformer 1.2, simplified for demonstration purposes.
; 64 cycles + 88 for dispatcher = 152 cycles.
; arPC - address register which is the virtual 6502  program counter
; drDATA - data register used for general purpose work
; drST - data register containing the 6502 status register flags
; drA - virtual 6502 accumulator
;
op69:
   move.b (arPC)+,drDATA   ; fetch number
   ori.b  #BITZ,drST       ; set Z flag (required by 68000)
   move.b drST,CCR         ; move C and Z flags to status register
   addx.b drDATA,drA       ; perform the add
   move.w SR,drDATA        ; get new status register
   and.b  #(BITN|BITZ|BITV|BITX),drDATA  ; mask
   and.b  #~(BITN|BITZ|BITV|BITX),drST  ; mask
   or.b  drDATA,drST     ;update
   DISPATCH
```

ST Xormer LISTING 4

ASSEMBLY

```
; ADC # from Xformer 2.1, simplified for demonstration purposes
; 48 clock cycles + 32 for dispatcher = 80 cycles
; same naming convention as above
;
op69:
   move.b (arPC)+,drDATA    ; fetch number
   add.b  drC,drC           ; set X bit in status register
   addx.b drDATA,drA        ; perform the add
   scs    drC               ; update C flag
   svs    bV(arEMUL)        ; update V flag
   move.b drA,drST          ; update Z
   ext.w  drST              ; update N
```

ST Xformer LISTING 5

ASSEMBLY

```
; ASL A from Xformer 1.2, simplified for demonstration purposes
; 40 cycles + 88 = 128 cycles
;
op0A:
   asl.b  #1,drA                        ; shift accumulator left
   move.w SR,drDATA                     ; get new status register
   and.b  #(BITN|BITZ|BITX),drDATA      ; mask
   and.b  #~(BITN|BITZ|BITX),drST       ; mask
   or.b   drDATA,drST                   ; update N Z C flags
```

ST Xformer LISTING 6

ASSEMBLY

```
; ASL A from Xformer 2.1, simplified for demonstration purposes
; 20 cycles + 32 = 52 cycles
;
op0A:
   add.b  drA,drA       ; perform the shift
   scs    drC           ; update C
   move.b drA,drST      ; update Z
   ext.w  drST          ; update N
```

IAN'S QUEST

by Ian Chadwick

Hooray! I recently got my upgrade of Empire (version 2.05) from Interstel, and I'm quite happy about it. Aside from several new commands and some improvements in the way the old commands worked, my favorite game is now significantly faster. Also, the graphics have been enhanced a teensy bit, improving the aesthetic appeal of the game. All in all, it makes Empire just that much better as an already terrific game. I have a whole panoply of ideas for them (helicopters, airborne troops, missiles, etc.), but for now I like it just as is.

On the other hand, the changes are documented in a barely readable file on the new disk, incongruously called "README.DOC," significant only in the excruciatingly bad grammar and numerous spelling errors that crop up in it like crabgrass on my lawn. And that's not to mention the inexcusable use of the passive voice throughout. The author of this miniature horror, Mark Baldwin, has crafted a superb game, but he can't write English worth snail trails.

I know, I know, no one expects programmers to be able to write in English, but surely they have the technical skills to use a spelling checker.

There's no reason a single written page should go out of a software publisher without being checked for spelling and grammar. When I get one of these little models of illiteracy, I always feel like I'm a Cro Magnon in a world of Neanderthals.

Come on, guys, this is supposed to be a *professional* business, right? Is it too much to ask for you to assemble a document without at least spelling errors?

Of course, Interstel isn't the only company guilty of mangling the English language.

Take a look at Avalon Hill's new release, *Spitfire*: "as does the up and down direction keys." The correct form of the verb, as you might guess, is "do." (You *did* guess that, didn't you?) Also, they repeatedly talk about the "verticle speed indicator."

Michtron's GFA BASIC 3.0 manual is chock full of little errata, a myriad of annoyances, including "compare to" (compare with is correct), "variable MENU(2) to MENU(15)" (variables; it's plural), and "the Atari ST has two interface ports to connect a mouse and a joystick" (to which to connect . . .). Its author is guilty of overwriting to the extreme: "this function centers the object on the coordinates that are specified." What's wrong with "the specified coordinates"? "This command waits for a keypress on the key board": why keypress (one word) and key board (two words)? And where does the author think a key will be pressed? On the disk drive?

Sometimes the errata are more the result of sloppy editing than any inability to write the Queen's English.

Omnitrend's *Paladin* was rewritten from the Breach manual and as such, several oversights happened. For example: "carry a very weak sword". Weak sword? Probably they meant a "light sword". What about "fired a sword"? "Swung a sword" is more likely.

Not that these problems usually affect the programs, but they do seriously upset any modestly literate reader. How many times have you read about output being "printed on the printer" or "displayed on the screen"? Where else will something be printed or shown? Or the phrase "data is"—data is plural and the correct phrase is "data are". Or "your" instead of "you're"?

Then again, I've seen more than a few errors crop up in menu bars, dialog boxes, help files and the like. Atari's release of *Microsoft Write* had oodles of grammatical glitches in the program messages. I think the blame for this one lies with Atari, since Microsoft, not unlike Pontius Pilate, washed its hands of the program a while back.

The ST world isn't the only group at fault, by the way. I recently got Borland's *Sprint* word processor for my AT, and it came with a README file with eight 8.5 by 11-inch pages of corrections and clarifications to their manuals! And the English is only marginally better. However, ST publishers seem particularly prone to scale the heights of mediocrity in manuals.

In one sense, the inadequacy of documentation is good: it keeps the computer book publishers in business. However, a weak manual reflects more on the inadequacies of the publisher than on the ability of a writer to overcome them.

I've been writing and editing computer manuals for five years now and I've seen a lot of inconsistencies, passive voice, grammatical atrocities, spelling errors and just plain bad writing. Every time I come across errors in a professional manual, I shudder.

There are basic rules that should be followed by every software publisher before printing:

1) Don't let the programmers write the manual.

2) Use a spelling checker before printing.

3) Have the manual(s) read by qualified third parties and by potential users. Documentation should be as thoroughly beta-tested as the program, both to the quality of the language and the correctness of the instructions and descriptions.

4) Use professional writers and editors to write the documentation—at least as equally well qualified as your programmers.

Pretty basic stuff. You'd be surprised how many companies ignore these rules. I've been working with ISD's *DynaCAD*—a superb program—but the manual is, simply put, inadequate. Why? Because the programmer wrote it (and he's a nice guy too, but as a writer, he's out of his depth).

Really, all of this is inexcusable. Several hundred books on the art and craft of writing English are available at any good bookstore or library. Personally, I feel any technical writer who can't quote chapter and verse from the Chicago Manual of Style should be shot. Can't quote Strunk and White? Another one to the wall.

"How many times have you had your hair turn grey trying to uncover the meaning of an obscure error message the writers neglected to document, but it happens every time you try to save a file?"

There are some good books available about writing documentation, including *Hardwords/Softwords* from Ashton-Tate and *The Computer Documentation Kit* from Reston. Apple publishes a decent style guide for Mac developers, but we pariahs have a hard time getting hold of it (besides, it's good but not *great*, and I have arguments with several parts of it).

The point is that there's no need for bad documentation. Of course, improved English won't help the terminally obscure manuals. How many times have you had your hair turn grey trying to uncover the meaning of an obscure error message the writers neglected to document, but it happens every time you try to save a file?

When I was at Batteries Included (BI), I wrote a basic style manual for writers, which never got published for obvious reasons. (Has anyone seen any BI products since the takeover?) Not that I'm an authority (I am, however, a technical writer by profession), but it's certainly something this industry needs. Perhaps it could be a place to begin an enlightened dialogue. I've considered redoing it and making it available (perhaps as a four- to six-part series of magazine articles). Any interest? Write to ST-Log and let them know. I'd be happy to do it.

Although I'm no longer writing the GFA BASIC column, I still get Michtron's product releases, so I should keep you up to date with their efforts. For those of you who haven't kept up with the state of affairs as far as GFA BASIC is concerned, Michtron has not kept still. And before I go any further, let me make one of those sweeping, unequivocal statements that always generate nasty mail: GFA BASIC is, as far as I'm concerned, the best programming language available for the ST, superior to C or Pascal. And it's certainly better supported by Michtron than any other company supports any of their languages, as far as I've seen.

Let's look at what they've been up to since GFA was first released. Unless noted, everything below is from Michtron.

With each release, GFA BASIC has improved, and by all means, that is true of the latest version. The language constantly improves and commands have been added, particularly in 3.0. Also, the manuals have improved considerably with each revision.

Training Reboot Camp is a beginner's guide with a silly title. It's a good introduc-

tion for the newcomer to BASIC, though rather weak in the advanced features.

The *Programmer's Reference Guide, Volume 1* is a moderate to advanced level guide, containing lots of useful information, but the major failing is the lack of short examples. I simply won't type in 20 to 30 pages of code to try and learn a simple technique to include in my own programs. About one half of the book rehashes the program manuals, with only moderately more programming information than they provide. The rest contains very good information for technical data, especially in the GEM, AES and VDI areas.

GFA Vector, a 3D-graphics program for GFA BASIC inclusion, is slow, but if 3D graphics are your interest, this package is pretty good. If you don't speak German, you won't understand the remarks in the code. Some of the examples don't work and there's no way to tell why because of the language barrier.

The *GFA BASIC Reference Card* is an accordion-type card with all the commands, minimally documented. It's good if you know the language and need a reminder.

Abacus has released the *GFA BASIC Quick Program Reference*, which is a modest substitute for the program manuals, with some information not contained in them and other material left out.

The *GFA BASIC Book* is an intermediate programming tutorial and very good for learning the language. A significant improvement over the original manuals, it's my favorite of the lot.

GFA Companion is a resource construction set with a horrendously restrictive requirement not to publish any code produced by it, except if compiled. This makes Companion of limited use to anyone. There are several public-domain dialog- box creation kits and example programs that will allow you to do about 80% of what this program accomplishes, without the nonsense about compiling code.

Okay, I said I got Paladin from Omnitrend, so I'd better say something about it. You liked *Breach*? You'll like Paladin. They're almost identical programs. Instead of grenades, you get orbs of fire; instead of missiles, you get crossbows—you get the idea. The differences between Breach and Paladin are mostly in the terminology. They are almost identical in function.

Yeah, they've thrown in a handful of magic spells (six spells altogether), but it's nowhere near the same sophistication as *Dungeon Master*.

Breach and Paladin also share the same serious programming flaw: only orthogonal movement is allowed, not diagonal. For the uninitiated, this means that if it costs five movement points to enter a square, the player has to move up one and across one, rather than move one diagonally, a total of ten movement points. However, the math of diagonal movement dictates that it should cost only seven points (rounded down), leaving three for other purposes. Obviously, Omnitrend has a terminal problem dealing with square roots. I can recommend a few basic books in math and algebra.

Paladin is a Conan-style magical adventure. The approach is tactical infantry style: lots of sword play, missiles and exploding weapons. The emphasis is on combat rather than on magic. The magic is restricted to things that suit combat and exploration. Forget trying to call up elementals, transforming your opponent into a duck, changing lead into gold. Paladin works on the "zap-em" level of magic.

Not that this is bad; it's merely limiting. It suits the game, and the scenarios are designed around objective-oriented situations, all involving a lot of hacking and slashing. It's not High Fantasy, but Paladin, like its predecessor Breach, is enjoyable and challenging despite the flaws. [A full review appears elsewhere in this issue—ed.] ▪

Ian Chadwick is a technical writer and editor based in Toronto, where he lives with his wife, Susan, and their numerous pets, including Kepler, a dog who is laboriously being taught to play chess. Ian is also trying to get published a 120,000-word fantasy novel he just wrote.

Attention Programmers!

ST-LOG Magazine is interested in programs, articles, and software review submissions dealing with the Atari home computers. If you feel that you can write as well as you can program, then submit those articles and reviews that have been floating around in your head, awaiting publication. This is your opportunity to share your knowledge with the growing famiily of Atari computer owners.

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

Send your programs and articles to:
ST-LOG Magazine
P.O. Box 1413-M.O.
Manchester, CT 06040-1413

ST—DATE PLANNER
LISTING 1
GFA BASIC

```
' ST Date Planner by David Plotkin
' for ST LOG
Dim Day$(7),Month$(12),Strip$(50),Lastday(12),Appt$(31,21),Astatus(31)
Gosub Initialize
Rez=Xbios(4)
' Rez=1 for med res (640x200),2 for hi res (640x400)
If Rez=0 Then
  A$="ST Date Planner must be|run in Med. or Hi Rez"
  Alert 1,A$,1," OK ",Dum
  End
Endif
If Rez=1 Then
  Nme$="SCR.PI2"
Else
  Nme$="SCR.PI3"
Endif
Path$="A:\*.*"
Nm$=""
Dxpos=160
Dwidth=340
If Rez=1 Then
  Dypos=50
  Dheight=100
Else
  Dypos=100
  Dheight=200
Endif
Attr=Windtab+2
Xpos=Windtab+4
Ypos=Windtab+6
Width=Windtab+8
Height=Windtab+10
Attr2=Windtab+14
Xpos2=Windtab+16
Ypos2=Windtab+18
Width2=Windtab+20
Height2=Windtab+22
Dpoke Attr,&H1
Dpoke Xpos,0
Dpoke Ypos,19+10*(Rez=1)
Dpoke Width,639
Dpoke Height,399+19*(Rez=2)+200*(Rez=1)+10*(Rez=1)
Dpoke Attr2,&H1
Dpoke Xpos2,Dxpos
Dpoke Ypos2,Dypos
Dpoke Width2,Dwidth
Dpoke Height2,Dheight
Gosub Menusetup
Menu Strip$()
On Menu  Gosub Respondmenu
On Menu Button 1,1,1 Gosub Menubutton
Openw 1
Gosub Clearappt
Gosub Drwmonth
Aftererror:
Do
  On Menu
Loop
Procedure Initialize
  For Index%=1 To 7
    Read Day$(Index%)
  Next Index%
  For Index%=1 To 12
    Read Month$(Index%)
  Next Index%
  For Index%=1 To 12
    Read Lastday(Index%)
  Next Index%
  Dayname$="  Sunday     Monday    Tuesday    Wednesday   Thursday    Friday    Saturday"
  Bar$="l          l          l          l          l          l          l          l"
  Lyne$="+----------+----------+----------+----------+----------+----------+----------+"
  Daytop$="Sun        Mon        Tues       Wed        Thurs      Fri        Sat"
  ' Bon$ is code to turn bold print on, Boff$ turns it off
```

```
    Data January,February,March,April,May,June,July,August,September
    Data October,November,December
    Data 31,28,31,30,31,30,31,31,30,31,30,31
    Holddate$=Date$
    Currmonth=Val(Left$(Holddate$,2))
    Curryear=Val(Right$(Holddate$,4))
  Return
  Procedure Menusetup
    For I%=0 To 50
    Bon$=Chr$(27)+"G"
    Boff$=Chr$(27)+"H"
    Data Sunday,Monday,Tuesday,Wednesday,Thursday,Friday,Saturday
      Read Strip$(I%)
      Exit If Strip$(I%)="***"
    Next I%
    Strip$(I%)=""
    Strip$(I%+1)=""
    Data Desk," Date Planner "
    Data --------------------
    Data 1,2,3,4,5,6,""
    Data  File , Save File , Save Screen , Load File , Quit ,""
    Data  Print , Print File , Print Screen ,""
    Data  Selection ," Select Date "
    Data ***
  Return
  Procedure Drwmonth
    ' Calculate and draw selected month on the screen
    St=False
    For Lpp=1 To 21
      If Appt$(Chosenbox,Lpp)<>"_________________________" Then
        St=True
      Endif
    Next Lpp
    If St=True Then
      Astatus(Chosenbox)=1
    Else
      Astatus(Chosenbox)=0
    Endif
    Deffill 1,1,0
    Curtitle$=" "+Month$(Currmonth)+" "+Str$(Curryear)
    Titlew 1,Curtitle$
    Clearw 1
    Print At(6,1);Daytop$
    Xstart=12
    For Lp=1 To 8
      Line Xstart,16+8*(Rez=1),Xstart,399+200*(Rez=1)
      Xstart=Xstart+88
    Next Lp
    Ystart=16+8*(Rez=1)
    For Lp=1 To 6
      Line 0,Ystart,639,Ystart
      Ystart=Ystart+57+29*(Rez=1)
    Next Lp
    M=Currmonth
    Y=Curryear
    If M<3 Then
      M=M+12
      Y=Y-1
    Endif
    N=2*M+Int(0.6*(M+1))+Y+Int(Y/4)-Int(Y/100)+Int(Y/400)+3
    N=Int((N/7-Int(N/7))*7+0.05)
    Q=N
    If M>12 Then
      M=M-12
      Y=Y+1
    Endif
    Dy=Lastday(M)
    If M=2 Then
      If (Y/4)=Int(Y/4) And (Y/100<>Int(Y/100)) Or (Y/400=Int(Y/400)) Then
        Dy=Dy+1
      Endif
    Endif
    If Q=0 Then
      Q=7
    Endif
    ' Q tracks days of week from 1 (Sunday) to 7 (Saturday)
    X=((Q-1)*11)+4
```

```
    Yy=3
    D=1
    Do
      Print At(X,Yy);D
      If Astatus(D)<>0 Then
        Print At(X+3,Yy);"*";
      Endif
      Exit If (D=Dy)
      D=D+1
      X=X+11
      If X>76 Then
        X=4
        Yy=Yy+3.5
      Endif
    Loop
  Return
  Procedure Respondmenu
    ' Respond to menu selection
    Hold$=Strip$(Menu(0))
    If Hold$=" Select Date " Then
    ' my version of the dialog box
    Titlew 2,"Select Date"
    Openw 2
    Clearw 2
    Tempyear=Curryear
    Tempmonth=Currmonth
    Print At(11,1);"Type in new year"
    Print At(7,2);"Press [Return] when through"
    Print At(15,3);" ";
    Input Yr$
    If Yr$<>"" Then
      Curryear=Val(Yr$)
    Endif
    Clearw 2
    Print At(1,2);"  Jan    Feb    Mar    Apr    May    Jun  "
    Print At(1,4);"  Jul    Aug    Sep    Oct    Nov    Dec  "
    For Lp=1 To 5
      Line Lp*56,0,Lp*56,(16+8*(Rez=1))*5
    Next Lp
    Yye=(16+8*(Rez=1))*2.5
    Line 0,Yye,Dwidth,Yye
    Line 0,Yye*2,Dwidth,Yye*2
    Alldone=False
    Do
      ' wait for mouse button
      Repeat
      Until Mousek=1
      Xc=Int(Mousex/56)
      Yc=Int(Mousey/Yye)
      If Xc>=0 And Xc<=5 And Yc>=0 And Yc<=1 Then
        Graphmode 3
        Pbox Xc*56,Yc*Yye,Xc*56+56,Yc*Yye+Yye
        Currmonth=(Xc+1)+Yc*6
        Alldone=True
        Graphmode 1
      Endif
      Exit If Alldone=True
    Loop
    Closew 2
    If (Curryear<>Tempyear Or Currmonth<>Tempmonth) Then
      Gosub Clearappt
    Endif
    Gosub Drwmonth
  Endif
  If Hold$=" Save File " Then
    On Error Gosub Errorroutine
    Nm$=Month$(Currmonth)+Str$(Curryear)
    Fileselect Path$,Nm$,Nm2$
    If Nm2$<>"" Then
      Defmouse 2
      Open "o",#1,Nm2$
      Print #1,Currmonth
      Print #1,Curryear
      For Lp1=1 To 31
        For Lp2=1 To 21
          Bput #1,Varptr(Appt$(Lp1,Lp2)),Len(Appt$(Lp1,Lp2))
```

```
      Next Lp2
    Next Lp1
    Close #1
    Gosub Gtnewpath(Nm2$)
    Defmouse 0
  Endif
  On Error
Endif
If Hold$=" Load File " Then
  On Error Gosub Errorroutine
  Fileselect Path$,Nm$,Nm2$
  If Nm2$<>"" Then
    Defmouse 2
    Open "i",#1,Nm2$
    Input #1,Currmonth
    Input #1,Curryear
    For Lp1=1 To 31
      For Lp2=1 To 21
        Bget #1,Varptr(Appt$(Lp1,Lp2)),Len(Appt$(Lp1,Lp2))
      Next Lp2
    Next Lp1
    Close #1
    Gosub Gtnewpath(Nm2$)
    Defmouse 0
    Gosub Checkstatus
    Gosub Drwmonth
  Endif
  On Error
Endif
If Hold$=" Date Planner " Then
  ' Credit for me!
  A$="ST Date Planner|by David Plotkin|Written in GFA Basic|May 1,1987"
  Alert 1,A$,1," OK ",Dum
Endif
If Hold$=" Quit " Then
  A$="You want to quit|this great program?|HAVE YOU SAVED YOUR DATA?"
  Alert 2,A$,1,"Yes|No",Dum
  If Dum=1 Then
    End
  Endif
Endif
If Hold$=" Save Screen " Then
  ' Save screen in DEGAS format
  On Error Gosub Errorroutine
  Fileselect Path$,Nme$,Nme2$
  If Nme2$<>"" Then
    Hidem
    Open "o",#1,Nme2$
    Degas$=String$(34," ")
    Mid$(Degas$,1,1)=Chr$(0)
    Mid$(Degas$,2,1)=Chr$(Rez)
    I%=3
    For Index%=&HFF8240 To &HFF8240+32 Step 2
      Dumn=(Dpeek(Index%) And &HFFF)
      Dumhi=(Dumn And &HFF00/256)
      Dumlo=(Dumn And &HFF)
      Mid$(Degas$,I%,1)=Chr$(Dumhi)
      Mid$(Degas$,I%+1,1)=Chr$(Dumlo)
      I%=I%+2
    Next Index%
    Bput #1,Varptr(Degas$),34
    Bput #1,Xbios(3),32000
    Close #1
    Showm
  Endif
  On Error
Endif
If Hold$=" Print File " Then
  ' Print out all days schedules for current month
  Alert 3,"Align Paper|Turn on Printer",1,"OK",Dum
  Defmouse 2
  For Lp=1 To Lastday(Currmonth)
    Daynum=((Lp-1)+Q) Mod 7
    If Daynum=0 Then
      Daynum=7
    Endif
    T$=Day$(Daynum)+" "+Month$(Currmonth)+" "+Str$(Lp)
```

(to page 38)

ST USER

by Arthur Leyenberger

I was just going though the past issues of ST-Log. In reading the previous ST User columns I noticed something peculiar. There is an excellent product that I use each and every time I turn on my ST, but I have failed to mention it even once. It is *DeskCart!* from QMI.

DeskCart! is a desktop accessory cartridge for the ST that simply plugs into the cartridge port on the computer. The cartridge contains a battery-backed-up clock and 14 desktop accessories. What is remarkable is that DeskCart! only takes up 84K of computer RAM (Random Access Memory). The battery-operated clock/calendar sets the system date and time of the machine and loads automatically.

The 14 accessories are all contained on a 64K ROM (Read Only Memory) in the cartridge and are contained under one accessory called "DeskCart!" Since the maximum number of allowable accessories on the ST is only six, QMI cleverly used only one desk accessory entry so you could use other accessories as well.

The 14 accessories include: a calendar with appointment scheduler and alarms; multiple notebooks with full-screen editing; a full-featured calculator with either mouse or keypad entry; a card filer with sorting, searching and printing functions; a typewriter function for quick labels and envelopes; an address book with searching and auto-dialing; a VT52 terminal emulator; keyboard macros for use with just about any program; a fast and resizeable RAMdisk driver; disk utilities including format, copy, delete and rename; a print spooler; a more complete control panel with load and save options; a screen dump to any graphics printer and a system memory test.

DeskCart! is always plugged into my ST. If nothing else, I use it for the battery-operated clock. But there are also a couple of the accessories that I use a lot. The calculator is one of them. It is a multi-function scientific and programmer's calculator, meaning that it has exponential notation and scientific operators, as well as hexadecimal calculation and arithmetic logic functions. It contains 100 storage memories that are kept for as long as the current desktop session. Further, when you save the control panel, the storage memories are also saved.

I like the convenience of being able to call up the calculator regardless of what I am doing at the time. If you are a programmer, you'll like the hexadecimal functions. It can do the following operations: 2's complement, left shift, right shift, arithmetic AND, OR and XOR (exclusive OR).

The notebook is another accessory that gets heavy use. Notebook files are standard ASCII files and can be up to 12 pages long. When displayed on the screen, the vertical slide bar includes the page number which is a useful touch. When you click on the notebook accessory, a window opens up with five icons across the bottom. Load and Save allow you to store and retrieve any notebook file from the DeskCart! File Selector.

The File Selector is another nice touch from programmers John DeMar, Robert Dolan and Richard Gortatowsky. Across the bottom of the File Selector window is a row of drive letters. Clicking on the drive letter is a preferred alternative to using the GEM File Selector close window box to move up through directories in order to access another drive. Also, the DeskCart! File Selector allows you to sort the listing in the window by either name, date or size.

The Print icon lets you print one or all of the pages from the currently displayed notebook file. The Find icon lets you move rapidly to the top or bottom of the document and the Erase icon wipes out the contents of the current window.

Disk utilities is another useful accessory. Being able to copy, delete and rename files from within another GEM application program is very handy. Also, I can format a disk with the DeskCart! utilities and display the status of any disk drive.

The screen dump is the other utility that gets frequent use. Although it is somewhat similar to the built-in GEM screen dump, this one allows you to print either vertically or horizontally. Also, you can load printer drivers from the screen dump window.

DeskCart! lists for $100 (Quantum Microsystems, Inc., P.O. Box 179, Liverpool, NY 13088, 315-451-7747) and has been around for about two years. Although there are various individual utility programs available either commercially or via freeware that handle some of the functions of DeskCart!, I know of no other product with as many features as DeskCart! I can't imagine anyone using an ST that couldn't use even just some of the features of this program. I consider it the single best add-on product for the Atari ST computer. It would certainly make a nice holiday gift.

More Favorite Hardware

There are two other products that are always attached to my ST system. Both of them come from Practical Solutions, and they are the *Monitor Master* and *Mouse Master* switch boxes. Monitor Master permits me to connect both a monochrome and a RGB monitor to the ST at once. By pressing a button on the front of the box, I can select between color or monochrome displays without having to hassle with disconnecting and re-connecting monitor cables. This doesn't relieve you from having to reboot your ST when you switch, but is nonetheless a time-saver. Monitor Master also provides a composite video signal (only on STs with an RF modulator) and an audio signal.

The composite output is especially useful for the ST's low-resolution mode. In fact, if you only have a monochrome monitor, you could get away with a TV for playing games on the ST. Monitor Master's video output is also useful to feed to a VCR. Just remember: the ST medium-resolution mode really requires an RGB monitor for clarity. Unfortunately, there is no way of getting around it.

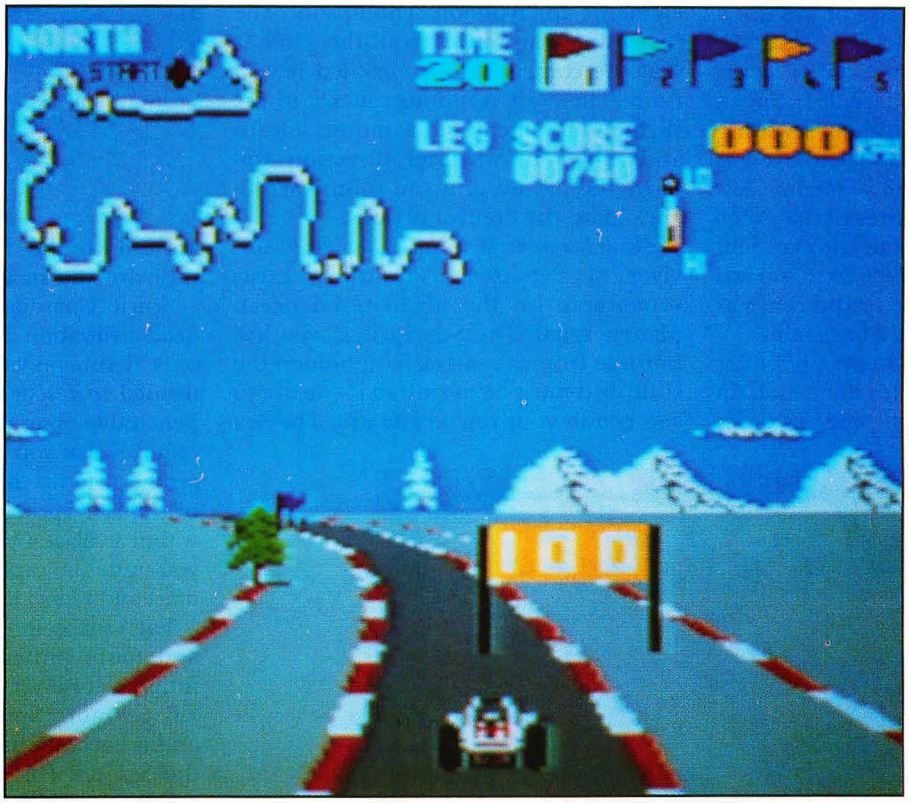
The other Practical Solutions product I use always is the Mouse Master. Mouse Master plugs into the two joystick ports of the ST and lets you connect a mouse and a joystick to Port 0 at the same time. A switch on the top of the box switches between mouse and joystick. There is also a Port 1 (joystick) connection. If you have a 1040ST, with the mouse/joystick ports underneath the computer, you *need* Mouse Master.

Monitor Master sells for $55, and Mouse Master sells for $40 (Practical Solutions, Inc., 1930 E. Grant Rd., Tuscon, AZ 85719, 602-884-9612). The good people at Practical Solutions tell me there will soon be (by the time you read this) a *Drive Master* that will allow you to switch between two disk drives—say an Atari ST drive and a 360K PC drive.

After Hours

Recently a couple of new games appeared on my doorstep. Naturally, in the interest of journalistic research, I immediately ran down to the "computer center" (it used to be a normal basement before the invasion of the machines) to begin some serious, scientific investigations on this software. Of course, I had to postpone the chores I had planned for the day. Hey, this is science.

One of the new games is *Speed Buggy* by Data East. When I saw it at last summer's Consumer Electronics Show, I really wasn't that impressed, but I decided to check it out anyway. Whoa, this is one neat game.



SPEED BUGGY • by Data East

"Speed Buggy by Data East, is a driving game that offers five race courses for you to master."

Speed Buggy is a driving game that offers five race courses for you to master. Four of them are racing circuits (more or less), and the fifth one is an off-road course. You compete against the clock and are disqualified from the race if you do not complete each leg within the allotted time.

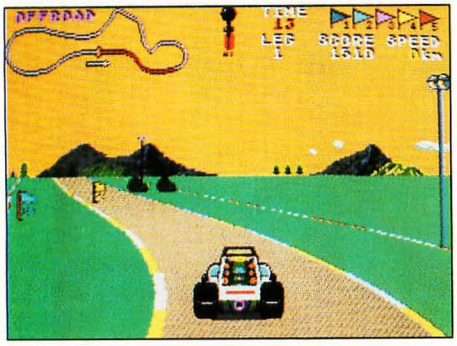
You must race across terrain littered with boulders, trees, brick walls, fences and a whole lot more. In addition, the track continues through narrow valleys, across even narrower bridges, through dark tunnels and past road construction (on your race track!). If you hit fences, rocks, walls or trees during the course, it will flip your car over (a nicely done animation) and slow you down. However, you can jump obstacles by hitting the logs that lay across the track which will send your buggy soaring through the air. You can also tilt the buggy on two wheels by driving over tree stumps or small rocks. This technique is useful for squeezing through narrow gaps.

Along the course are gates, each of which is worth from 100 to 500 points if you can pass through them. Passing through a special "time gate" adds an additional two seconds to your next leg. There are also differently colored flags, each worth from 10 to 30 points. If you pick up the flags in the specified order, bonus points are awarded.

I have long been a sucker for racing games, going back eight years to my early 2600 VCS days, so the genre and I are well acquainted.

Speed Buggy is a nicely implemented racing game for the ST. There is plenty of challenge and the graphics are well-done. At first, I was oversteering and had a difficult time controlling the car, but one particularly nice feature of the game is the self-centering characteristic of the

buggy. A light touch is all that is needed on the steering, especially at higher speeds.



I highly recommend Speed Buggy regardless of whether you are or are not a hard-core racing-game fan. For about $30 (Data East USA, Inc., 470 Needles Drive, San Jose, CA 95112, 408-286-7074), Speed Buggy is worth the investment in fun.

The other new game that arrived is *Typhoon Thompson in Search for the Sea Child* by Dan Gorlin. You may recall the excellent game that Dan wrote for the 8-bit Atari computer called *Choplifter*. It too was from Broderbund Software and ranks as one of the all-time best 8-bit games.

One of the things that made Choplifter so successful was its detailed animation. In that game you flew a helicopter in order to rescue hostages behind enemy lines. As you landed the chopper, the people would run toward it waving their arms and screaming. Once you returned the hostages to the friendly U.S. position, they filed out, stopping to wave to you in thanks.

When Choplifter first came out, the Iran hostage affair was still fresh in our minds. I don't know if that influenced my perception of the game, but I do know that the detailing of the characters was excellent. It was easy to identify with those little people on the screen. Another thing that made Choplifter so good was the control action. The helicopter was easy to fly. Indeed, it felt *natural* to fly.

Dan Gorlin has brought both of those elements to Typhoon Thompson. The first thing you notice about the game is that the control of your Jet-Sled is very fluid. The craft responds quickly to the slightest movements of the mouse. It leans and moves in whatever direction you move the mouse, and you soon learn to anticipate the control actions.

About the game: It seems that a child survivor of a passenger transport exists somewhere on the unchartered ocean planet Aguar. The Sea Sprites, who live in these tropical waters, have hidden the child and must be outwitted if you are to succeed in your rescue attempt. The Sea Sprites use different kinds of Flyers [sic] as their defense. They range from the benign Bumper which just pushes your sled around to the fatal Zapper who can vaporize you in a flash.

"A high score roster is provided so you can compete with yourself or your friends".

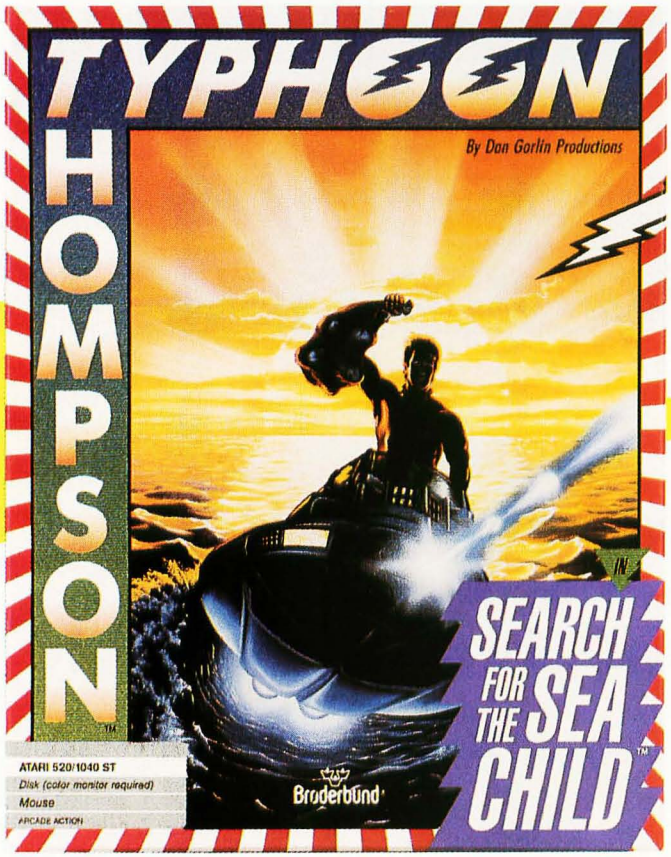
TYPHOON THOMPSON
by Dan Gorlin

Spirit Guardians from Aguar's long-dead civilization are there to help you but only if you provide them with the tools needed to rescue the baby from the impenetrable observatory dome. Each round requires that you retrieve one of the four artifacts. To do that you must capture all of the Sea Sprites from all seven islands within a village. Once that is accomplished, you trade the captured Sprites for the artifact and return it to the Spirit Guardians.

Each succeeding mission presents new and more difficult challenges. Greater numbers of Flyers attack from the island village, more Sprites must be captured and it becomes more difficult to capture the Sprites. Fortunately, the Guardians will arm you with additional, more powerful weapons in order to capture the elusive Sea Sprites.

Your final score is based upon the number of tasks you complete, the number of islands won, the number of Sprites captured, and the amount of time it took to complete the tasks. A high score roster is provided so you can compete with yourself or your friends. With the challenging game play, 3-D graphics, quick animation and overall quality of Typhoon Thompson, there will be plenty of competition. This game is really addicting. Nice job, Broderbund!

Typhoon Thompson is priced at $34.95 (Broderbund Software, Inc., 17 Paul Drive, San Rafael, CA 94903-2101, 800-527-6263). ■



STPlus•STPlus•STPlus•STPlus

P.O. 1197, Berkeley, Ca. 94701 • add 3% credit card for hardware

Front line NEWS: GENLOCK for the ST, $400, preorder.

Spectre 128 (run Mac SE programs) $179 • PCDitto, DOS, & Drive $399

BUSINESS

DBMan 4.0	175.00
Datamanager	39.95
Superbase	104.95
Trimbase	69.95
Phasar 3.0	63.95
Zoomracks 2	84.95
Base 2	42.95
The Informer	69.95
Wordperfect	189.95
1st Word Plus	69.95
Word Up!	64.95
Best Accounting	279.95
Equal Plus	139.95
Inventory Mgr.	69.95
Rolobase Plus	63.95
Logistix Spread	104.95
Microlawyer	49.95
Payroll Master	69.95
Construction EST.	35.00
Microsoft Write	94.95
Datatrieve	35.00
STOneWrite	48.95
VIP GEM	104.95
DacEasy Payroll	48.00
DacEasy Acctg	52.00
WordWriter ST	56.00
SwiftCalc	39.95
EZ Calc by Royal	48.95
Analyze Spread	25.95
Final Word	99.95
PublishingPartner	140.00
T-works Publisher	89.95
EZData Base	48.95
Chart Pak	35.00
Compute Roots	27.95
Thunder NEW!	28.95
Habawriter 2	48.95
Text Pro	35.00
Becker Text	62.95
Expert Opinion AI	59.95
Time Link	35.00
Partner ST	48.95
Labelmaster Elite	35.00
ST Accounts	149.00
The Juggler 2.0	35.00
Max Pack	35.00
Stuff	27.95
Flash 1.5	21.00
Omni Res	27.95
Turbo ST(~blitter)	35.00
Signum *technical word processor*	249.95
SBT DLedger	195.00
SBT DPayables	195.00
SBT DInvoices	195.00
SBT DMenu	49.95
Neo Desk	27.95
Sales Pro	69.95
Mail Manager	39.95
Mighty Mail	35.00
First Word 1.6	14.00

GRAFHICS

Degas Elite	41.95
CAD 3D 2.0	63.95
Cyber Paint	49.00
Quantum 4096	27.95
Adv Art Studio	26.00
Spectrum 512	49.00
EzDraw&Superch	104.95
Canon Scanner	1040.0
GFA Artist 1000cl	55.95
Drafix 1	139.95
General Symbols	105.00
Elec, or Arch, Sym	105.00
Athena 2	69.95
Circuit Maker	55.95

CLIPART

Warriors(720)	14.95
Outdoor(720)	14.95
Buildings(720)	14.95
Victorian(720)	14.95
Etchings(720)	14.95
People (720k)	14.95
Politics(720k)	14.95
Religion(720)	14.95
Holidays(720)	14.95
Ad Art (720k)	14.95
Vehicles(720)	14.95
Boats (720k)	14.95
Planes (720k	14.95
All (9.2 meg)	99.95

GAMES

Gunship	35.00
Shadowgate	35.00
Uninvited	35.00
Mouse Quest	14.00
Slaygon	27.95
Barbarian	27.95
Obliterator	27.95
Gauntlet	35.00
Dark Castle	27.95
F-15 Strike Eagle	27.95
Star Trek-Rebel U	27.95
Questron II	35.00
Lock-On	27.95
Carrier Command	32.50

MUSIC

Passport	280.00
Master Tracks	104.95
MasterTracks Jr.	69.95
Midisoft Studio	
Hybrid Arts	
Smpte Track	499.95
Sync Track	299.95
EZ Track Plus	48.95
Midiscore	call
EZ Score Plus	104.95
DX-Android	139.95
CZ-Android	69.95
Gen-Patch	104.95
Dr. T's	
KCSequencer	199.95
KCS 1,6 w/PVG	289.95
MIDI rec studio	56.00
Copyist level 1	75.95
Copyist level 2	185.95
Copy3-Postscript	299.95

Remember: Every one thousandth purchaser gets a hundred dollars credit, and the ten thousandth purchaser will get a 10 meg Supra floppy. RULES: *Have your customer number or credit card ready. Purchase as often as you like. We're your computer supermarket and we're ready to fulfill your dreams but we're not an information service. We ship right away and its your part to know what you want. Don't be cheap, at these prices you can splurge!*

SPECIAL-SPECIAL-SPECIAL

MICROSOFT WRITE FOR ONLY $50

while supplies last.

HARDWARE

10 Meg Supra flppy	875.00
20 Meg w/clock	599.95
30 Meg Supra	749.95
60 meg Supra	1249.95
33 Meg Tulin	699.95
51 Meg Tulin	839.95
80 Meg BMS RRL	1249.95
20 Meg SH205	639.95
250 Meg	3250.0
10 Meg Floppy	849.95
AST PS LASER	3350.00
Canon Scanner	1040.00
IMG Scanner	90.00
ComputereyesMon	120.00
Supra 2400 modem	159.95
Atari SX212 modem	79.95

PROGRAMMING

GFA Basic 3.0	56.00
GFA Book	35.00
GFA Compiler	56.00
Mark Williams "C"	125.00
Laser "C"	159.95
Cambridge Lisp	139.95
RAID	27.95
Fast Editor	35.00
Alice Pascal	69.95
OSS Pascal	59.95
Fortran 77 GEM	139.95
BCPL	104.95
Modula 2 dev. kit	104.95
Assempro	48.95
Fast Basic	56.95
True Basic	69.95

EDUCATIONAL

Arakis Series	14.00
Unicorn Series	27-35
True Basic Stuff	69.95

Autumn 520 STFM Special

$599 Mono

$799 Color

Your Art Scanned $10 per page 75 to 300 dpi

GAMES

Tanglewood	27.95
Test Drive	35.00
Chessmastr2000	32.95
Starglider II *NEW*	35.00
Hunt for Red Oct	35.00
TyphoonThompson	27.95
Aliants	24.95
Alien Fire	35.00
Santa Paravia	21.00
Lurking Horror	27.95
Star Fleet 1	39.95
Empire	39.95
Liesure Suit Larry	27.95
Gridiron	35.00
Dungeon Master	27.95
Flight Simulator	35.00
Trailblazer	27.95
····SPECIALS····	
Jewel of Darknss	19.95
Silicon Dreams	19.95
Cardiac Arrest	48.95

The Best European Games

I Ball (neat,fast)	19.95
RanaRama(d&d)	29.95
Warlock'sQuest	29.95
The Flintstones	29.95
Trivial Pursuit	29.95
The Enforcer	19.95
Seconds Out	29.95
Scruples(board)	29.95
Livingstone	29.95
Battle Ships	29.95
Outrun(fast cars)	29.95
Crazy Cars	29.95
Tetris (from USSR)	29.95
Screaming Wings	29.95
Spitfire	29.95
Blue War	29.95
Star Quake	29.95
Enduro Racer	29.95
BMX Simulator	29.95
Arkanoids	29.95
Better Dead n Alien	29.95

Are you a gamer? How about joining our game of the month club? $10 gets you the hottest new title at an extra 5% off & you can return it for 75% credit. Plus you'll be eligible each purchase to win as #1000 or #10,000. Call us and be first to play the new ones. Start now with Typhoon Thomson for $25.95.

ONE NEW NUMBER **800-759-1110** Prices subject to change without notice.

We ship ANYWHERE! $4.00 min S&H. No 1040's or Megas mail order. Hand delivery only, List plus $100.

CIRCLE #117 ON READER SERVICE CARD.

(from page 33)

```
      Gosub Outputprint(Lp)
    Next Lp
    Defmouse 0
  Endif
  If Hold$=" Print Screen " Then
    ' Print out current month calendar
    Defmouse 2
    Lprint Bon$;
    Lprint "                                    ";Month$(Currmonth)+" "+Str$(Curryear)
    Lprint Boff$
    Lprint Dayname$
    Lp=0
    For Lp2=1 To 6  !each week
      Lprint Lyne$
      For Lp3=1 To 4
        Lprint Bar$
      Next Lp3
      For Lp3=1 To 7
        Lp=Lp+1
        Lprint "l ";
        C=Lp-(Q-1)
        If C>=1 And C<=Lastday(M) Then
          Lprint C;
          If C<10 Then
            Lprint "        ";
          Else
            Lprint "       ";
          Endif
        Else
          Lprint "         ";
        Endif
      Next Lp3
      Lprint "l"
      For Lp3=1 To 4
        Lprint Bar$
      Next Lp3
    Next Lp2
      Lprint Lyne$
      Lprint
      Defmouse 0
    Endif
    Menu Off
  Return
  Procedure Errorroutine
    A$="Error accessing|the disk"
    Alert 1,A$,1," OK ",Dum
    Menu Off
    Gosub Drwmonth
    Resume Aftererror
  Return
  Procedure Menubutton
    ' Respond to mouse button when calendar is on the screen
    Xmouse=Menu(10)
    Ymouse=Menu(11)
    Xstart=Int((Xmouse-12)/88)*88+12
    Xend=Xstart+88
    If Rez=1 Then
      Ystart=Int((Ymouse-28)/28)*28+8
      Yend=Ystart+28
    Else
      Ystart=Int((Ymouse-51)/57)*57+16
      Yend=Ystart+57
    Endif
    Xchosen=(Xstart-12)/88
    ' Xchosen counts the days from the beginning of the week (0-6)
    ' Ychosen counts the weeks from the beginning of the month (0-6)
    If Rez=1 Then
      Ychosen=(Ystart-8)/28
    Else
      Ychosen=(Ystart-16)/57
    Endif
    Chosenbox=((Xchosen+1)-(Q-1))+(Ychosen*7)
    If Chosenbox>=1 And Chosenbox<=Lastday(M) And Xmouse>11 And Xmouse<628 Then
      Graphmode 3
      Pbox Xstart,Ystart,Xend,Yend
      Graphmode 1
      Daynum=((Chosenbox-1)+Q) Mod 7
```

```
    If Daynum=0 Then
      Daynum=7
    Endif
    ' Daynum is the number of the weekday (1-7)
    T$=Day$(Daynum)+" "+Month$(Currmonth)+" "+Str$(Chosenbox)
    Titlew 2,T$
    Dpoke Ypos2,19+10*(Rez=1)
    Dpoke Height2,399+210*(Rez=1)
    Openw 2
    Clearw 2
    Tme=7
    For Lp=1 To 21
      Print At(1-(Tme<10),Lp);Int(Tme);
      If Even(Lp)=-1 Then
        Print At(3,Lp);":30"
      Else
        Print At(3,Lp);":00";
      Endif
      Print At(6,Lp);Appt$(Chosenbox,Lp);
      Tme=Tme+0.5
      If Tme=13 Then
        Tme=1
      Endif
    Next Lp
    Color 1
    Box 250,90+45*(Rez=1),320,150+75*(Rez=1)
    Print At(36,8);"OK";
    Box 250,190+95*(Rez=1),320,250+125*(Rez=1)
    Print At(34,14);"Print";
    Box 250,290+145*(Rez=1),320,350+175*(Rez=1)
    Print At(34,20);"Clear";
    Psy=1
    Psx=1
    Gosub Prntit !put in initial cursor
    On Menu Key Gosub Respondkey
    On Menu Button 1,1,1 Gosub Respondbutton
    On Menu  Gosub Dummy
    Alldone=False
    Do
      On Menu
      Exit If Alldone=True
    Loop
    Closew 2
    Dpoke Ypos2,Dypos
    Dpoke Height2,Dheight
    Gosub Drwmonth
    On Menu Key Gosub Dummy
    On Menu Button 1,1,1 Gosub Menubutton
    On Menu  Gosub Respondmenu
  Endif
Return
Procedure Respondkey
  Key=Menu(14) And &HFF
  Lkey=Menu(14)
  If Key=27 Then !Escape
    Alldone=True
  Endif
  If (Key>=32 And Key<123) Then
    Mid$(Appt$(Chosenbox,Psy),Psx,1)=Chr$(Key)
    Print At(Psx+5,Psy);Chr$(Key);
    Psx=Psx+1
    If Psx>25 Then
      Psx=25
    Endif
    Gosub Prntit
  Endif
  If Key=13 Then !Return
    Gosub Oldprint
    Psx=1
    Psy=Psy+1
    If Psy>21 Then
      Psy=21
    Endif
    Gosub Prntit
  Endif
  If Key=8 Then  !Backspace
    Gosub Oldprint
```

(to page 46)

The reviews are in . . .

"'A Best Buy' I'm impressed"
David H. Ahl, Atari Explorer, Nov-Dec 1987

"If you've got an Atari, you probably need this program."
Jerry Pournell, Byte Magazine, October 1987

"pc-ditto is a winner."
Charlie Young, ST World, July 1987

"This is the product we have been looking for."
Donna Wesolowski, ST Informer, August 1987

"This truly incredible software emulator really works."
Mike Gibbons, Current Notes, September 1987

NOW! RUN THESE IBM PROGRAMS ON YOUR ATARI ST.

Lotus 1-2-3	Flight Simulator	Framework	Symphony
Enable	Ability	DESQview	Q&A
Sidekick	Superkey	Norton Utilities	dBase II, III, III+
Crosstalk IV	Carbon Copy	Chart-Master	Print Shop
EasyCAD	DAC Easy Accounting	BPI Accounting	Turbo Pascal
GW Basic	Managing Your Money	Silvia Porter's	pfs:Professional File

And Hundreds More!

---

pc-ditto is a software-only utility which taps the power of our Atari St to imitate an IBM PC XT. No extra hardware is required (an optional 5.25-inch drive may be required for 5.25-inch disks). All your IBM disks will work "out-of-the-box".

pc-ditto features include:

- All ST models supported (520, 1040, & Mega)
- up to 703K usable memory (1040 & Mega)
- not copy-protected -- installable on hard disk
- imitates IBM monochrome and IBM color graphics adapters
- access to hard disk, if hard disk used
- optionally boots DOS from hard disk
- parallel and serial ports fully supported
- supports 3.5-inch 720K format and 360K single-sided formats
- supports optional 5.25-inch 40-track drives

System requirements:

- IBM PC-DOS or Compaq MS-DOS version 3.2 or above recommended
- optional 5.25-inch drive is required to use 5.25-inch disks
- 3.5-inch 720K DOS disks require a double-sided drive (Atari SF314 or equivalent)

See pc-ditto today at an Atari dealer near you, or write for free information!

pc-ditto
by
Avant-Garde Systems
381 Pablo Point Drive
Jacksonville, FL 32225
(904) 221-2904

$89.95

Avant-Garde Systems, 381 Pablo Point Dr.
Jacksonville, Florida 32225 (904) 221-2904
Yes! Please send information on pc-ditto.
Name_____
Address_____
City_____ State_____ Zip_____

CIRCLE #129 ON READER SERVICE CARD.

STEP 1

OF MICE AND MEGABYTES

PART II · MICE FOLLIES

by Maurice Molyneaux

For those who missed last issue (shame on you!), I discussed how, in late 1986, I obtained copies of what was eventually to become Epyx's *Art & Film Director*. At the time, these Hungarian-designed programs were to have been published in the U.S. by Broderbund. With the software, I had produced a series of animations for a Star Trek game proposal, which were put together on videotape. A copy of the video was taken to Broderbund (by Stephen Friedman, American representative of the Hungarian company Novotrade Software) in order to show them what kinds of things could be done with the software.

I received a call from Steve a few weeks after delivering the tape. He said, "Broderbund wants to know how much you would charge to make a video for *them* to use to promote Art and Film Director."

Naughty But Mice

Surprise! It was the last thing I'd expected to come out of that Star Trek demo! A challenge to be sure. They made no specifics; I had to submit my own concept. Fortunately, I got an idea almost right away. Months earlier I had been toying around with the idea of producing a computer-generated cartoon that would feature a rather smart-alecky mouse, and which would demonstrate various forms of animation (cel, stop motion, claymation) and poke fun at them (to show that Saturday-morning cartoon characters are flat and lifeless the mouse would slap a Smurf, causing it to spin, and we would see that indeed it does have the depth of cardboard!).

I adapted this concept. The video would compare Art and Film Director to other paint and animation programs, and, specifically, poke fun at Saturday-morning or Japanese cartoons. The mouse character would be pitted against the obligatory narrator and the programs themselves. In other words, whenever a feature was shown, it would be demonstrated by using it on the mouse.

The written outline I submitted was accepted, and I was asked to submit a storyboard. I had begun storyboarding (sometimes brainstorming with Vince Reynolds) parallel with writing the proposal, but it still took a long time to get the whole thing hammered out. I was determined to make a "cartoon," not just a series of unrelated animated vignettes, so the structure and pacing of the whole thing, the way gags built on gags, etc., was important. The following, from the original written proposal, makes clear what my intentions were:

". . .The thrust . . . is such that the viewer waits anxiously to see just what can happen [to the poor character] next, and the improbability of the situations keeps it funny. . .By making the demo tape something of a film, complete with a continuing character, people will more likely stop and watch, to see the whole demo and how it ultimately ends, than if we just threw a bunch of pictures on the screen."

At this point I decided to create a cartoon human character. I doodled a short, chubby, bald fellow with a huge round nose. The reasons for creating him were twofold. First, the abrasive personality planned for the mouse was inappropriate for a few scenes I had in mind. Secondly, I was worried that I might not be able to do a good job creating the mouse. If the mouse proved impractical, this character would take his place.

A Star is Bored

The completed preliminary storyboard consisted of 112 "thumbnail" sketches. These were assembled, four to a page, with text describing the action, narration, etc. The basic gist of the video was this:

The human character was to be involved in three opening sequences: a "presented by" gag and two scenes demonstrating the limitations of traditional art forms. Next, we would poke fun at "crude" paint programs, then show off a flood of Art Director features. A bunch of graphic bits would then crash together to form the mouse, and Art Director tools would be demonstrated on him. Next, the mouse would be knocked around by the Film Director interface, then would run a movie projector and show some miscellaneous animations. Finally, the projector bulb would burn out, and the narrator, deciding there was nothing else to show, would erase the mouse animation. The mouse would then push the GEM Desktop screen off the monitor, jump out of the screen and throw the Art & Film Director disk at the "camera," cracking its lens.

Whew!

I estimated the demo would run six to seven minutes. Broderbund wanted ten, but I thought it would be too much work to try for that much time.

The storyboards were approved, but changes were requested. The only serious blow was that I had to remove the sequence making fun of Japanese cartoons, because Broderbund was afraid that viewers passing by at a show would see only that part and think *that* scene was indicative of the programs' capabilities! (The storyboard called for choppy movement,

obnoxious voices, and, of course, out-of synch dialog—just like a real Japanese cartoon.)

By this time the video had been christened "Notions in Motion." It was the beginning of April, 1987. Broderbund wanted the tape for the June CES.

War and Pieces

One of the first things done was the design of the human character—actually making shape tables. I had to figure out a way to make a truly flexible character within the limits of the hardware and software.

(A quick technical discussion: Film Director creates animation using two basic building blocks: polygons and patterns. A polygon is a closed, unfilled, multi-sided outline defined with Film Director's own tools. Patterns are rectangular graphic blocks [with the background color transparent to other colors] "clipped" from pattern pages [full-screen, low-resolution pictures], which are usually imported from Art Director. These elements can be combined into more complex elements, such as Groups, Actors and Stages, which are then combined to make the frames of the animation.)

The idea of drawing a separate image for each pose was quickly discarded. Sixteen screens of pattern data could be loaded into the then-new versions of Film Director, but that wasn't enough for the complex movements the storyboards called for, especially with a character who was to be almost a half-screen tall.

The demo animation of a little boy that came with Film Director was interesting. The boy's shape table took only a single screen and contained all of his parts (hands, feet, etc.), each drawn in numerous positions. Great, except that in order to get such a flexible character in such a small space, the animator(s) had left out a few small details—specifically, limbs. The boy had no arms, legs or neck! He consisted of only a head, body, two hands and two feet. Simple to animate, easy to create shape tables for, yes. What I was after, no. I wanted to make a traditional cartoon character, not a bunch of disjointed parts.

I dug into my animation references and found an article about an obscure animated film called "Twice Upon a Time." That film used what is called the "Lumage" process, which refers to backlit animation using cutout parts. They had libraries of arms, legs and so on that they would put together on an animation board. I had seen the film and was not really impressed by the animation. Yet I was struck by the resemblance between the technique used in Film Director's boy demo and the Lumage process.

It was clear that the best way to design the characters was in pieces. However, it would be necessary to create shape tables of great complexity.

It was clear that the best way to design the characters was in pieces. However, it would be necessary to create shape tables of great complexity. I shuddered at the idea of having to draw scores of heads in every conceivable position. I needed to find a better way—and I did. The character would be composed of pieces, yes, but smaller pieces than the boy had. Rather than drawing complete heads, the head would itself be composed of separate elements, which I could then alter individually. Eyes and other facial parts were created independent of the actual head object, and thus I could have a character twitch an eyebrow a pixel or shift an eye or perform any number of small movements, *without* having to have complete parts for each and every pose.

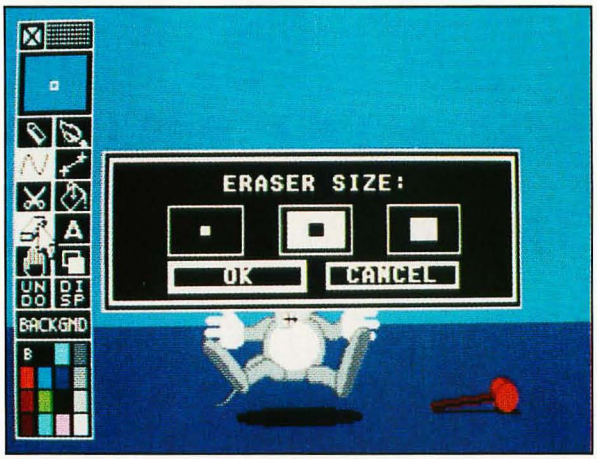
A price was paid for this flexibility. In a given frame of animation, the human character ended up consisting of twice as many parts as the boy in the original Film Director demo, and his shape tables, when properly organized, filled two and a half screens.

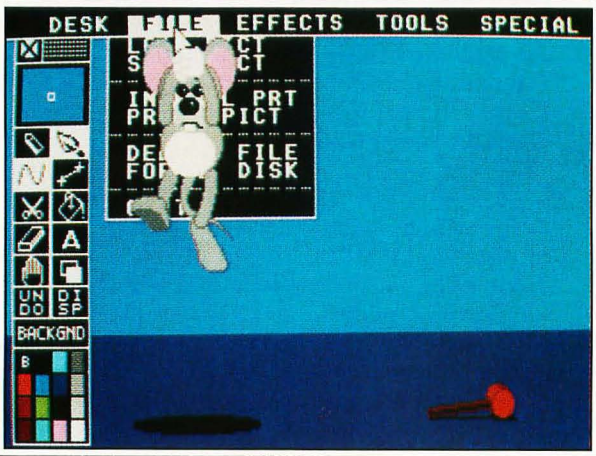
Vince labeled the disk containing these shape tables "tiny little man." *Seeing this,* I dubbed the character "Tiny Mann."

Corny Concerto

The first shot to be animated was called "The Sculptor," wherein Tiny played artist, chiseling on a statue and causing it to crumble. To show how on a computer you can undo your last action, the statue reassembled itself. Of course, it's not possible to reassemble such traditional art forms—so *smash!* the statue collapsed again.

The storyboards called for a statue of "The Thinker." I got the bright idea to swipe the picture of Michelangelo's "Moses" from the Art Director title screen. Not only did it eliminate the need to draw a statue from scratch, but it tied in with the product much better. The picture was resized, recolored, and a base added to the statue. Copies were made of chunks of it to produce the rubble required. A hammer, chisel and Tiny were all that was needed to begin. I sat down and animated, trying to give Tiny the right kind of walk.





Problems were encountered when the statue shattered. I was animating 40 chunks of rubble, plus 12 pieces of Tiny; thus, things slowed to a crawl during playback. A little quick thinking solved this problem, and I had my first completed animation for the video.

I played it back, and, despite my knowledge of each and every frame, it surprised me. It was funny! It was funny in a way I hadn't expected it to be. Somehow, the plight of this little character got through to me.

I next tackled one of the more difficult sequences. It called for Tiny to play maestro and conduct color changes on a painting in time with the opening part of Bach's Tocatta and Fugue in D Minor (the music to be dubbed onto the final videotape).

Redesigning Tiny was no problem. Change his clothes to black, make his shirt a shirt-and-jacket combo, make his hands into white gloves (a five-second job using Art Director's X-Color option), add some tux tails, make a white "Leopold Stokowski" wig, and voila!

I had planned to time key points in the music, break these down into a chart based on frames per second and animate according to that. Afterwards, I would adjust frames to compensate for uneven playback speed on the computer. (It varies because the more screen updating that has to be done, the slower the machine is.) With a planned deadline looming before me, I skipped the timing chart and winged it. Picture a man endlessly replaying the opening strains of Bach's Tocatta and Fugue in D Minor, adding, deleting, changing frames, all to get the animation and music in synch. *Five nights* of this I went through! I thought I'd never want to hear that bloody piece of music again!

Complicating matters, I couldn't just take Tiny's hands from point A to point B and stop. Real animation is the art of movement, and frozen poses are a major no-no! To keep Tiny from freezing up, I had to make his hands quiver as he held them aloft. I also had to make sure to vary the movements slightly so that the quivering wouldn't fall into a pattern and look mechanical.

The fun part of this otherwise painful exercise was animating all the secondary action: his tux tails flapping, his posture, body twist, etc. Tiny "anticipates" thrusting his hand up by first pulling it back. At one point, he swings his arm so forcefully that one foot comes off the floor!

Plane Daffy

The final animation needed of Tiny was to be the opening shot. In it, one letter of the software company's name (originally the "o" in Broderbund, later the "Y" in EPYX) has fallen out of the logo, and Tiny goes to ridiculous extremes to put it in place. He was to throw it, catapult it with a seesaw contraption, and finally blast it in place using a cannon.

As props were drawn, I began to feel that the seesaw gag wasn't going to work. Tiny would have to leave the screen, push the thing in, set the letter on it, and then move around and jump on it. Then he'd have to get rid of it when it didn't work! That would take far too much time.

Megabit's first appearance is in a scene where a bunch of graphic bits fly together and, *bang!*, form a mouse.

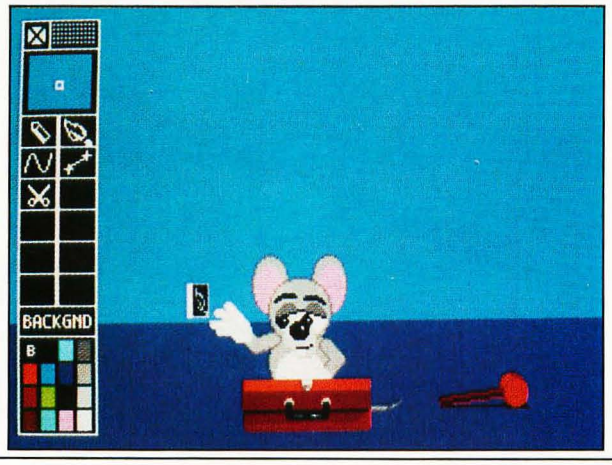
Vince suggested that maybe Tiny should pilot an airplane over the logo and drop the letter like a bomb. I almost rejected this outright, because there was no room above the logo. But then I decided to try it. To show the plane we'd pan up so the logo would be at screen bottom, and pan down again when the letter dropped. The only problem was to make it absolutely clear that the "camera," and not the logo, was moving. To facilitate this, clouds were added during the pan so the viewer's eye would have something to relate the logo to. A simple solution, and no one has ever mistaken the pan for anything else.

From Hand to Mouse

With the completion of Tiny and a number of miscellaneous animation sequences (including an example of a "crude" paint program called "NEO-Lith, Version 40,000 B.C."), it was time to roll out the star of the show. Oh, and I'd given him a name too: "Megabit Mouse" (or just plain "Mega" for short).

I had lavished a lot of care in designing Megabit's parts. Tiny's design had forced me to keep his movements kind of stiff, because of thick limbs and other design flaws. These did not affect Tiny, as he was intended to be kind of "stuffy," but Megabit was supposed to be emotional, flamboyant and flexible. His design reflected the experience with Tiny. Mega's limbs were thinner, allowing for greater range of movement. Thus, more poses were created. His feet were bigger and more cartoony. Best of all, his face was far more expressive, this primarily due to wide, well-defined eyes. (Tiny's eyes were usually shut, and you never see his mouth.)

All along I'd feared that I wouldn't be able to do a decent job animating a mouse's tail. I find freehand drawing on a computer difficult, even with a good graphics tablet. Drawing them any other way resulted in their looking stiff. I solved this problem by drawing dozens of tail positions on paper, then tracing those positions onto a graphics tablet. A little cleanup, a *lot* of reorganizing, and I had my tails. Good thing too, because the tail really ended up accenting Mega's expressions. It straightens when he's startled, twitches when he's mad, and so forth.



Megabit's first appearance is in a scene where a bunch of graphic bits fly together and, *bang!*, form a mouse. This was simple. I posed Mega in his "formed" position, then duplicated the frame. I went back to the first one and pulled Mega apart, scattering his components to the screen edges. A quick three-frame Tween (to create in-between positions) and, *voila!*, instant mouse! After that, all he had to do was look around, then get turned into a paintbrush.

The following shots were not so simple. The next sequence has Mega carrying in a tool chest, tossing the Art Director function icons into the on-screen "toolbox," as well as getting smacked by an alert box, dragged across the menu bar by the pointer, dropped on his butt, magnified, and knocked down by the toolbox. Finally, he shatters the toolbox with a hammer!

Next shot: Mega's colors are altered, and antlers are added to his

head to change him from "A MOUSE" to "A MOOSE." He grabs the eraser from the toolbox to erase the antlers, is poked by the cursor and erases his own face, then fumbles for the UNDO button. Once his face is restored, he is trapped in a window, filled around, scraped through so we can see his skeleton and finally spun. He dizzily collapses, stars spinning over his head.

Then he is flattened against the inside of the monitor glass, knocked around by the Film Director user interface, after which he walks into a Saturday morning cartoon and decks a Smurf-type character. Following that he runs a projector to show the miscellaneous animation.

Animating Mega wasn't much different than animating Tiny, but it was more difficult in some ways. Mega's design was far more complex, averaging 25 to 26 patterns per frame as opposed to Tiny's 12 (which also necessitated running Film Director's playback at full speed just to keep everything running smoothly). Mega's parts filled four entire screens of shape data and consisted of approximately *600 patterns!* Imagine trying to pick just the right hand out of 220 possible choices!

(A comparison: *Aegis Animator's cel function saves to disk each image clipped from a page of artwork. To animate Megabit using Aegis would require saving over 600 cel files to disk! And* you would have to try to remember which of those 600 files was the one you wanted on a given frame, unless you wished to load by trial and error. This is why I consider Film Director to be the only true cel system available on the ST.)

Mega's tail was a mixed blessing. While it did add a lot to his appeal, it was a pain to work with. If I had it swaying from side to side and needed to add or delete a few frames during that movement, I usually ended up having to adjust the tail over the next 20 to 30 frames just to get everything smoothed out, otherwise the movement would be uneven or jerky.

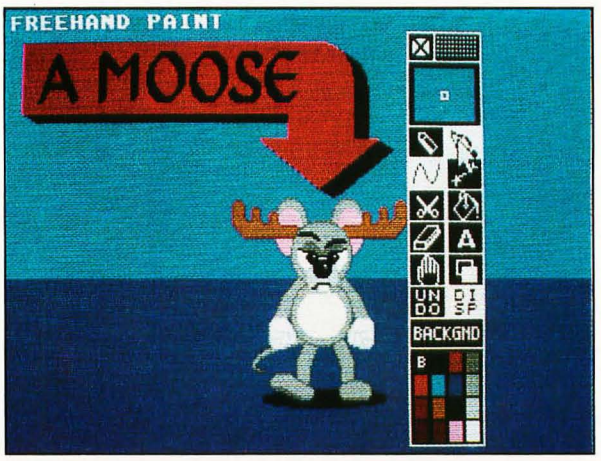
To further complicate matters, these shots required me to simulate GEM drop-down menus, alert boxes, mouse pointers, the Art and Film Director toolboxes, and on and on!

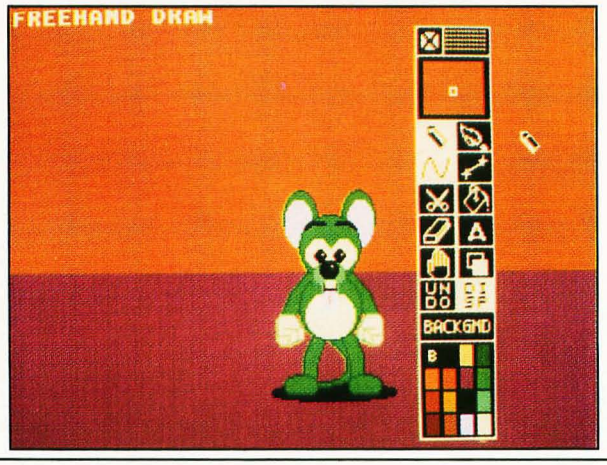
How long did it take? Even with fully completed shape tables and all patterns and groups defined, it still took an average of two 16-hour days to animate each of the longer sequences (averaging 1.25 minutes each). Mind you, this was two days of just putting cels on the screen and manipulating them. No drawing, just animating.

Mouse Wreckers

Fortunately, Broderbund had decided to skip having the demo for June CES and wanted it for later in the year. It was agreed that I would present a "Director's rough cut videotape" to Steve Friedman by September 20, 1987, that he would deliver to Broderbund for review.

The challenge of getting the animation on tape again reared its ugly head. I couldn't just borrow any RF/composite-equipped 520ST, for most of the animation I'd created required one megabyte of RAM to run. My old mid-1985 edition 520ST, upgraded to one meg, had no RF output and thus no composite video line to take to a VCR. No RGB-to-composite converters could be found either. The situation looked desperate, but my faithful ST dealer came to my rescue. His tech swapped out the old Revision B motherboard in my 520ST for a new Revision H board complete with RF modulator, moved my RAM expansion board onto it, and only charged me $20!





In spite of all my last-minute rushing, it became clear that the final scene wasn't going to be ready by the deadline. I had hoped to get access to some Genlock equipment to composite Megabit onto some video of a real computer desk. That hadn't happened. The rough cut had to be received before Steve left for Europe on September 21st, so I couldn't delay. On the 16th I dumped all the animations (except the ending) to VHS videotape, rough-dubbed the Bach music and shipped it off. I expected comments in a few weeks, followed by a round of final corrections (and the completion of the ending), and then the okay to go to the final tape. I was in for a big surprise.

Shortly after Steve's return from Europe, Broderbund cancelled the programs. They would not ship after all, meaning they no longer needed the video.

Mouse Warming

Broderbund and I came to terms where they relinquished all rights to the video to me, and I did not demand any further payment. My hope was that some other publisher would pick up the programs and want the video.

Months later, in early 1988, Epyx decided to pick up the programs *and* the video. A contract was signed, and I had to get back to work and complete it. The final shot was still needed. Trouble was, I had soured to the original ending. While visually interesting (Megabit jumping out of the monitor and shattering the camera lens with a disk), it ended on a negative note (Mega very angry), not a funny one. I felt this was not right.

Finally, I decided that when the projector bulb burned out, the narrator would ask Mega what they should do next. Mega would think of things which the narrator would reject. Finally, Mega would pull out a disk and toss it to the narrator, whose hand would appear on screen to catch it. The disk contained the end credits, and that's what they would show. But, just before those ran, Mega would *again* be clobbered by an alert box (which

would state "The End") and have the antlers placed on his head one last time. But, rather than getting mad, he'd just shrug as the "iris" closed to black.

Bingo! The shot allowed for some funny interaction between the narrator and Megabit, and it ended on an upbeat note. The only potential problem was how to have a near-life size digitized human hand come into view without the animation slowing to a crawl. A little cel trickery was all that was needed, and the video was done.

Well, almost. The Smurf-type character was *too* Smurfy for Epyx, so I had to change it. Also, as the programs had been ported to the Apple IIGS, I had to remove all ST-specific references. The title was also changed from "Notions in Motion" to "Art & Film Director" to make it clear what the tape was.

A final VHS rough cut tape was made. Then, as I tried to get the narrator's voice and the music recorded, I was informed that the tape had to be ready for the Applefest in Boston (May '88)—in less than two weeks! Furthermore, the rough cut had come out to 14 minutes—fully twice my originally planned length! Epyx wanted no more than 12.

A mad scramble was undertaken. Final changes were made, the soundtrack was recorded, paperwork was prepared for the video editor. I packed up a 1040ST, my hard drive, and my (then just arrived) Video Key and headed out to dump the animation onto three-quarter-inch videotape. All was taped, and everything was packed up and shipped off to the editor (Jim Yocom, at the Indiana Vocational Technical College) via overnight air freight.

After much straining from all involved parties, the editor's rough cut tape made an appearance at the Applefest in Boston. (The funny thing is, not a *scrap* of the video had been created on a IIGS!) A few weeks later, the release version of the tape made its debut at June CES in Chicago—a full year after the thing was originally planned to be finished.

Touch and Go

How is the final product? Personally, I was a bit disappointed. A last-minute order forced cutting two entire minutes of material, taking it from 12 to 10 minutes, throwing the pacing off in a few places. Also, due to the last-minute rush, there was no time for fine-tuning. Some lines of the narration needed a little more emphasis, and some of the sound on the animation was deadened in a few spots. Also,

the original music which appears over the credits does its creator (Randy McClanahan) no credit. The music degraded over the course of rerecording, dubbing, etc., so it sounds uneven and a little off-tempo in places, although the original recording sounded just fine.

Not to say it came out badly. Indeed, considering the touch-and-go status of the project and the crash effort it took to get it finished, it came out very well. I particularly want to take a moment to mention some of the people who helped make it possible: Stephen Friedman of Software & Video arts for negotiating the deals and seeing the thing through; Randy McClanahan for music and the caveman's "Ugh!" (yes, that's him!); Mrs. Andy Morrison for narrating the prerelease cut; Vince Reynolds for his help in producing parts of the animation; Digital Vision for providing a color Computereyes digitizer; Jim Yocom for doing such a good job editing a less-than-perfect master and getting it to look so good in so little time; and, finally, Madeline Canepa and Joe Miller at Epyx for their help in getting it done at last!

Believe It or Else

And thus ends the story of how the video came to be. However, there is a subject I've avoided until now, and it is the real core of this whole matter.

At the Worcester Atari Show (in Worcester, Massachusetts, October, 1987), just before Broderbund dropped Art and Film Director, I gained some interesting and important insights. At the show, a lot of CAD-3D and Cyber animations were displayed, including an eight-minute, four-megabyte animation by Tom Hudson called "Spider Patrol." The odd thing was that people tended to look at those flashy demos, say "neat," and just go about their business.

Meanwhile, just for laughs, ST-Log's editor, Clayton Walnum, and I set up a 1040ST at the ANALOG booth, and I popped in the animations of Megabit (not the video tape; the actual computer animations). I left one of those running as I went around the show, and whenever I went by the ANALOG booth I was flabbergasted to see people crowding around that little monitor, watching the animation not just once but *several* times, and bringing other people over to see it. I am not attempting to toot my own horn here. What happened was that I hit on something that is fairly obvious, but most people don't realize.

To wit: Here we are, with a wealth of powerful graphics tools on a system within the price range of almost anyone. And yet, the stigma of "cold" computers nicely fits the majority of animations and graphics created on them. They are technically intriguing, but the designers are so caught up in doing the neat technical trick that they don't make the work interesting—they don't make their graphics *involving* to those who view it. Megabit Mouse drew an awful lot of attention, and he really isn't doing anything phenomenally complex, nor is he even that impressive graphically. However, he *does* radiate a definite personality: he reacts to his situations, and, most importantly, he communicates these attitudes to the viewer. To paraphrase Joe Adamson: Megabit Mouse does not exist! Yet he lives!

Think about that a moment. We are often so wrapped up in making computers "productive," with word processors, databases, et al., that we miss many of the other tasks they are beautifully suited for. What I learned is that it is possible to make what is really nothing more than a bunch of colored dots on a screen be more than just technically impressive, but actually elicit *reactions* from an audience!

That, my friends, is one small example of the *real* power computers give us.

Author's Note: In honor of all those old Warner Brothers cartoons that inspired me during the making of the Art & Film Director video, I used the names of actual Warner cartoons as subheadings in this article, and also for the title—so don't blame me for the bad puns! ■

When not writing articles for ST-Log, Maurice Molyneaux designs game graphics, consults for software companies and creates animated cartoon productions using microcomputers. Despite a ridiculously French name, he claims to having been born in Vicenza, Italy, and denies vicious rumors that he eats escargots and calamari while computing. His DELPHI username is MAURICEM.

(from page 39)

```
      Psx=Psx-1
      If Psx<1 Then
        Psx=1
      Endif
      Mid$(Appt$(Chosenbox,Psy),Psx,1)=Chr$(95)
      Gosub Prntit
    Endif
    If Lkey=19200 Then  !left arrow
      Gosub Oldprint
      Psx=Psx-1
      If Psx<1 Then
        Psx=1
      Endif
      Gosub Prntit
    Endif
    If Lkey=19712 Then  !right arrow
      Gosub Oldprint
      Psx=Psx+1
      If Psx>25 Then
        Psx=25
      Endif
      Gosub Prntit
    Endif
    If Lkey=20480 Then  !down arrow
      Gosub Oldprint
      Psy=Psy+1
      If Psy>21 Then
        Psy=21
      Endif
      Gosub Prntit
    Endif
    If Lkey=18432 Then  !up arrow
      Gosub Oldprint
      Psy=Psy-1
      If Psy<1 Then
        Psy=1
      Endif
      Gosub Prntit
    Endif
    If Lkey=21375 Then !Delete
      For Lpp=Psx To 24
        Mid$(Appt$(Chosenbox,Psy),Lpp,1)=Mid$(Appt$(Chosenbox,Psy),Lpp+1,1)
      Next Lpp
      Mid$(Appt$(Chosenbox,Psy),25,1)=Chr$(95)
      Print At(6,Psy);Appt$(Chosenbox,Psy)
      Gosub Prntit
    Endif
    If Lkey=20992 Then !Insert
      Dm$=Appt$(Chosenbox,Psy)
      For Lpp=Psx To 25
        Mid$(Appt$(Chosenbox,Psy),Lpp,1)=Mid$(Dm$,Lpp-1,1)
      Next Lpp
      Mid$(Appt$(Chosenbox,Psy),Psx,1)=Chr$(95)
      Print At(6,Psy);Appt$(Chosenbox,Psy)
      Gosub Prntit
    Endif
    If Lkey=18176 Then !Clear
      Appt$(Chosenbox,Psy)="_________________________"
      Print At(6,Psy);Appt$(Chosenbox,Psy)
      Gosub Prntit
    Endif
  Return
  Procedure Dummy
    ' dummy procedure for turning off ON MENU commands temporarily
    Menu Off
  Return
  Procedure Clearappt
    For Lp1=1 To 31
      Astatus(Lp1)=0
      For Lp2=1 To 21
        Appt$(Lp1,Lp2)="_________________________"
      Next Lp2
    Next Lp1
  Return
  Procedure Prntit
    Dpoke Contrl,32
    Dpoke Contrl+2,0
    Dpoke Contrl+6,1
    Print At(Psx+5,Psy);Chr$(32);
    Dpoke Intin,4  !reverse transparent--looks like a cursor
    Vdisys
    Print At(Psx+5,Psy);Mid$(Appt$(Chosenbox,Psy),Psx,1)
    Dpoke Intin,1  !back to replace mode
    Vdisys
  Return
  Procedure Oldprint
    Print At(Psx+5,Psy);Mid$(Appt$(Chosenbox,Psy),Psx,1)
  Return
  Procedure Respondbutton
    Ys=90+45*(Rez=1)
    Ye=150+75*(Rez=1)
    If Mousex>250 And Mousex<320 And Mousey>Ys And Mousey<Ye Then
      Graphmode 3
```

```
      Pbox 250,Ys,320,Ye
      Graphmode 1
      Alldone=True
    Endif
    Ys=190+95*(Rez=1)
    Ye=250+125*(Rez=1)
    If Mousex>250 And Mousex<320 And Mousey>Ys And Mousey<Ye Then
      Graphmode 3
      Pbox 250,Ys,320,Ye
      Defmouse 2
      Gosub Outputprint(Chosenbox)
      Defmouse 0
      Pbox 250,Ys,320,Ye
      Graphmode 1
    Endif
    Ys=290+145*(Rez=1)
    Ye=350+175*(Rez=1)
    If Mousex>250 And Mousex<320 And Mousey>Ys And Mousey<Ye Then
      Graphmode 3
      Pbox 250,Ys,320,Ye
      Graphmode 1
      For Lp=1 To 21
        Appt$(Chosenbox,Lp)="____________________________"
        Print At(6,Lp);Appt$(Chosenbox,Lp)
      Next Lp
      Gosub Prntit
      Graphmode 3
      Pbox 250,Ys,320,Ye
      Graphmode 1
    Endif
    Yss=0
    Yee=336+168*(Rez=1)
    If Mousex>40 And Mousex<240 And Mousey>Yss And Mousey<Yee Then
      Cx=Int((Mousex-40)/8)+1
      Cy=Int((Mousey/(16+8*(Rez=1))))+1
      Gosub Oldprint
      Psx=Cx
      Psy=Cy
      Gosub Prntit
    Endif
  Return
  Procedure Gtnewpath(Pt$)
    L=Len(Pt$)
    Tp$=""
    While (Tp$<>"\" And Tp$<>":" And Tp$<>"/")
      Tp$=Mid$(Pt$,L,1)
      L=L-1
    Wend
    Path$=Left$(Pt$,L+1)+"*.*"
  Return
  Procedure Outputprint(Which)
    Lprint
    Lprint
    Lprint Bon$+T$+Boff$
    Lprint
    Tme=7
    For Lp2=1 To 21
      Tm$=Str$(Int(Tme))
      If Tme<10 Then
        Tm$=" "+Tm$
      Endif
      If Even(Lp2)=-1 Then
        Tm$=Tm$+":30"
      Else
        Tm$=Tm$+":00"
      Endif
      Tme=Tme+0.5
      If Tme=13 Then
        Tme=1
      Endif
      Lprint Tm$+" "+Appt$(Which,Lp2)
    Next Lp2
    For Lp2=1 To 8
      Lprint
    Next Lp2
  Return
  Procedure Checkstatus
    For Lm=1 To 31
      St=False
      For Lpp=1 To 21
        If Appt$(Lm,Lpp)<>"____________________________" Then
          St=True
        Endif
      Next Lpp
      If St=True Then
        Astatus(Lm)=1
      Else
        Astatus(Lm)=0
      Endif
    Next Lm
  Return
```

END

■ **by Bob Retelle**

Without giving away too much that might detract from your enjoyment of the game, let's take a tour of the dungeon, check out some of the more difficult puzzles, and introduce you to those charming monsters.

The first level you encounter is where the Hall of Champions is located. Here is where you'll select the party of four brave souls that you will guide down into the dungeon. Once you've assembled the group, and found the exit into the dungeon, the game begins. This level is safe from danger and merely serves to introduce you to the things you'll see later on. You'll find food and supplies and learn how to work the doors. Your first magical-spell scroll is here, as is an Altar of VI where you can resurrect the bones of party members who are unlucky enough to die.

Unlike the first level, which is well lit, the second level is dark and gloomy until you either light a torch or use magic to create some light. From this point on you'll have to maintain your own source of light. This is important because there are many tiny details to watch for and many items which may be hard to see in poor light. Grey keys on the grey floor are one example.

This level is still mainly an introduction to the dungeon. You'll fight your first monsters here and learn how to use the weapons you've found. There are many puzzles on this level, but most of them are easy to solve and just serve to show you the kinds of things you'll be encountering later. If you run into anything which seems to stop you cold, try getting *very* angry with it. The keys on this level are all out in plain sight, but as mentioned before, you'll have to watch very closely to find some of them.

Be careful of falling into pits while your party is still fairly weak. While there are some items of interest in pits, the damage you'll take from falling might be fatal. On this level you'll fight Screamers and Mummies. The Screamers are slow moving and not too dangerous if you use the strategic retreat, or "jump forward, hack, and jump back" method of fighting them. The Mummies, however, are fast, so you'll have to retreat quickly to avoid being badly injured. Remember to throw weapons at them!

The third level gives you more difficult puzzles to solve and more dangerous creatures to fight. The Blue Trolls with their nasty clubs can be fatal if you let them hit you. Get them from a distance if you can. The Rockpile Monsters on this level are slow, but hard to kill. Even worse, their bite is poisonous. The poison is only temporary, though, so you don't necessarily have to use a "Cure Poison" potion. Again, throw things at them, and use the strategic retreat.

Another thing this level teaches you is the value of slamming a door in a monster's face and resting safely on the other side. You can always retreat into the large central room and sleep to regain your health and mana, as long as you remember to close *all* the doors.

This level has two puzzles which require quick movement. Be sure none of your party is carrying too much to move quickly. Their LOAD indicators should not be yellow. If any of them are, have them drop items.

If you find a compass, use it in the Matrix room to show you the action of the invisible "spinner" fields. In the Chamber of the Guardian, persistence will pay off. You'll need to search the large central room to find something you can use to pay for entrance to the Vault; then you'll need to use magic to open a door you can't reach. Once the door is open, you'll see what you need to do to close the pit in front of the door.

The Time Is of the Essence room is the second area where you need to move fast. Where it says "Hit and Run," do exactly that to the button on the wall. If you run to your left fast enough, you'll see what to do next. The Room of the Gem is fairly straightforward, once you've used something to close the pit; but watch out behind you, as another action you take may trap you! Also, be *very* watchful around corners, or you may find yourself sandwiched between pairs of Rockpile monsters! This room gives you the opportunity to drop monsters into a pit, something that while not giving you any experience points, is at least satisfying to do.

Creature Cavern is just what the name implies. Be ready to do a lot of fighting in there! You will need at least four gold keys to proceed to the next level and a fifth one to open a room near the end.

Just when you think you've learned to survive whatever the dungeon can throw at you, level 4 feeds you to the Purple Worms. This may in fact be the nastiest level of the entire dungeon. Once you get through the Purple Worms—assuming you

(to page 64)

ANCIENT

Illustration by Steve Sterling.

CORRIDORS

Part 3 · A Guide to Dungeon Master

Line

Numbers part 2: Multiple-precision multiplication and multiple-multiple-precision division

For example, suppose you have the binary number 1101. Now let's imagine that this number is held in two bytes of two bits each. The first (high) byte contains 11, and the second (low) byte contains 01. If we want to print this number as a string of hexadecimal digits, there's no problem: we convert the first byte (separately) to 3, and the second byte (again, separately) to 1. But what if we want to convert the number to a decimal string? Binary 1101 equals decimal 13, and there's no way you're going to come out with this result by converting the bytes separately.

No matter how many separate locations binary 1101 is contained in (well, as long as it's not more than four, at any rate), there's no great difficulty in converting it to decimal ASCII on the 68000, since the value 13 is well within the limits of its *divu* (divide unsigned) instruction. But our 64-bit products are much too big for *divu*, and, to make matters more difficult, they're bigger even than a data register—twice as big, in fact. So our 64-bit division routine will have to be able to divide two numbers, each of which can be up to twice as long as the 68000's register length. When I first realized that I had to upgrade my 32-bit divide to 64 bits, I thanked heaven I had chosen the simple (i.e., stupid) way to do it the first time. The modifications weren't hard at all.

Assembly

by Douglas Weir

Well, well, well. We've successfully multiplied two 32-bit numbers. Now what do we do with the product? First of all, how do we print it out? It doesn't take long to realize that, in order to convert a 64-bit number to an ASCII string, we need more than the 32-bit division routine in our original program; we now need the ability to divide 64-bit numbers.

Of course, it's easy enough to print separately the high and low longwords that *mul32* returns. But we can't just print these numbers next to each other to get the full double-longword result. We want to represent our product as a decimal number, but it's held in memory in binary form, and there's no direct correspondence between the binary and decimal digits.

Subtract and conquer

Although there wasn't space to discuss it last time, you've no doubt noticed that *div32* doesn't really do any dividing. Instead, one number is "divided" into another by simply subtracting it from the dividend until the dividend is either less than the divisor or equal to zero. At this point the number of subtractions is equivalent to the quotient from dividing the numbers, and what's left of the dividend is the remainder.

This is a perfect example of a "brute force" algorithm. There's an interesting characteristic of brute force approaches—they almost always work, and when they don't, you know why. You can't say that about "complicated" algorithms, and "real" long division in assembly language is definitely complicated.

Take a look at *div64* in this month's listing (the entire program has been reprinted, since the 64-bit upgrade made other changes necessary too).

To subtract two double-longword values, you begin by subtracting the two low longwords and storing the result. There may be a carry generated as a result of the subtract; this is the equivalent of a "borrow." If, for example, you were to subtract decimal 17 from 24, you would begin by subtracting 7 from 4. Since 4 is less than 7, you borrow a 1 (actually a 10) from the next decimal place to the left, and make the 4 a 14. Now 14 minus 7 is 7; you write the 7 down and subtract the next column, and so on.

A borrow can occur in exactly the same way in a 68000 data register. When this happens, the Carry bit in the Condition Codes register is automatically set to 1. You can thus subtract numbers a hundred bytes (or words, or longwords) long, simply by subtracting byte by byte, checking the Carry bit after each subtract, and subtracting an extra 1 from the next byte whenever a borrow has occurred. Why the extra 1?

Because when a borrow "occurs" in the 68000 (or most other processors, for that matter), it doesn't really happen—the Carry bit lets you know that it *should* happen. After all, the 68000 has no way of knowing the location of the next-higher unit of the value you're subtracting from. It could be a register or a memory location. So, when this situation occurs, the current subtraction is handled as though a borrow was completed, but it's up to you—the programmer—to make sure that the borrowed digit is actually subtracted from the "next" value. Understanding the mechanics of subtracting a larger from a smaller binary value requires learning about signed binary numbers, and we'll leave that for another time.

Since we only want to subtract double longwords, all we have to do is subtract the low longwords, check for a carry, and (if the Carry bit is set) subtract an extra 1 when we subtract the high longwords. However, there's an even easier way to do this on the 68000.

Extending a helping hand

The designers of the 68000 wanted to make it as easy as possible for programmers to implement multi-precision arithmetic operations. There are times when it's not convenient to detect and react to a carry condition right away. However, we know that the condition codes are very volatile—the Carry bit is set or reset by most instructions. So instead of forcing the programmer to explicitly save the condition in such circumstances, the 68000 has a special bit in its Condition Codes register that behaves identically to the Carry bit, but only when certain instructions are used. The bit is called the Extend bit, and instructions that use or affect it always contain an *x* in their names. For example, *subx*: subtract with extend.

"To subtract two double-longword values, you begin by subtracting the two low longwords and storing the result."

A *subx* has exactly the same effect as a plain *sub*—it subtracts the source from the destination operand, and puts the result in the latter. The difference is twofold: first, the Extend bit is set or cleared in exactly the same way as the Carry bit (which is still affected as before, by the way), and it won't be altered until another "extend" instruction is executed or the programmer explicitly changes the contents of the Condition Codes register. (We'll see how this works in a moment.) You can execute a *subx* and then execute any number of other instructions and be sure that the carry condition will be saved in the Extend bit until you need it.

The second difference in *subx* is that the carry from the previous *subx* will automatically be incorporated in the next *subx*. In other words, if a subtract operation generates a borrow, then the next subtract will subtract both its source operand and the borrowed bit from its destination operand, if both operations are executed with *subx*. It seems that all of our work will be done for us.

There's one hitch with the extend instructions, though. You are always restricted to much fewer addressing modes when you use them. For example, *subx* will allow you only to subtract a data register from another data register, or a value in memory from another value in memory, both of which must be accessed with the predecrement addressing mode. This won't be a problem for us here: our operands are already in data registers.

Since the effect of any *subx* always depends partly on what the value of the Extend bit already is, we would like to be sure that this bit has a value of zero when we do the first subtract—otherwise a spurious borrow could be subtracted the first time around. There is a way to write values into the Condition Codes register, but it's a rather roundabout way.

Logical operations

Yes, I know—all my operations are logical. But what I have in mind here is a special kind of logic whose rules have a lot to do with the way computer hardware is implemented.

Suppose we want to know whether a statement (we'll call it X) is true or false. Suppose further that we know that, if statement *A* is true or if statement *B* is true, then *X* is true. Now all we need to know is the value of *A* or *B*. If *A* is true, then *X* is true, no matter what the value of *B*. And if *B* is true, then again *X* is true, no matter what the value of *A*.

On the other hand, it's easy to imagine a situation where both *A* and *B* have to be true in order for *X* to be true. In this

case, if we know that *A* or *B* is true, then we don't know enough: everything depends on whether the other statement, *B* or *A*, is true also. But if we know that *A* or *B* is false, then we can stop right there: *X* must be false also, since both *A* and *B* must be true if *X* is to be true.

Now let's substitute the values 1 and 0 for the rather abstract concepts of "true" and "false." If we have three variables, called *A*, *B* and *X*, then we can rephrase our rules as follows. In the first case we can say:

If A = 1 OR B = 1 then X = 1

And in the second case we can say:

If A = 1 AND B = 1 then X = 1

Now let's make up a little table showing all the possible values of the three variables for the two relationships:

A	OR	B	X
1		0	1
0		1	1
1		1	1
0		0	0
A	AND	B	X
1		0	0
0		1	0
1		1	1
0		0	0

This is not as trivial as it looks. Semiconductor "logic" is implemented by complex combinations of transistors that "gate" electrical current through various paths according to these rules (and others). There are also 68000 instructions that operate on registers and memory locations according to these rules, and that's what we're interested in here.

Think of *A*, *B* and *X* not as variables but rather as bits in a binary representation of data. The 68000 instruction *andi.b #$01,d4* will take the immediate byte value 1 hex (i.e., a byte with only its rightmost bit set) and match it with the low byte in register d4. Now the bits in the two values will be compared according to the logical relationship specified (in this case, AND), and the result will be put in the destination operand, register d4. In other words, bit 7 of the source operand will be ANDed with bit 7 of d4, bit 6 with bit 6, and so on. We don't know what's in d4, so we can't say definitely what the result of executing this instruction would be, but we can say that d4 will have to end up with either the value 1 or 0, since the zeroed upper bits of the source operand prevent any of those bits from being 1 in the result.

Changing conditions

We need to know about the 68000's logical operations here because these are the only instructions that allow us to write to the Condition Codes register. We want to make sure that the Extend bit is 0 before we execute our first *subx*, but we don't want to affect any of the other bits. According to the rules above, if we AND a 1 with a 0, the result will be 0; and if we AND a 1 with a 1, the result will be 1. Also, if we AND a 0 with anything, the result will always be 0.

So we want to AND an immediate value with the Condition Codes register that will contain 1s to be matched against all but the Extend bit. The Motorola manual tells us that the Extend bit is bit 4, so we want a binary value that looks like this (the Condition Codes register is only 8 bits long): 11101111. This is the same as hex EF. Actually, we don't have to worry about bits 5, 6 and 7: they aren't used. And since bit 4 is 0, we can shorten the number to binary 1111, or hex F. The instruction would then be written as *andi #$0f, ccr*. There's no size specifier, since operations on the Condition Codes register are automatically byte size. This instruction, when executed, will clear the Extend bit and leave the others alone.

"A *subx* has exactly the same effect as a plain *sub*—it subtracts the source from the destination operand, and puts the result in the latter."

Dividing 64 bits

Aside from these new instructions, *div64* is fairly straightforward. Since we're dealing with double-longword values, we have to work with register pairs for the operands and for the "subtract counter," which will contain our quotient. The routine begins by making sure that the divisor isn't zero. If it is, then it loads register d4 with an error code and returns to its caller. Note that none of the "return registers" (d0 through d3) are used for the error code. Now that we're dealing with all possible 32-bit values as valid return values, we have to reserve a separate register for any return codes.

If all is well, then the main loop is entered. The numbers must be compared to see if the dividend is still larger than the divisor—when it isn't, or when it becomes zero, then we're finished. The subtract itself is accomplished with two *subx* instructions—one for each longword. Each complete subtract is counted in d5; when there's a carry out of d5, it's added explicitly into d4. Since we have a pair of data registers (d6 and d7) that aren't being otherwise used, we could have used a pair of *addx* (add with extend) instructions instead by loading, say, d6 with 0 and d7 with 1 at the beginning. Then we could do the following:

```
addx.l  d7,d5  count subtract
```

First the 1 in d7 is added to d5. Then the 0 in d6 is added to d4. If there was a carry out of the first add, then the Extend bit was set, and it will be included in the second *add*. There's no need to clear the Extend bit before this operation: if our comparisons at the top of the loop are correct, then there's no possibility that the second *subx* could have generated a borrow. Instead of using *addx* here, however, I thought I'd show how the operation is done explicitly.

Finally, the remainder and quotient are transferred to the correct return registers, a "success" code is loaded into d4, and *div64* returns to its caller.

ASCII you shall receive

Of course, the divisors table had to be expanded to allow a 64-bit conversion, and *cv64*, the conversion-to-ASCII routine, is changed also as a result. Most of the differences between it and *cv32* from last time involve the data registers used for certain operations, and the new double-longword table.

The algorithm used here for ASCII conversion is simple. The number is divided by decreasing powers of 10 (beginning with a number equal to a 1 followed by 19 zeros—I'm not sure what it's called!). The quotient from each of these operations is a number from 0 to 9, which is then converted to ASCII by adding it to the ASCII value for zero; the remainder is then divided by the next-lower power of 10 to get the next digit, going from left to right, and so on, until 10 to the power of 0 (i.e., 1) has been used as a divisor,

which yields the rightmost digit. The powers of 10 used by *cv64* come from the divisors table.

The routine begins by writing spaces to the string area in *dgts*. Then it checks to see if leading zeros are to be written into the result. Register d6 gets the result of this test. You can think of d6 as a flag that tells whether the converted number is, so far, a zero (TRUE) or non-zero (FALSE). In the former case, spaces are written instead of leading zeros; otherwise, zeros are written whether leading or not. If leading zeros are not desired, then d6 is immediately set to TRUE: the number is considered to be a zero until a non-zero digit occurs. Otherwise, d6 is set to FALSE, the number is considered to be non-zero from the start, and consequently all zeros are written. At the end of the loop, if d6 is TRUE, then the number was a zero and only spaces have been written so far; so a single zero is written in the last byte of the string.

The digits are converted leftmost first, and the converted bytes are written to the string with the Address Register Indirect with Index mode. Register a0 points to the end of the string. Register d5 is loaded with the length of the string; this value is then negated with the *neg* instruction. When the negative value in d5 is added to a0, the result is the address of the first byte in the string; subsequent additions to d5 point further into the string; when d5 contains 0, the end of the string has been reached.

Back to numbers

The routine *cvnbr* handles the ASCII-to-number conversion. It has been changed slightly since last time in order to work with the new version of the factors table and also to use d1 to return a possible error code. The numbers to be multiplied can only be up to 32 bits long, so our task here is a bit easier. The routine begins by doing a crude check on the size of the string it's received. Register d1 is loaded with the maximum size allowed plus 1, and is then used as a counter as the string is indexed through until (or if) a null is found. The *dbeq* instruction will get us out of the loop as soon as a null is found or when d1's value becomes -1, whichever happens first. If it's the latter, then the string was too long.

(If you remember our discussion of the *dbcc* family of instructions several episodes back, then you know that they decrement their counter to -1, not to 0, before terminating. We loaded d1 with a value 1 greater than required, which would seem to set up for two extra iterations of the loop. However, we branch immediately to the *dbeq* the first time around, which uses up one of the extra iterations; and the null terminator, which is counted, uses up the other.)

Although this test will catch any input number longer than 10 digits, there are some illegal numbers it won't catch: any 10-digit number greater than 4,294,967,295 to be exact. Numbers larger than this are too big to be held in a 32-bit register; multiplying them with this program will give a result of zero. There are various ways these values could have been checked for, but I didn't want the routines to get too long.

After this, it's just a matter of reading each digit from the string (starting with the leftmost), multiplying it by the next factor from the factors table, and adding the product to a cumulative total. We know that our product here will be much smaller than even 32 bits, so the high longword is simply discarded. The table of factors is simply the divisors table reading in the opposite direction; register a1 is decremented to index through it. Since *divisors* has double-longword values but only longwords are needed for *factors*, a1 is decremented an extra time to skip each extra longword. Converting an ASCII digit to a number is only a matter of subtracting the ASCII value of zero from it.

"Typing in a number larger than 10 digits will end the program with an error message; otherwise, you can end it by pressing Control-C."

Carried away

Only one instruction has changed in *mul32*, but it's an important one. The instruction just before the label *m_nxt0* adds in a carry from the previous addition to the product. Up until now I'd never used the high longword returned by *mul32*; I converted some random values by hand, and they seemed to be correct. But as soon as I was able to look at a lot of 64-bit conversions, it became clear that there was a subtle error in the way I was handling the carry.

Remember that a carry out of a data unit means that the bit can't be represented in that area: the number is now too big for it. As *mul32* points back through the product area to add in successive partial products, it does point to the new memory area, which is where the carry should go; however, the indexing is done by words (i.e., two bytes backward at a time), whereas the partial products are added in as longwords. In other words, each new partial product "overlaps" with the old by a word. This is the way it should be—that's the way the numbers are added. However, it means that the carry shouldn't be added directly into the partial product as a longword—that would be the same thing as trying to stuff it back into the longword it just popped out of. Instead, it should be added into the first entirely free word (or byte, to be precise) in the product area.

With a0 and d0 pointing to the new partial product area, we can access the first non-overlapping word by writing to the address as a word rather than as a longword. If we tried this with a data register, it wouldn't work; we would be writing to the low word of the register. But in memory it's different: writing to a word at a given address will access the two bytes at that address and the next higher one; writing to the longword will access those two bytes and the next two as well, which are just the ones we don't want to touch. The new instruction *addq.w #1, 0(a0,d0.l)* adds in the carry correctly.

Why *jsr*?

I promised last time to explain why you can *jsr* but not *bsr* to a subroutine indirectly. The *bsr* instruction calculates the new address to be loaded into the Program Counter (PC) by using a number which is an offset from the current value of the PC. In other words, when a program is assembled, the assembler calculates for every *bsr* instruction the distance from it (or rather the next instruction after it) to the label in the instruction's destination field. The assembler can do this because both the value of the label and the location of the *bsr* are known to it, and the distance to (the "relative address" of) the destination is the difference between the two.

Jumping to an indirect address is different, however. The assembler knows only that the address will be in a certain register; it can't possibly know what that address will be, so it can't calculate the distance between the jump and the address. So this has to be an absolute jump; the *jsr* instruction simply loads its destination address into the PC, and—jumps.

But why do we need relative addressing at all? The answer is that it allows "position-independent" code—programs that can be loaded and run anywhere in memory, since what seem to the programmer to be absolute addresses—e.g., labels of subroutines—are actually represented as relative addresses in the program.

"Remember that a carry out of a data unit means that the bit can't be represented in that area: the number is now too big for it."

The rest of the program

The beginning of the program should be self-explanatory. GEMDOS function 10 (hex 0A), READLINE, is used to read the number strings from the keyboard. It's an easy way to input an edited string. To use it, you just push the address of a string area (more about this in a moment), then the function code, and call GEMDOS. The first byte of your string should contain a number (less than or equal to 255, of course) that tells GEMDOS the maximum number of characters to input. When the function returns, the second byte of the area will contain the number of characters actually input. If you add this number to the address of the third byte in the area (which is where the input characters begin), you will get the address of the carriage return (hex 0d) that ended the input. Note that if you want a null-terminated string, this is where you should insert the null; READLINE does not do this.

Typing in a number larger than 10 digits will end the program with an error message; otherwise, you can end it by pressing Control-C.

Now that we can divide 64-bit numbers, it seems only natural to multiply them. Of course, to check our result, we'd have to have a 128-bit divide routine. But it seems silly to stop at 128 bits—I wonder if IBM started this way?

Program listings begin on page 78.

INTRODUCING

VideoGames

And Computer Entertainment Magazine

12 ALL COLOR ISSUES ONLY $19.95

Save over $15 off the cover price!

- GAME REVIEWS
- ARCADE ACTION
- STRATEGY GUIDES
- TECHNICAL REPORTS
- COMPUTER SOFTWARE

VideoGames
And Computer Entertainment Magazine

P.O. BOX 16927, N. HOLLYWOOD, CA 91615

☐ **Yes!** Sign me up for 12 issues for only $19.95—I'll save over $15!

☐ Payment Enclosed — Charge My ☐ VISA ☐ MC

_____ EXP_____

SIGNATURE_____

NAME_____

ADDRESS_____

CITY_____ STATE_____ ZIP_____

Your first issue will arrive in 6 to 8 weeks.
WATCH FOR IT!!
Offer Expires 3/1/89 .

Money back on unused subscriptions if not satisfied!
Foreign—add $7 for postage.

CIYWA

BOOT UP TO BIG SAVINGS!

1 YEAR FOR ONLY $28

SAVE $14 OFF THE COVER PRICE

1 YEAR *WITH DISK* ONLY $79

SAVE TIME AND MONEY SUBSCRIBE TO ST-LOG

SAVE $14 OFF THE COVER PRICE WITH THE CONVENIENCE OF HAVING ST-LOG DELIVERED DIRECTLY TO YOUR DOOR *BEFORE* IT EVEN HITS THE NEWSSTANDS! GET THE MOST OUT OF YOUR COMPUTER

SUBSCRIBE TO ST-LOG TODAY!

☐ 1 YEAR @ $28 — SAVE $14! MCKWW
FOREIGN — ADD $7 PER YEAR
☐ 1 YEAR *WITH* DISK @ $79 DCKWW
FOREIGN — ADD $15 PER YEAR
☐ PAYMENT ENCLOSED ☐ BILL ME
CHARGE MY: ☐ VISA ☐ MC #_____

EXPIRATION DATE SIGNATURE

MONEY BACK ON ALL UNUSED PORTIONS OF SUBSCRIPTIONS IF NOT SATISFIED.

NAME

ADDRESS

CITY STATE ZIP

MAKE CHECK PAYABLE TO L.F.P., INC., P.O. Box 16928, N. Hollywood, CA 91615.
Your first issue will arrive in 6 TO 8 weeks. OFFER EXPIRES MARCH 1, 1989.

WATCH FOR IT!

Public Domain Software

Over **575** Disks Available for the ST
$4.00 Each

Christmas Special
Buy 6 Disks get 1 FREE
Buy 10 Disks get 2 FREE
Buy 20 Disks get 5 FREE

Utilities, Games, MIDI, Clip Art, Applications, Music, Educational, Graphics and more
24 Hour Shipping Telephone Support
Free Catalog Updates

Call or Write for FREE Catalog
(800) 622-7942

Clip Master

10 Screens per disk
$19.95 per disk
FREE Bonus Disk with Purchase
Dealer Inquires Welcome

Disk 1 : Animals, Christian, Folks, Food, Home, Music, Plants
Disk 2 : Americana, Patriotic, US Map, Outlines of all 50 States
Disk 3 : Arrows, Designs, Kid Stuff, Signs, Symbols, Christmas
Disk 4 : Animals, School, Thanksgiving, Transportation & more

Clip Art for Desktop Publishing Programs

Dungeon Master	$26.95	Omnires	$25.95	Spectrum 512	$49.95
DM Hint Book	$9.95	Turbo ST	$37.95	Labelmaster Elite	$29.95
DM Hint Disk	*$15.95*	Procopy	$24.95	Tune Up	$32.95
Olds	$22.95	Neodesk	$19.95	Mouse Master	$32.95
Typhoon Thompson	$24.95	Flash	$19.95	Monitor Master	$39.95
GFA Basic 3.0	$39.95	Shadow	$22.95	Helpful Hints (Book)	$16.95
GFA Reboot Camp	$16.95	Interlink ST	$29.95	*Supra 2400 Modem*	*$139.95*
Publisher ST	*$79.95*	Degas Elite	$45.95	**Call for More**	

VISA

BRE Software Dept. STL
352 W. Bedford, Suite 104
Fresno, CA 93711
(209) 432-2159 in CA

MasterCard

CIRCLE #112 ON READER SERVICE CARD.

ProCopy

ST BACKUP UTILITY

You can't backup your software because copy protection locks you out. **ProCopy** is the key!

- Protects against the accidental loss of expensive software
- Works with all Atari STs
- Copies both S/S & D/S disks
- Use with 1 or 2 disk drives
- Not copy protected
- **FREE** shipping and handling
- **TOLL-FREE** telephone number
- Updates available to registered owners
- Orders shipped same day
- Disk analyze function included

Dealer Inquiries Welcome

VISA MasterCard and C.O.D. orders

Call (800) 843-1223

$34.95

Send check for $34.95 (overseas add $2.00 for air mail) to:

PROCO PRODUCTS

P.O. BOX 665, CHEPACHET, RHODE ISLAND 02814 USA
(401) 568-8459

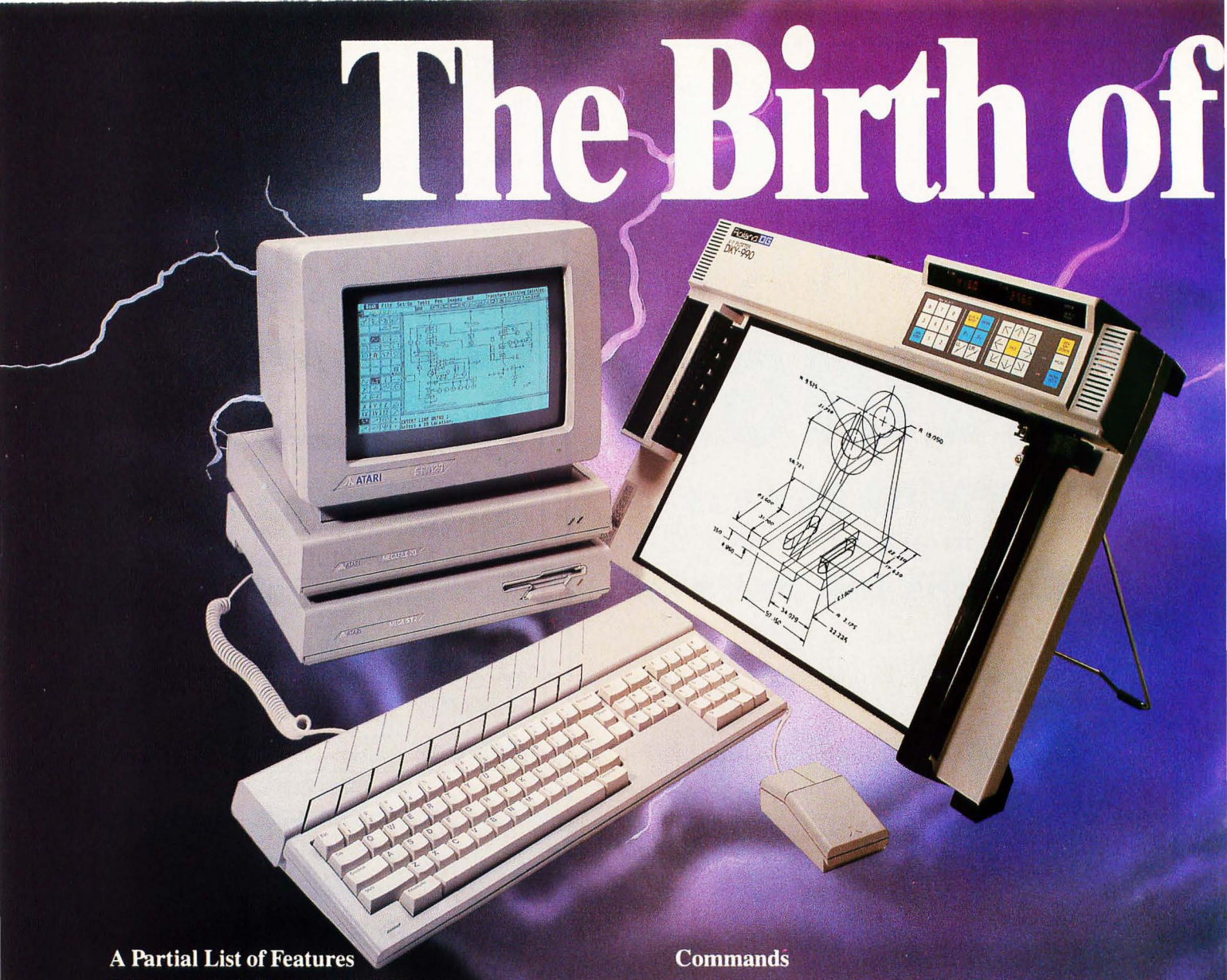
Available in Europe

THE MOVING FINGER CO.
Building 2
Shamrock Quay
Southampton, SO1-1QL
England
Tel. 0703-229041

TECH-SOFT COMPUTER WHOLESALE
324 Stirling Highway
Claremont
Western Australia 6010
Tel. 09-385-1885

CIRCLE #113 ON READER SERVICE CARD.

The Birth of



A Partial List of Features

Fully interactive 2 dimensional & TRUE 3 dimensional capabilities
Multiple 3D views can be opened and modified at any time
A user can work in any combination of views with all views instantly updating at all times
Automatic generation of ANY orthographic view including user defined auxillary views
Entities can be selectively hidden in any view allowing easy generation of true orthographically sound views
Extremely user friendly
Full GEM interface, pull down menus, mouse or keyboard commands, dialog boxes and more.
256 Layers can be activated invidually or in groups
9 zoom modes allow magnification's of up to 1,000,000 times
The ability to overlay other programmes from within DynaCADD

Dimensioning

Auto Dimensioning includes;
Mechanical and Architectural formats
Full 2D and 3D dimensioning is supported
Baseline, chaining, angular and linear dimensioning is supported
Automatic tolerancing in any of three different styles
Text orientation using any one of the three different systems (uni-directional, angled or aligned)

Commands

Sophisticated command nesting allows the following partial list of commands to be accessed at any time:
Zoom in or out
Zoom a window
Scroll or "Pan" the page
Center the page on a point
Multiple 3 Dimensional dynamic rotations at any angular increment
Rotate any 3 Dimensional view to a predefined co-ordinate plane
Zoom a 3 Dimensional view in or out...

Fonts

Full featured Vector Font Editor
Automatic proportional spacing can be activated or deactivated with the click of a mouse
Up to 16 extremely high resolution fonts can be active in any drawing with no need for repeated disk access

Insertion

Insert POINTS, LINES, CIRCLES, ARCS, FILLETS, ELLIPSES, ELLEPTICAL ARCS, TEXT, BOX, POLYGON, POLYFIGURE and SUBFIGURES using a wide variety of modifiers

a New Breed

DynaCADD™
Computer Aided Design and Drafting

Introducing DynaCADD™...a Professional CADD Solution at a Fraction of the Cost

DynaCADD is a 2D and *true* 3D Computer Aided Design and Drafting software solution that will turn your personal computer into a sophisticated and powerful CADD work station. With electrical, mechanical, architectural or civil applications DynaCADD provides a total design and drafting solution. DynaCADD revises, designs and details drawings in 2D or true 3D, reads and writes the industry standard DXF file formats and provides compatibility with the new Desktop Publishing solution, *Calamus*. DynaCADD proves itself invaluable from conception to output by offering an alternative solution to expensive, difficult to use 2D and 3D CADD systems. DynaCADD brings mainframe CADD capabilities to the CADD Professional at a fraction of the cost.

SPECIAL OFFER!!

For a limited time only
Save up to $150.00
off our introductory price of U.S. $695.00
Trade in your existing Atari ST CAD package and receive up to $150.00 in credit against the purchase of DynaCADD
Call us for details

Information Commands

DynaCADD gives you the ability to:
Measure distances, angles and perimeters
Verify location and style
List database extents, drawing parameters...

Entity Transformation

Translation between 2D or 3D positions, circular arrays, mirror, linear arrays, stretch, sweep, rescale, project, revolve, trim and divide
3 Dimensional entity transformation of any view into 2 Dimensional entities
Select entities for transformation & editing using:
Digitize, Windows, Windows Out, View Windows or Last...
By entity color
By entity layer
By chaining with direction and much more

Plotter Output

Both HPGL and DM/PL and compatible plotters are supported
Plots can be generated at a constant 1:1 ratio or a drawing can be automatically scaled to any degree
Plotting extents can be defined using:
The drawing page
The current extents
A definable plotting window
The database extents
Plotter and printer output can be directed through the serial port, parallel port or to a disk file (ie; i/o redirection)

Printer Output

Laser Printers, draft and final mode
Epson and compatible printers are supported, both 8 and 24 pin
All drivers allow:
Quickplots, Final drafts utilizing the printers highest graphics mode, multi sheet prints, scaled and constant ratio prints

Data Transfer

DynaCADD reads and writes both DXF Entity and DXF Drawing Files
DEF File Format is included to insure upward compatibility to all versions on all micro computers
DynaCADD allows compatibility with the new Desktop Publishing solution, Calamus

System Requirements:

Atari 1040ST or Mega series, minimum 1 MByte of memory, 1 MByte disk drive and a Monochrome Monitor

Commitment to the Future

A modular design structure provides the basis for continued upward growth of the DynaCADD solution. A strong commitment to research and development and a world wide support mechanism combined with a young experienced dynamic programming team, are your insurance that the DynaCADD package will remain on the leading edge of CADD technology. Upgrade modules currently under development include surface design and image design, both with the same easy to use graphic interface of the DynaCADD base module.

To order and for more information please contact:
ISD Marketing, Inc. 2651 John Street, Unit #3, Markham, Ontario, Canada L3R 2W5
Tel: (416) 479-1880 Fax: (416) 479-1882

You'll discover that you're able to buy most anything that you could find at your neighborhood shopping center, *and* it's all accessible from your keyboard.

DATABASE DELPHI

by Andy Eddy

Merry Christmas to all of you and thanks for stopping by! If you're at all like me, you've got about two weeks left until Christmas, still have most—if not all—of your shopping left to do and, most of all, *despise* fighting crowds in the stores. As much fun as this cheery holiday is, it's a ritual that takes its toll on my mental and physical well-being.

This month, fittingly enough, we'll take a glimpse into DELPHI's solution to some of this tedium. From the main menu you can enter an area called the "Merchants Row," an electronic shopping mall of sorts. You'll discover that you're able to buy most anything that you could find at your neighborhood shopping center, *and* it's all accessible from your keyboard.

Let's start at the Main menu. From here, all you need to do is type "MER" (the abbreviation for "MERCHANTS ROW"), and this is what you'll see: [Figure 1]

You can see how much power you have at your fingertips. The first step you should take, if you plan on buying anything, is to use the "Set Billing Address" selection which will inform DELPHI and the merchants where your order should be shipped. Simply typing "SET" from the Merchants' menu, then entering "ADD" will get you to the proper area for input of your name, address and phone number.

Once the vital information is in place, you can enter any of the areas—or "stores," if you will—and cruise around for anything that strikes your fancy. Again, DELPHI's menus are not hard to understand, since they don't require you to memorize numbers or cryptic letter combinations to get where you want to go.

At the time of this writing, this is a list of the vendors available from the Merchants' Row and what they have in their online stores:

FIGURE 1

```
Merchants' Row Menu:

Boston Computer Exchange      Long Distance Roses
Classified Ads                MaryMac Industries
Comp-u-store OnLine           Xpress Music
Computer Express              Vendor & Product Support Services
DELTA Gold Voyager Modem      Set Billing Address
Fantasy Plaza                 Help
Florida Fruit Shippers        Exit
Investment Software

MERCHANTS>(Please select a service)
```

Boston Computer Exchange

This is a database that links used computer sellers with needy buyers. You can search through the listings for specific equipment or create your own listing to get rid of computer gear you want to sell off. Listings are categorized, such as Micro, Mini, Laptop, as well as accessories like Printers, Word Processors and Monitors.

Long Distance Roses

As the name implies, this is an online florist. They'll ship roses, orchids and selected FTD bouquets nationwide via overnight delivery.

MaryMac Industries

This is a computer dealership that focuses on Tandy equipment but also has other general computer accessories available as well.

Comp-u-store OnLine

These people claim to have over 250,000 products available, which range from computers to cars, stereos to sofas, and more. There is an additional charge for maintaining membership in the club, which must be taken care of before you can purchase items through them.

Express Music

Have a record album, cassette or CD that you're interested in buying? Do it simply with Express Music, that claims over 25,000 titles are available online. You can scan the database from start to finish, or select specific artists or titles of interest.

Computer Express

They carry products specific to a variety of computer brands and software, in addition to general computer equipment.

DELTA Gold Voyager Modem

A pocket-sized, 300/1200 baud modem with Hayes compatibility.

Fantasy Plaza

A center for interesting software items.

Florida Fruit Shippers

As the name implies, you can order various types of fruit—limes, pineapple, coconuts, mangos and more—as well as fresh seafood, rum cake and other exotic items.

FIGURE 2

```
MERCHANTS>(Please select a service) bos

Welcome to the BoCoEx Database, the oldest and largest of its
kind.

You can search for bargains; you can post the equipment you want
to sell.  We are here to match buyer and seller at fair market
prices. We charge the seller a 10% commission or $25 minimum at
the time of the sale.  Further sales between partners we
introduce are subject to the same commission.  The seller is
responsible for delivering working equipment, or accepting back
or repairing damaged goods.

We've made thousands of trades all over the world. Let us help
you sell equipment or find what you need. These listing data are
provided in good faith and deemed reliable, but are
offered subject to errors, omissions, changes in price or
withdrawal without notice. Incomplete listings will be refused,
and we reserve the right to reject any listing.

We are here to help you find what you need. If you find a
bargain, send EMail to BoCoEx or call a broker at 617-542-4414.

Happy hunting in a great database!
Copyright 1988 Boston Computer Exchange Corporation

Boston Computer Exchange Menu:

Search
Enter a New Listing
Comments to List Manager
Help
Exit

BOCOEX>(Search, Enter, Comments, Help or Exit): sea

Please answer the following questions.  They allow a search of
the BoCoEx Database for the items you want.  Note: The default
answers appear in the brackets [].
```

continued on next page

FIGURE 2 CONTINUED

```
If you are looking for items that are available to buy, type BUY.
If you are looking for items that other people wish to buy and
that you may be able to sell to them, type SELL.

(Please note that the BUY/SELL question has been recently
updated)

Which database? (BUY or SELL) [BUY] buy

EQUIPMENT Menu:

Accessories     Monitor
Disk Drive      Network
Hard Disk       Plotter
Laptop          Printer
Main Frame      Software
Micro           Terminal
Mini            Word Processor
Modem

>What type of product? [] lap

MAKER Menu:

Altos               NEC
Apple               Novation
Ashton-Tate         Okidata
AST                 Panasonic
AT&T                Princeton
Burroughs           Quadram
Control Data        Racal-Vadic
Corvus              Seagate
Data General        Sony
Digital             Talltree
Eagle               Tandon
Epson               Tandy
Hayes               Tecmar
Hercules            TRS-80
HewlettPackard      Vector
IBM                 Wang
Intel               Xerox
Iomega              Zenith
Lotus               OTHER

>What manufacturer? [press RETURN for all manufacturers] nec
What model? [press RETURN for all Models]

If you want to buy what other people have for sale, what is the
HIGHEST price you would be willing to pay?  If you want to sell,
what is the LOWEST price you would accept?

Price? [press RETURN to not specify price]

Searching...Press Control-C at any time to interrupt search

  1 Item selected.

CODE        : 18,107
LIST DATE   : 07-26-1988
QUANTITY    : 2
EQUIPMENT   : LAPTOP
MAKER       : NEC
MODEL       : MULTISPEED
RAM         : 640 K RAM
DISK DRIVE  : 2 @ 720 K
PRINTER     : Panasonic 1091
MONITOR     : CGA Princeton Mac 12
HOW OLD     : 1 yr
PRICE       : $1,225.00

(Enter Number, Scan, Contact Me, (?" or Exit):
```

Investment Software

IBM-compatible, financial software.

As I mentioned, the menus are all designed for maximum simplicity and comfort. To show how easy it is, Figure 2 is a sample session starting from the Merchants' Row menu and continuing on through the listings of the Boston Computer Exchange.

The Merchants' Row also offers a couple of other services of interest and help to online shoppers. The first is the Classified Ads section. Here you can scan through listings from other DELPHItes, or post your own advertisements, in a variety of categories: Employment, Real Estate, Messages, Automotive Sales, Business Services, Computer Marketplace and Other Items to Buy/Sell.

The other area is Vendor & Product Support Services, which offer assistance to Apple, Macintosh and CAD/CAM professionals. For obvious reasons, we won't go into them here.

More Online Entertainment

DELPHI is working on a few games that promote interactivity between users, though at this writing (August) the final locations aren't set. The first is called *Flip-It*, which is an Othello-type game that can be played against the system or against another human player. It's a thought-provoking contest and a lot of fun.

The other game under construction is called *Scramble*, which is an addicting and challenging game of wordplay. It's held in a separate conference room, so players—as many as you want—can chat with each other between games. When a player types "GO," it starts the action: DELPHI throws a 4x4 grid of letters on the screen, and the players present must enter as many words at they possibly can in 90 seconds. The larger the word, the bigger the score; but smaller words can be built and entered more quickly, though at a lesser point count.

Scramble really tests your vocabulary and speed, but most of all is hard to pull away from. By the time you read this, both Flip-It and Scramble should be in their set locations, and we'll give you an update in the next installment of Database DELPHI.

That ought to be enough for now. Again, have a happy holiday and I'll see you *next year*. Till next month, C U online. ■

(from page 48)

do—nothing else, no matter how bad or how nasty, will be quite as scary as watching far out ahead of you, in the waning light of your torch, for the flicker of purple that means Worms!

Use the "DM Two-Step" here if you can, because the worms are fast enough to pursue you down the longest corridor if you retreat. Hit them with doors, use fireballs if you've learned that spell, throw boots at them, and, finally, if you need to, run back to the stairs and up to level 3 to rest. Thankfully, nothing will pursue you up a staircase. The Worm's bite is poisonous, and you'll need to use a "Cure Poison" potion to cure your champions.

You'll also encounter Rockpile monsters and Screamers on this level, but that isn't the worst of it. There are three places where the each of the monsters can regenerate if you take too long getting through the level. If you retreat too often, you may find yourself fighting your way back through more monsters each time. Worse yet, you may find yourself retreating from Rock monsters and suddenly encounter a group of Screamers behind you. Oh yes, this is a *bad* level!

The one good thing about level 4 is that, near the end of the level, there is a room where Screamers regenerate. If you walk away from the room with the door left open, you'll hear the rattle of the door closing when you're a little way down the hall. If you return to see what's going on, you'll find that there's another group of Screamers in the room. You can repeat this process as often as you like.

This not only gives you an unending supply of food, in the form of Screamer Slices, but it gives you a golden opportunity to get some fairly safe fighting experience for your weaker characters and to advance to higher levels for your strong ones. Be sure to close the door behind you when you go into the room, though, so that you don't suddenly find a stray group of Purple Worms chewing on your backside! If you take much damage while fighting the Screamers, you can always just sleep in the room, with the door closed, to regain your health.

Further down level 4, you will find the Prisoner. Unfortunately, his fate is sealed when you realize that the floor plate he is standing on is what is holding shut the iron grating which blocks the stairs to the next level. Be well rested and healthy when you get to this point though, because the worms aren't through with you yet!

Level 5 presents you with another set of rooms, with even more difficult puzzles to solve, to get into the Treasure Stores. These puzzles will require a lot of patience to solve, as most of them involve finding a pattern of buttons or movement. In addition, you will meet two nasty new creatures. The Flying Rattlesnake is fast and poisonous. Luckily, a fireball will take care of him—if you're fast enough.

The other new creature looks for all the world like a deadly Gumby. It's poisonous, too, and has the nasty habit of throwing fireballs back at you.

One nice thing about level 5 is that there is a stairway up to a secret door on level 4 near the Screamer Regeneration room, and there is a hidden doorway near the end of level 5 that connects back to the beginning of the level. This way you can make a quick trip back when you need food. An even better plan is to throw a good supply of Screamer Slices down the stairs, out in front of the party as you go. This way your food supply is always close at hand, and your characters will get a lot more Ninja level practice.

Some of the items you find in the Treasure Stores will be needed on level 6 in the Riddle Room. Here you will find four riddles that you have to answer by placing the correct object in a niche in the wall next to each riddle. You only need to correctly answer three of the riddles to open the door into the rest of the level, but you get a bonus if you answer all four. The answers are fairly logical, and shouldn't be too much of a problem if you think of them in terms of the items you've found. Don't be afraid to use something which appears to be important to the party—as you can take it back once the door is open.

This level presents some of the most difficult puzzles in the dungeon. One of the toughest is in a room near the Riddle Room. This room has a door that closes when you approach it, a transporter mist controlled by a lever, a pit and several floor plates. To solve this puzzle, watch carefully to see where you are transported; then watch what happens when something else is transported.

In the King Midas room, find a way to give the King some gold. The Combination room requires patience. Later, when you find a tiny floorplate that you can't keep depressed, try to get someone, or something else, to stand on it for you, then find a way to freeze them in place temporarily. You'll then find yourself locked between closed doors, with a blinking transporter facing you. This is another place where fast movement is necessary. Practice the timing of the transporter and use magic to open the door behind it. Lighten the load of the Champions, and you'll get it.

Incidentally, just a little before this area is where you'll want to look for that hidden Vorpal Blade.

When it's time to "Test Your Strength," look around the corridors a little first. Then when you've tried to get yourself, or something else down the "Test" corridor, look around again. You may find that you've passed the test without knowing it!

This level introduces you to bands of Skeletons, which can be fairly easily killed, and the golden Eyeballs, or "Beholders." Watch the pupils of the eyeballs; when they turn red, get ready to duck! There are Giant Wasps on this level too. You'll already have made their acquaintance, back on the Purple Worm level. Use a low level fireball on them, but be quick about it, because their sting is poisonous! There's one more place where you need to move quickly, but by now you should be able to handle it easily.

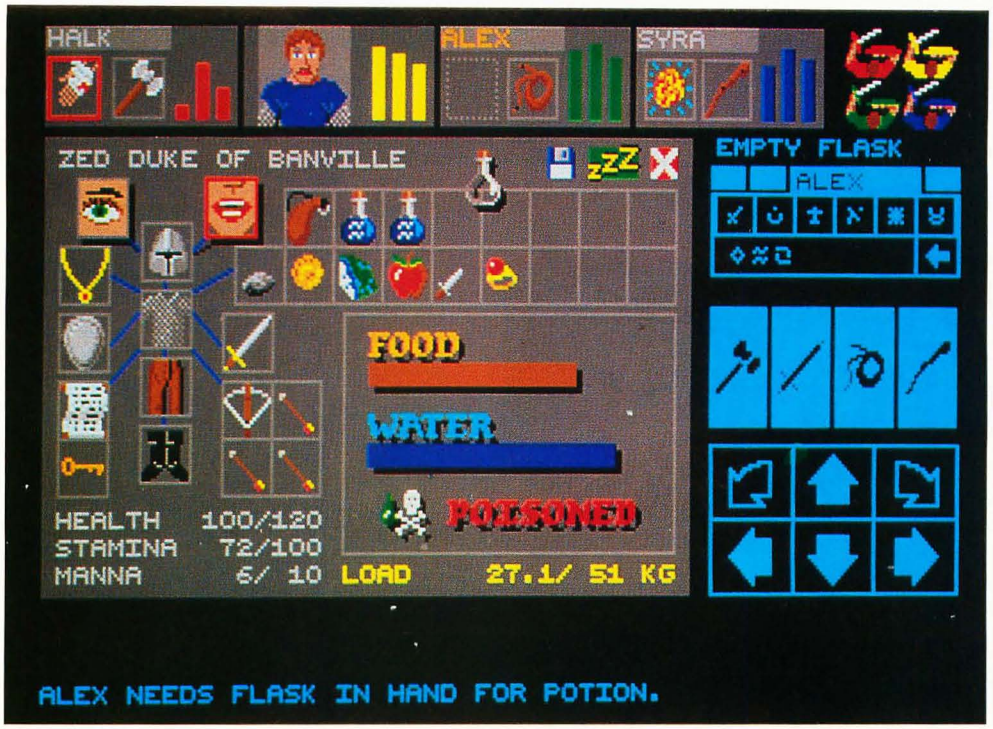
Level 7 is a important level, but for now you won't be able to get into much of it. The keys you need are buried deeper in the dungeon.

Level 8 is one of the most devious traps in the entire dungeon. Unless you can find a way to shut off the incessant rain of fireballs, you may not survive for long. Luckily there's an Altar of VI back up on level 6! This level is important because it's at the top of the Express Stairway which runs down the "back way" to all lower levels. You need to find the Skeleton Keys to open the doors to this stairway though, and the doors can only be opened from within the individual levels. Watch the walls for the places to use the Skeleton Keys; they'll be obvious.

This huge open area is populated not only by Mummies and Skeletons, but by the Green Ghosts or Banshees that you first met on the Purple Worm level. Since they are non-material, you'll need to use either a magical spell to weaken them, or the Vorpal Blade to kill them.

Also, for the first time, you'll meet the annoying giggling Thief. This character won't hurt you, but will steal things right out of your ready hands and run away with them. He can be killed easily, but will lead you on a merry chase, usually right into the clutches of something a lot more deadly! Try not to carry anything in the champions' left hands while on this level. If he steals your torch, it'll get dark all of a sudden, and you

THE DARK LORD CANNOT BE DESTROYED. YOUR MISSION IS TO CAPTURE HIM WITH THE GOOD SIDE OF YOUR MASTER, RESTORING BALANCE TO THE WORLD. THUS YOU HAVE TO USE THE POWERS OF THE FIRESTAFF TO CAGE HIM AND THEN FUSE HIM.



won't be able to see to catch him. There are quite a few hidden doors all around the outside edge of this room, so be very observant.

Most of the lower levels of the dungeon simply introduce more deadly monsters and more intricate and clever puzzles. By the time you've gotten this far, there shouldn't be much you can't handle, given plenty of resting and hiding.

Level 9 has a couple of places where you may need a hint or two though. When you read a riddle engraved in the wall, think about what the wall is made of, and then imagine what you could do if that wall wasn't really there. Don't get a headache from beating your head on the wall, but give it a try a few times nearby. You must find the Corbomite to proceed through this level too. When you find it, examine it closely so you don't overload whoever has to carry it. Finally, by now your party should be strong enough to survive a jump into a pit. You'll need to do that to get past a closed door on this level, but you'll have to do some climbing first.

There are three notable locations on level 10. The first is the puzzle that faces you right at the beginning. The area at the foot of the stairs is trapped with transporters, and there is only one path through the blocks which will get you to the other side. The message on the wall "gives" you a clue as to the "shape" of the path you must follow to get through.

Once you get through, you are faced with two doors and only one hard-to-see key. Choose either one; they both end up at the same place. The only difference is the monsters you'll encounter behind them. They'll bring you out at the Zoom room, which is one of the most fun places in the dungeon. The key to getting off the "merry-go-round" is timing. Count how many steps there are in a full circle, then jump off when the door comes around. If you want to try for the niche with the objects in it, be warned that if you miss and fall into the transporter mist, you'll be sent all the way back to the beginning of the level.

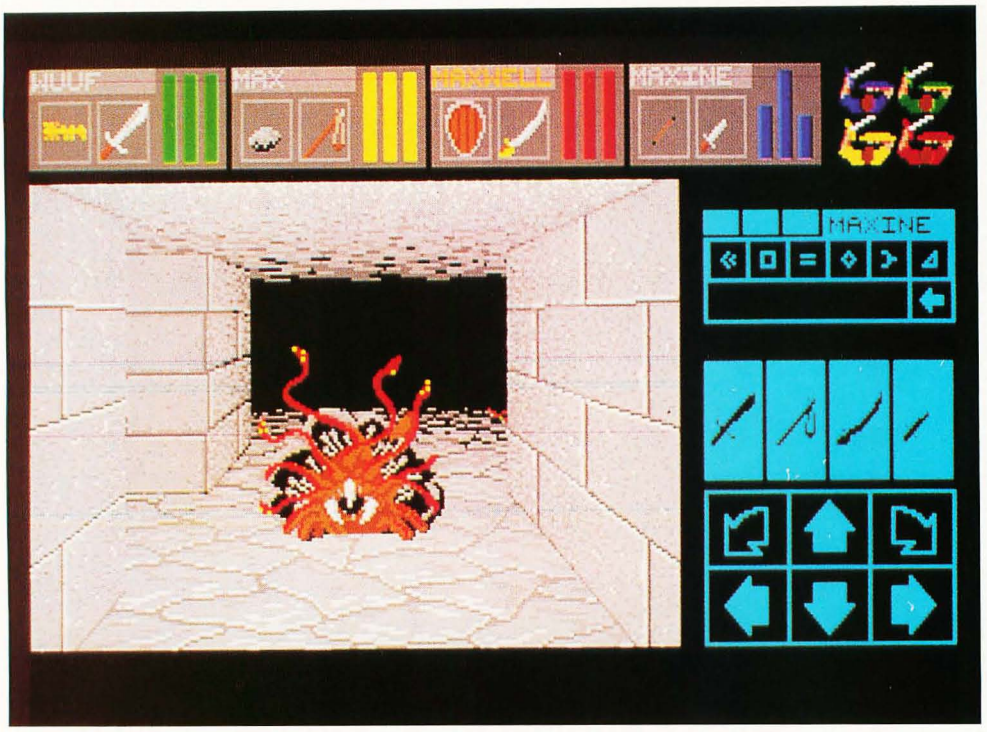
This level will present you with the nastiest monsters since the Purple Worms, in the form of Scorpions. One big problem is that they regenerate almost as fast as you can kill them, and their sting is very definitely poisonous. This is a good area to run like heck past them! Unfortunately, the Skeleton Key door to the Express Stairway is back in their domain, so you'll have to return to the area. But when you do, you should notice a difference. You'll need to examine the walls in this area anyway, because there are some valuable items hidden on this level.

The Clockwise room on level 11 is one of the strangest puzzles in the dungeon. You'll need to go forward and backwards, both clockwise and counterclockwise around the room before you'll get out. Watch the floor for keys, and leave the items you'll find in the corners to mark your progress.

Once you get out, you'll find the Diamond Edge Sword down a long passageway with ominous looking holes in the walls. Be prepared to hold your breath a long time once you pick up the sword. A hint for surviving this trap: The trap is triggered by the sword, not by you. If you drop the sword and jump back, things will clear up soon, and you'll be able to repeat the action.

Nearby is also the Fury Sword which throws a limited number of Fireballs. This is nice for times when your magic users are exhausted, but try to save it for later when you'll really need it.

Later on this level you'll again be faced with a "Let's Make a Deal" situation of three doors and only one key. (The key, again, is grey on the grey floor in front of the doors.) All three doors eventually lead to the same place, but one is fairly safe. Through another you meet the "club" and the last is a real "hornet's nest." If you ever need help deciding which way to go, you can place a coin in your action hand and flip it; but in some cases, like this one, the most direct approach is the safest.



YOU CAN ONLY SAVE ONE GAME PER DISK, SO IT MIGHT BE A GOOD IDEA TO DEDICATE SEVERAL DISKS TO THE GAME WHILE YOU'RE PLAYING. YOU *CAN* RENAME AND COPY SEVERAL SAVED GAMES ONTO ONE DISK, BUT THAT CAN BE A LOT OF HASSLE.

Hopefully, you've thought to bring along all the coins you've found, because near the end of this level is a room full of "slot machines" which will reveal all kinds of goodies. Also hidden behind an illusionary wall near the end is a pair of Boots of Speed. They're hard to find, but very valuable! The last puzzle on this level is one worthy of Sherlock Holmes. You'll need an item that he would carry, from the previous level. The item revealed by this puzzle is absolutely essential to the game!

The most notable feature of level 12 is your discovery of the Knights. These animated suits of armor are one of the most fearsome looking of the creatures in the dungeon, as they each wield two swords. They can cause a lot of damage and are hard to kill. If there was ever a place to use a door as a weapon, and those Magical Boxes you've (hopefully) been saving, this is it! When you do manage to kill the Knights, be sure to sift through the wreckage, as you never know when something may be carrying an item you might need.

By now you should have a rope, which you'll need unless you discover how to turn off the invisible pits you'll unexpectedly find. Some things in the dungeon are "toggled" on and off by the same control.

When you fall into the pits, you'll have to fight a Fire Elemental to get back up. Again, this is a non-material being. More semi-non-material beings on this level are the Materializers, which slowly appear from just a faint blue globe of light, to a fuzzy outline, and finally to their disgusting solid form, and then back to almost nothing again. They can be hit by normal weapons only when they are fully materialized, so just go after them with your Vorpal Blades. Be wary of them though, because they use both Fireballs and Poison Clouds against you.

If you get through all of that, you'll still need to wade your way through a horde of Giant Spiders. They don't need any special attacks, but they're very tough, and there are *hundreds* of them. (Well, maybe not that many, but it *feels* like there are!)

Once you're past the Spider parade, you'll see a closed door at the end of a corridor. This is the Spider Regeneration room, where death awaits. There are a few useful items inside, but unless you can freeze the Spiders in their tracks, the items may not be worth dying for. Be careful because that room is a trap. If you get too close, you may end up having to fight the Spiders anyway.

The last hurdle on this level is another test of your quickness, beyond which there is one more key that is essential to the game.

Level 13 is where Lord Chaos lives. You won't be ready to fight him yet, so it's a good idea to just skip this level for now. He's not alone on this level either!

The last level in the dungeon is the lair of the Dragon. He is incredibly tough and incredibly dangerous. Piles of ashes, which may be all that remains of other foolhardy adventurers, are strewn around on the floor. It's a good idea to investigate these ashes; you might find something important.

The Dragon can be fought with the same move, turn, hit, move, turn, hit technique as the Purple Worms. Just be careful not to slip up and end up facing him! One blast of his fiery breath may be enough to kill most of your party. Once you kill the Dragon, you'll be set for food for the rest of the game.

You should now have all the keys necessary to go back up to level 7 and investigate the Wizard's Workshop. Use the "backstairs" to go directly up to level 8; then run quickly back to the stairs up to level 7. With the RA keys, open the shimmering force-field doors; then with the Ruby Key, enter the workshop.

Inside you'll find all kinds of useful things and many "red herrings." Be sure to read all the scrolls you'll find, as they give hints about what to do when you meet Lord Chaos. Search this

area meticulously because you still need more keys.

There are four Turquoise Doors and only one Turquoise Key, so again, it's pick-a-door time. A hint about these doors: To see what's behind all of them, you can save the game before opening any of them, then restart the game from that point. Some of the items are more useful than others. With the last RA key, find your way to the Tomb of the Firestaff and unlock its door with the Master key.

Oh yes—those giant Stone Golems. They'll try to keep you from taking the Firestaff, but they are slow so you should be able to outrun them. But they are hard to kill! Again, bashing with doors and freezing with magic is a good way to approach them.

Once you have the Firestaff, you need to find the Power Gem to energize it. If you were lucky on the Dragon level, you may have already found it. If not, go back to the Wizard's Workshop and locate the secret exit directly down to the bottom levels of the dungeon. If you search carefully near that exit from the workshop, you'll find an interesting key which will let you out at the bottom of the Express Stairs.

Once you find the Power Gem, there will be a scroll nearby which will tell you what to do with it. Before you do anything, though, be sure that you have all the water and food, and whatever else you think you may need, to finish up the game. Once you free the Gem, all exits back to the upper levels are sealed. When you've freed the Gem and assembled it with the Firestaff, you'll be ready to go back up to level 13 and face Lord Chaos.

The Dark Lord cannot be destroyed. Your mission is to capture him and recombine him with the good side of your master, restoring balance to the world. Thus you have to use the powers of the Firestaff to cage him and then fuse him. This is easier said than done, because he has an annoying habit of transporting out of your cage just before you can get him. If you watch how he moves though, you may notice that he always walks in the direction he's facing, even if there's a completed Fluxcage there.

Of course, Lord Chaos is not alone on this level. While you're dancing with him, you'll also have to be dealing with the Demons which are wandering around and the Black Fire elementals that block your path. Luckily the Firestaff has an option which will help you deal with these too.

This last level may take you some time to get through, and the function of all the coins laying around on the floor in the Dragon's room will become apparent once you notice that there's an Altar of VI behind the door you have to pay to open. Hopefully, you brought *lots* of coins down from other levels too.

The last hint about playing *Dungeon Master* is perhaps the most important: *Save the game often*!! Whenever one of your Champions makes a new experience level, *save the game*! When you reach the stairs down to the next dungeon Level, *save the game*! When you run into a new type of monster that looks nasty, *save the game*! (You get the idea?)

You can only save one game per disk, so it might be a good idea to dedicate several disks to the game while you're playing. You *can* rename and copy several saved games onto one disk, but that can be a lot of hassle. By having several disks to save to, you can always go back to an earlier saved game and try things differently without having to exit to the Desktop and rename anything. If something nasty lurking just around a corner should take you and your party by surprise and wipe them all out, you'll be able to start over from a point fairly close to there if you *saved the game*!

Hopefully this tour of the dungeon and the hints presented here will make your stay in the world of Dungeon Master more enjoyable—not to mention a little longer lasting! ■

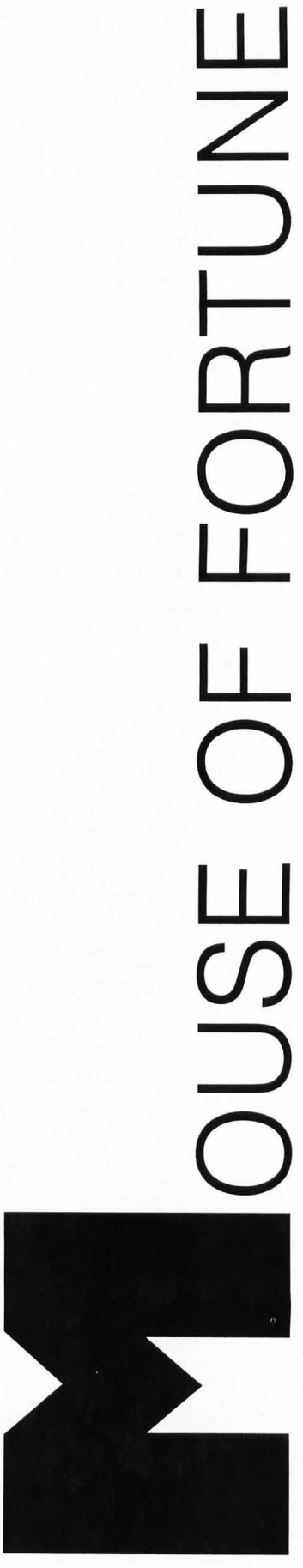
■ *Bob Retelle has been a professional in the field of telecommunications for the past several years (look for BOBR online) and has been writing (and playing) computer adventure games for longer than he'd like to admit.*

MOUSE OF FORTUNE
LISTING 1
GFA BASIC

```
' ************************
' **  Mouse Of Fortune  **
' **        by          **
' **    Kirk Stover     **
' ************************
'
On Break Gosub Dne
On Error Gosub Dne_error
Max_puzzles%=99
Dim Oldpal%(15),Score%(2),Game%(2),Amount$(20),Ltr_status!(25)
Dim Word!(Max_puzzles%),Title$(Max_puzzles%),Word$(Max_puzzles%)
Gosub Initialize
Gosub Play
Gosub Terminate
End
'
Procedure Terminate
  Cls
  Gosub Fix_color(0)
Return
'
Procedure Initialize
  Local T%
  T%=Random(Timer)
  Gosub Check_color
  Gosub Fix_color(1)
  Gosub Set_color
  Gosub Read_amounts
  Gosub Make_sprite
  Gosub Set_word_status
  Gosub Load_data
Return
'
Procedure Check_color
  Local Res%
  Res%=Xbios(4)
  If Res%<>0
    Alert 3,"Low Resolution| |    Only!",1,"OK",Button%
    End
  Endif
Return
'
Procedure Fix_color(Flag%)
  Local X%
  For X%=0 To 15
    If Flag%=1
      Oldpal%(X%)=Xbios(7,W:X%,W:-1)
    Else
      Void Xbios(7,W:X%,W:Oldpal%(X%))
    Endif
  Next X%
Return
'
Procedure Set_color
  Black%=0
  Setcolor 0,0,0,0
  Yellow%=1
  Setcolor 15,7,7,0
  White%=2
  Setcolor 1,7,7,7
  Red%=3
  Setcolor 2,6,0,0
  Light_gray%=4
  Setcolor 4,5,5,5
  Light_red%=5
  Setcolor 6,7,0,0
  Blue%=6
  Setcolor 3,0,0,4
  Light_blue%=7
  Setcolor 5,0,0,7
  Light_green%=8
  Setcolor 7,0,7,0
  Gray%=15
  Setcolor 13,0,4,4
Return
'
Procedure Read_amounts
  Local X%
  For X%=0 To 20
    Read Amount$(X%)
  Next X%
Return
'
Procedure Set_word_status
  Local X%
```

```
  For X%=0 To Max_puzzles%
    Word!(X%)=True
  Next X%
Return
'
Procedure Load_data
  Puzzles_loaded!=False
  Repeat
    Deffill Red%,2,15
    Prbox 0,0,319,199
    Graphmode 2
    Deftext Blue%,4,0,12
    Text 32,20,"MOUSE OF FORTUNE"
    Deftext White%,4,0,12
    Text 28,20,"MOUSE OF FORTUNE"
    Deftext White%,0,0,4
    Text 115,40,"By Kirk Stover"
    Graphmode 1
    Alert 0,"                                  |    Use puzzles from:          | |",3,"File|Create|Program",Button%
    On Button% Gosub Load_puzzles,Create_puzzles,Read_puzzles
  Until Puzzles_loaded!
Return
'
Procedure Read_puzzles
  Number_words%=0
  Do
    Read Title$
    Title$(Number_words%)=Title$
    Read Word$(Number_words%)
    Word!(Number_words%)=False
    Inc Number_words%
    Exit If Number_words%>Max_puzzles%
  Loop
  If Number_words%>0
    Puzzles_loaded!=True
  Endif
Return
'
Procedure Load_puzzles
  Local File_error!
  File_error!=False
  Number_words%=0
  Fileselect "\*.PUZ","",Filname$
  If Filname$<>""
    If Exist(Filname$)
      Open "I",#1,Filname$
      Do
        Input #1,Title$
        Exit If Eof(#1)
        If Len(Title$)>10
          File_error!=True
        Else
          Title$(Number_words%)=Upper$(Title$)
        Endif
        Input #1,Word$
        Exit If Eof(#1)
        If Len(Word$)>26
          File_error!=True
        Else
          Word$(Number_words%)=Upper$(Word$)
        Endif
        Word!(Number_words%)=False
        Exit If File_error!
        Inc Number_words%
        Exit If Number_words%>Max_puzzles%
      Loop
      Close #1
      If File_error!
        Alert 3,"Invalid data|loaded for|the puzzles.",1,"OK",Button%
      Else
        If Number_words%>0
          Puzzles_loaded!=True
        Endif
      Endif
    Endif
  Endif
Return
'
Procedure Create_puzzles
  Number_words%=0
  Fileselect "\*.PUZ","",Filname$
  If Filname$<>""
    If Not (Exist(Filname$))
      If Right$(Filname$)<>"\"
        Open "O",#1,Filname$
```

```
        Cls
        Print "Enter the puzzle (alpha and space only)"
        Print "Enter 'END' to quit"
        Print
        Print "Example:"
        Print
        Print "Puzzle topic: TITLE"
        Print "Puzzle data:  GONE WITH THE WIND"
        Print
        Do
          Repeat
            Print "Puzzle topic: ";
            Form Input 10,Title$
            Title$=Upper$(Title$)
            Exit If Title$="END"
            Gosub Check_title
          Until Title_ok!
          Exit If Title$="END"
          Repeat
            Print "Puzzle data:  ";
            Form Input 26,Word$
            Word$=Upper$(Word$)
            Exit If Word$="END"
            Gosub Check_word
          Until Word_ok!
          Exit If Word$="END"
          Print "Is this correct? "
          Print
          Repeat
            X$=Inkey$
          Until X$<>""
          If Upper$(X$)<>"N"
            Write #1,Title$,Word$
            Inc Number_words%
          Endif
          Exit If Number_words%>Max_puzzles%
        Loop
        Write #1,Chr$(13)
        Close #1
      Endif
    Endif
  Endif
Return
'
Procedure Check_title
  Local X%,X$
  Title_ok!=True
  For X%=1 To Len(Title$)
    X$=Mid$(Title$,X%,1)
    If X$<"A" Or X$>"Z"
      If X$<>" "
        Title_ok!=False
      Endif
    Endif
  Next X%
  If Not (Title_ok!)
    Print "INVALID TOPIC, TRY AGAIN!"
  Endif
Return
'
Procedure Check_word
  Local X%,X$
  Word_ok!=True
  For X%=1 To Len(Word$)
    X$=Mid$(Word$,X%,1)
    If X$<"A" Or X$>"Z"
      If X$<>" "
        Word_ok!=False
      Endif
    Endif
  Next X%
  If Not (Word_ok!)
    Print "INVALID DATA, TRY AGAIN!"
  Endif
Return
'
X=Asc(X$)
Procedure Make_sprite
  Local X%
  Let Mouse$=Mki$(0)+Mki$(0)
  Let Mouse$=Mouse$+Mki$(0)
  Let Mouse$=Mouse$+Mki$(0)
  Let Mouse$=Mouse$+Mki$(5)
  For X%=1 To 16
    Read Foregrnd,Backgrnd
```

```
    Let Mouse$=Mouse$+Mki$(Backgrnd)+Mki$(Foregrnd)
  Next X%
Return
'
' **** Game Play Loop
'
Procedure Play
  Local Answer$
  Do
    Gosub New_game
    Do
      Gosub New_round
      If More_puzzles!
        Gosub Sound_begin_puzzle
        Do
          Put 0,56,Screen_save$
          Gosub Player_option
          On Choice% Gosub Choose_letter,Buy_vowel,Solve_puzzle,Loseturn
          Exit If Solved!
        Loop
        Put 0,56,Screen_save$
        Print At(6,12);"Player ";Player%;" wins the round!"
        Score%(3-Player%)=0
        Gosub Print_scores
        Gosub Sound_winner
        Add Game%(Player%),Score%(Player%)
        Score%(Player%)=0
        Pause 60
        Gosub Print_scores
        Pause 40
      Endif
      Exit If Not (More_puzzles!)
      Exit If Round%=3
    Loop
    Exit If (Not (More_puzzles!)) And Round%=1
    Put 0,56,Screen_save$
    If Game%(1)>Game%(2)
      Print At(6,12);"Player 1 wins the game!"
    Else
      If Game%(1)<Game%(2)
        Print At(6,12);"Player 2 wins the game!"
      Else
        Print At(6,12);"Tie game!"
      Endif
    Endif
    Gosub Sound_winner
    Pause 90
    Exit If Not (More_puzzles!)
    Put 0,56,Screen_save$
    Print At(3,12);"(P)lay new game or (Q)uit?";
    Repeat
      Answer$=Inkey$
      Answer$=Upper$(Answer$)
    Until Answer$="P" Or Answer$="Q"
    Exit If Answer$="Q"
  Loop
Return
'
Procedure New_game
  Game%(1)=0
  Game%(2)=0
  Round%=0
Return
'
Procedure New_round
  Local X%
  Solved!=False
  Score%(1)=0
  Score%(2)=0
  Inc Round%
  Gosub Set_ltr_status
  More_puzzles!=False
  For X%=0 To Number_words%-1
    If Word!(X%)=False
      More_puzzles!=True
    Endif
  Next X%
  If More_puzzles!
    Repeat
      X%=Random(Number_words%)
    Until Word!(X%)=False
    Word!(X%)=True
    Word$=Word$(X%)
    Title$=Title$(X%)
    Gosub Draw_screen
```

```
    Player%=2
    Gosub Next_player
  Else
    Alert 3,"           |There are no|more puzzles|left.",1,"QUIT",Button%
  Endif
Return
'
Procedure Set_ltr_status
  Local X%
  For X%=0 To 25
    Ltr_status!(X%)=True
  Next X%
Return
'
Procedure Draw_screen
  Local B%
  Cls
  Defline 1,2
  Color Light_blue%
  Rbox 0,0,319,80
  Defline 1,2
  Line 0,20,319,20
  Deftext Red%,4,0,12
  Text 30,16,"MOUSE OF FORTUNE"
  For B%=0 To 25
    If B%>(Len(Word$)-1)
      Deffill Gray%,2,3
    Else
      If Mid$(Word$,B%+1,1)=" "
        Deffill Gray%,2,3
      Else
        Deffill Blue%,2,4
      Endif
    Endif
    Pbox B%*12+5,40,B%*12+14,55
  Next B%
  Deftext White%,8,0,6
  Text (160-(Len(Title$))*4),35,Title$
  Deftext White%,0,0,4
  Text 20,70,"Letter value $0000"
  Get 0,56,319,100,Screen_save$
  Deftext Light_red%,0,0,8
  Text 30,150,"Player 1"
  Deftext Light_gray%,0,0,8
  Text 200,150,"Player 2"
  Deftext White%,0,0,6
  Text 130,170,"ROUND "+Str$(Round%)
  Text 140,190,"GAME"
  Gosub Print_scores
Return
'
Procedure Print_scores
  Local Score$
  Deftext Light_red%,0,0,6
  Score$="$"+Right$("000000"+Str$(Score%(1)),6)
  Text 40,170,Score$
  Score$="$"+Right$("000000"+Str$(Game%(1)),6)
  Text 40,190,Score$
  Deftext Light_gray%,0,0,6
  Score$="$"+Right$("000000"+Str$(Score%(2)),6)
  Text 210,170,Score$
  Score$="$"+Right$("000000"+Str$(Game%(2)),6)
  Text 210,190,Score$
Return
'
Procedure Pause
  Local X$,Pause_save$
  X$=Inkey$
  If X$<>""
    Get 0,100,319,116,Pause_save$
    Print At(12,14);"GAME IN PAUSE MODE";
    Repeat
      X$=Inkey$
      If X$=Chr$(27)
        Gosub Terminate
        End
      Endif
    Until X$<>""
    Put 0,100,Pause_save$
    T%=Timer
  Endif
Return
'
Procedure Player_option
  Local B%,T1%,X%,Y%
```

MOUSE OF FORTUNE

```
    Color Light_blue%
    Defline 1,1
    Box 4,80,110,96
    Box 110,80,210,96
    Box 210,80,316,96
    Deffill White%,2,8
    Pbox 6,82,108,94
    Pbox 112,82,208,94
    Pbox 212,82,314,94
    Graphmode 2
    Deftext Black%,0,0,4
    Text 8,90,"  CHOOSE LETTER     BUY A VOWEL     SOLVE PUZZLE   "
    Gosub Letters_left
    Gosub Vowels_left
    If Not (More_letters!)
      Deffill Light_blue%,3,1
      Fill 7,82
      Fill 112,82
      Gosub Sound_end_of_letters
    Else
      If (Not (More_vowels!)) Or (Score%(Player%)<250)
        Deffill Light_blue%,3,1
        Fill 112,82
      Endif
    Endif
    Graphmode 1
    Choice%=0
    T%=Timer
    T1%=Timer
    Do
      If (Timer-T1%)>200
        Gosub Sound_clock
        T1%=Timer
      Endif
      Gosub Pause
      Mouse X%,Y%,B%
      If B%=1
        If Y%>80 And Y%<96
          If X%>8 And X%<110
            If More_letters!
              Choice%=1
            Else
              Gosub Sound_invalid_choice
            Endif
          Endif
          If X%>110 And X%<210
            If (Not (More_letters!)) Or (Not (More_vowels!)) Or (Score%(Player%)<250)
              Gosub Sound_invalid_choice
            Else
              Choice%=2
            Endif
          Endif
          If X%>210 And X%<313
            Choice%=3
            Gosub Sound_valid_choice
          Endif
        Else
          Gosub Sound_invalid_choice
        Endif
      Endif
      If (Timer-T%)>2000
        Choice%=4
      Endif
      Exit If Choice%>0
    Loop
  Return
  '
  Procedure Buy_vowel
    Local X%,Y%,B%,T1%,L%,N%,Dne!
    Put 0,56,Screen_save$
    Sub Score%(Player%),250
    Gosub Print_scores
    Deftext Light_green%,0,0,4
    Deffill Light_green%,2,8
    Color Light_blue%
    Defline 1,2
    For X%=0 To 25
      Box X%*12+4,80,X%*12+16,96
      If (Not (Ltr_status!(X%))) Or (X%<>0 And X%<>4 And X%<>8 And X%<>14 And X%<>20)
        Pbox X%*12+5,81,X%*12+15,95
      Else
        Text X%*12+8,90,Chr$(X%+65)
      Endif
    Next X%
    Deftext Yellow%,16,0,16
```

```
    Dne!=False
    L%=99
    T%=Timer
    T1%=Timer
    Do
      If (Timer-T1%)>200)
        Gosub Sound_clock
        T1%=Timer
      Endif
      Gosub Pause
      Mouse X%,Y%,B%
      If Y%>80 And Y%<96
        '       N%=Int((X%-4)\12)
        N%=(X%-4)/12
        If N%)25
          N%=25
        Endif
        If (Ltr_status!(N%)) And (N%=0 Or N%=4 Or N%=8 Or N%=14 Or N%=20)
          A$=Chr$(N%+65)
          If L%<>N%
            Text 155,74,A$
            L%=N%
          Endif
          B%=Mousek
          If B%=1
            Dne!=True
          Endif
        Endif
      Endif
      Exit If Dne!=True
      Exit If (Timer-T%)2000)
    Loop
    If Dne!
      Ltr_status!(L%)=False
      Gosub Find_vowel
    Else
      Gosub Loseturn
    Endif
  Return
  '
  Procedure Choose_letter
    Local X%,Y%,B%,T1%,L%,N%,Dne!
    Put 0,56,Screen_save$
    Gosub Dollar_amt
    If (Not (Bankrupt!)) And (Not (Loseturn!))
      Deftext Light_green%,0,0,4
      Deffill Light_green%,2,8
      Color Light_blue%
      Defline 1,2
      For X%=0 To 25
        Box X%*12+4,80,X%*12+16,96
        If (Not (Ltr_status!(X%))) Or (X%=0 Or X%=4 Or X%=8 Or X%=14 Or X%=20)
          Pbox X%*12+5,81,X%*12+15,95
        Else
          Text X%*12+8,90,Chr$(X%+65)
        Endif
      Next X%
      Deftext Yellow%,16,0,16
      Dne!=False
      L%=99
      T%=Timer
      T1%=Timer
      Do
        If (Timer-T1%)>200)
          Gosub Sound_clock
          T1%=Timer
        Endif
        Gosub Pause
        Mouse X%,Y%,B%
        If Y%>80 And Y%<96
          N%=(X%-4)\12
          If N%)25
            N%=25
          Endif
          If (Ltr_status!(N%)) And (N%<>0 And N%<>4 And N%<>8 And N%<>14 And N%<>20)
            A$=Chr$(N%+65)
            If L%<>N%
              Text 155,74,A$
              L%=N%
            Endif
            B%=Mousek
            If B%=1
              Dne!=True
            Endif
          Endif
```

MOUSE OF FORTUNE

```
      Endif
      Exit If Dne!=True
      Exit If (Timer-T%)>2000)
    Loop
    If Dne!=True
      Ltr_status!(L%)=False
      Gosub Find_letter
    Else
      Gosub Loseturn
    Endif
  Endif
Return
'
Procedure Solve_puzzle
  Local Answer$
  Put 0,56,Screen_save$
  Print At(1,12);Title$;" = ";
  Form Input (Len(Word$)),Answer$
  If Word$=Upper$(Answer$)
    Solved!=True
    Gosub Show_letters
  Else
    Gosub Sound_wrong_letter
    Gosub Next_player
  Endif
Return
'
Procedure Letters_left
  Local X%,L$
  More_letters!=False
  Last%=Len(Word$)
  For X%=1 To Len(Word$)
    L$=Mid$(Word$,X%,1)
    If L$<>" "
      L=Asc(L$)-65
      If Ltr_status!(L)
        If (L$<>"A" And L$<>"E" And L$<>"I" And L$<>"O" And L$<>"U")
          More_letters!=True
        Endif
      Endif
    Endif
  Next X%
Return
'
Procedure Vowels_left
  Local X%
  More_vowels!=False
  For X%=0 To 25
    If Ltr_status!(X%)
      If X%=0 Or X%=4 Or X%=8 Or X%=14 Or X%=20
        More_vowels!=True
      Endif
    Endif
  Next X%
Return
'
Procedure Show_letters
  Local X%,A$
  Deftext Yellow%,0,0,6
  Graphmode 2
  For X%=1 To Len(Word$)
    A$=Mid$(Word$,X%,1)
    If A$<>" "
      Gosub Sound_puzzle_solved
      Text (X%-1)*12+6,51,A$
      Pause 5
    Endif
  Next X%
  Graphmode 1
Return
'
Procedure Find_letter
  Local X%,Found!
  Found!=False
  For X%=1 To Len(Word$)
    If Mid$(Word$,X%,1)=A$
      Found!=True
      Deftext White%,0,0,6
      Graphmode 2
      Text (X%-1)*12+6,51,A$
      Graphmode 1
      Gosub Sound_right_letter
      Pause 30
      Add Score%(Player%),Val(Amount$(Amount%))
      Gosub Print_scores
```

```
    Endif
  Next X%
  If Not (Found!)
    Gosub Sound_wrong_letter
    Gosub Next_player
  Endif
  Pause 30
Return
'
Procedure Find_vowel
  Local X%,Found!
  Found!=False
  For X%=1 To Len(Word$)
    If Mid$(Word$,X%,1)=A$
      Found!=True
      Deftext White%,0,0,6
      Graphmode 2
      Text (X%-1)*12+6,51,A$
      Graphmode 1
      Gosub Sound_right_letter
      Pause 30
    Endif
  Next X%
  If Found!=False
    Gosub Sound_wrong_letter
    Gosub Next_player
    Pause 30
  Endif
Return
'
Procedure Next_player
  Inc Player%
  If Player%>2
    Player%=1
    Sprite Mouse$,60,120
  Else
    Sprite Mouse$,230,120
  Endif
Return
'
Procedure Bankrupt
  Put 0,56,Screen_save$
  Print At(6,12);"Player ";Player%;" is BANKRUPT!"
  Score%(Player%)=0
  Gosub Print_scores
  Gosub Sound_bankrupt
  Pause 40
  Gosub Next_player
Return
'
Procedure Loseturn
  Put 0,56,Screen_save$
  Print At(6,12);"Player ";Player%;" loses turn!"
  Gosub Sound_loseturn
  Pause 40
  Gosub Next_player
Return
'
Procedure Dollar_amt
  Local X%
  Deftext White%,0,0,4
  Bankrupt!=False
  Loseturn!=False
  For X%=1 To Random(20)+30
    Amount%=Random(21)
    Text 104,70,Amount$(Amount%)
    Sound 1,15,(Abs(Amount%-8)+1),4
  Next X%
  Sound 1,0
  If Amount%=4 Or Amount%=19
    Gosub Bankrupt
    Bankrupt!=True
  Endif
  If Amount%=10 Or Amount%=14
    Gosub Loseturn
    Loseturn!=True
  Endif
Return
'
' **** sound routines ****
'
Procedure Sound_begin_puzzle
  Local Nte%
  For Nte%=2 To 6
    Sound 1,15,1,Nte%,5
  Next Nte%
  Sound 1,0
Return
'
Procedure Sound_end_of_letters
  Local Dur%
  For Dur%=1 To 5
    Sound 1,15,3,5,Dur%
    Sound 1,0
    Pause 4
  Next Dur%
Return
'
Procedure Sound_invalid_choice
  Sound 1,15,1,6,5
  Sound 1,0
Return
'
Procedure Sound_valid_choice
  Sound 1,15,1,5,3
  Sound 1,15,1,4,3
  Sound 1,0
Return
'
Procedure Sound_clock
  Sound 1,11,3,5,3
  Sound 1,0
Return
'
Procedure Sound_puzzle_solved
  Print Chr$(7);
Return
'
Procedure Sound_wrong_letter
  Sound 1,15,7,2,15
  Sound 1,0
Return
'
Procedure Sound_right_letter
  Print Chr$(7);
Return
'
Procedure Sound_bankrupt
  Local Oct%,Nte%,Dly%
  For Oct%=7 To 1 Step -1
    For Nte%=12 To 1 Step -1
      Sound 1,15,Nte%,Oct%,0
      For Dly%=1 To 300
      Next Dly%
    Next Nte%
  Next Oct%
  Sound 1,0
Return
'
Procedure Sound_loseturn
  Local X%
  For X%=1 To 3
    Sound 1,15,3,4,5
    Sound 1,15,1,3,5
  Next X%
  Sound 1,0
Return
'
Procedure Sound_winner
  Local Oct%,Nte%
  For Oct%=5 To 6
    For Nte%=1 To 12
      Sound 1,15,Nte%,Oct%,1
      Sound 1,15,13-Nte%,10-Oct%,1
    Next Nte%
  Next Oct%
  Sound 1,0
Return
'
'      Dollar Amounts
'
Data 0250,0100,0150,0200,0000,0400,0500,0600,0750,1000,0000
Data 0100,0150,0200,0000,0400,0500,0600,0750,0000,0250
'
'      Sprite Data For Mouse
'
Data 128,0,448,0,672,0,2032,0,992,0,2032,0,4088,0,8188,0
Data 8188,0,8188,0,8188,0,4088,0,2032,0,25568,0,4544,0,3840,0
'
'      Puzzles
```

```
'
Data "PLACE","LITTLE BIGHORN"
Data "PLACE","MOUNT RUSHMORE"
Data "PLACE","MIAMI FLORIDA"
Data "PLACE","ACAPULCO MEXICO"
Data "PLACE","NEW YORK CITY"
Data "PLACE","YELLOWSTONE NATIONAL PARK"
Data "PLACE","BERMUDA TRIANGLE"
Data "PLACE","SAN FRANCISCO CALIFORNIA"
Data "PLACE","MEXICO CITY"
Data "PLACE","WEST BERLIN"
Data "PLACE","BUFFALO NEW YORK"
Data "PLACE","CHATTANOOGA TENNESSEE"
Data "PLACE","CINCINNATI OHIO"
Data "PLACE","ROCHESTER MINNESOTA"
Data "PERSON","ABRAHAM LINCOLN"
Data "PERSON","ERNEST HEMINGWAY"
Data "PERSON","WILLIAM FAULKNER"
Data "PERSON","RONALD REAGAN"
Data "PERSON","JACKIE ROBINSON"
Data "PERSON","JOHN LENNON"
Data "PERSON","YOKO ONO"
Data "PERSON","BILLY JOEL"
Data "PERSON","BURT REYNOLDS"
Data "PERSON","PRINCE CHARLES"
Data "PERSON","ROBERT REDFORD"
Data "PERSON","BRUCE SPRINGSTEEN"
Data "PERSON","FRANK SINATRA"
Data "PERSON","NELSON ROCKEFELLER"
Data "PERSON","HENRY FORD"
Data "PERSON","JACKIE GLEASON"
Data "PERSON","CHARLES SCHULTZ"
Data "PERSON","ELVIS PRESLEY"
Data "PERSON","OMAR SHARIFF"
Data "PERSON","ORSON WELLES"
Data "PERSON","MARILYN MONROE"
Data "PERSON","SHIRLEY TEMPLE"
Data "PERSON","ALAN ALDA"
Data "PERSON","PAUL NEWMAN"
Data "PERSON","TOM CRUISE"
Data "TITLE","RAIDERS OF THE LOST ARK"
Data "TITLE","FROM HERE TO ETERNITY"
Data "TITLE","BRIDGE ON THE RIVER KWAI"
Data "TITLE","JAILHOUSE ROCK"
Data "TITLE","BEACH BLANKET BINGO"
Data "TITLE","OPERATION OVERLORD"
Data "TITLE","THE WAY WE WERE"
Data "TITLE","CATCHER IN THE RYE"
Data "TITLE","TRUTH OR CONSEQUENCES"
Data "TITLE","UP THE DOWN STAIRCASE"
Data "TITLE","THE HILLS HAVE EYES"
Data "TITLE","THE WIZARD OF OZ"
Data "TITLE","ON THE WATERFRONT"
Data "TITLE","A CHRISTMAS CAROL"
Data "TITLE","GREEN EGGS AND HAM"
Data "TITLE","SATURDAY NIGHT LIVE"
Data "TITLE","BEVERLY HILLS COP"
Data "TITLE","DIARY OF ANNE FRANK"
Data "PHRASE","FROM TOP TO BOTTOM"
Data "PHRASE","TEMPORARILY OUT OF SERVICE"
Data "PHRASE","WASTE NOT WANT NOT"
Data "PHRASE","FROM RAGS TO RICHES"
Data "PHRASE","DONT TOUCH THAT DIAL"
Data "PHRASE","I NEVER SAW IT COMING"
Data "PHRASE","HAVE I GOT A DEAL FOR YOU"
Data "PHRASE","BURNING THE MIDNIGHT OIL"
Data "PHRASE","UNTIL WE MEET AGAIN"
Data "PHRASE","FRONT AND CENTER"
Data "PHRASE","NEEDLE IN A HAYSTACK"
Data "PHRASE","KEEP YOUR EYES PEELED"
Data "PHRASE","THE BUCK STOPS HERE"
Data "PHRASE","THE WALLS HAVE EARS"
Data "PHRASE","PATIENCE IS A VIRTUE"
Data "PHRASE","EVERY DOG HAS HIS DAY"
Data "PHRASE","THE BEGINNING OF THE END"
Data "PHRASE","AS LUCK WOULD HAVE IT"
Data "PHRASE","EQUAL OPPORTUNITY EMPLOYER"
Data "PHRASE","GET A GRIP ON YOURSELF"
Data "THING","UNIDENTIFIED FLYING OBJECT"
Data "THING","WALL STREET JOURNAL"
Data "THING","TOLL FREE NUMBER"
Data "THING","BROOKLYN BRIDGE"
Data "THING","MAGAZINE ADVERTISEMENT"
Data "THING","WIRELESS REMOTE CONTROL"
Data "THING","STEREO CASSETTE DECK"
Data "THING","ALARM CLOCK BUZZER"
Data "THING","GREAT WALL OF CHINA"
Data "THING","DIGITAL CLOCK RADIO"
Data "THING","OFF BROADWAY PLAY"
Data "THING","TELEPHONE EXTENSION"
Data "THING","YELLOW LEGAL PAD"
Data "THING","SPIRAL STAIRCASE"
Data "THING","DEEP DISH PIZZA"
Data "THING","FLUORESCENT LIGHT FIXTURE"
Data "THING","EGYPTIAN PYRAMID"
Data "THING","REVOLVING DOOR"
Data "THING","ELECTRIC GUITAR"
Data "THING","CHRISTMAS TREE ORNAMENT"
Data "THING","AUTOMATIC TRANSMISSION"
Data "THING","CONTACT LENS"
Data "THING","THREE SEASON PORCH"
'
Procedure Dne
  Gosub Fix_color(0)
  End
Return
'
Procedure Dne_error
  Gosub Fix_color(0)
  Print Err
Return
```

MOUSE OF FORTUNE

END

ASSEMBLY LINE LISTING 1

ASSEMBLY

```
*****************************************************************************
*
* Assembly Line Nr 9: Multiplying and Converting Numbers
*
* by Douglas Weir
*
* Copyright 1988, ST-Log
*
*****************************************************************************

        text


getnbr:
        move.l  #str_top,-(sp)          push address of template
        move.w  #$0a,-(sp)              code=read edited string
        trap    #1                      do it
        addq.l  #6,sp                   pop args
        movea.l #string,a0              get address of string
        clr.l   d0                      clear d0 for indexing
        move.b  str_cnt,d0              get nr of chars input
        move.b  #0,0(a0,d0.l)           insert a null terminator
        move.l  #string,-(sp)           get address of string
        bsr     cvnbr                   convert to number
        addq.l  #4,sp                   pop arg
        tst.b   d1                      error?
        bmi     too_long                if so
        move.l  d0,d7                   else save conversion
        bsr     c_return                print newline

        move.l  #str_top,-(sp)          push address of template
        move.w  #$0a,-(sp)              code=read edited string
        trap    #1                      do it
        addq.l  #6,sp                   pop args
        movea.l #string,a0              get address of string
        clr.l   d0                      clear d0 for indexing
        move.b  str_cnt,d0              get nr of chars input
        move.b  #0,0(a0,d0.l)           insert a null terminator
        move.l  #string,-(sp)           get address of string
        bsr     cvnbr                   convert to number
        addq.l  #4,sp                   pop arg
        tst.b   d1                      error?
        bmi     too_long                if so
        bsr     c_return                else print newline

        move.l  d7,-(sp)                get first operand
        move.l  d0,-(sp)                get second operand

        bsr     mul32                   multiply them

        move.l  (sp)+,d0                get high longword
        move.l  (sp)+,d1                get low longword
        move.w  #FALSE,-(sp)            no leading zeros
        bsr     cv64                    convert to ascii
        addq.l  #2,sp                   pop arg
        move.l  #dgts,-(sp)             get address of string
        move.w  #$09,-(sp)              code=print string
        trap    #1                      do it
        addq.l  #6,sp                   pop args
        bsr     c_return                print newline

        bsr     c_return                print a newline
        bra     getnbr                  go back to start...

too_long:
        move.l  #oo_string,-(sp)        "input overflow" string
        move.w  #$09,-(sp)              code=print string
        trap    #1                      do it
        move.w  #0,-(sp)                code=terminate
        trap    #1                      do it

*****
*
* c_return-- prints a newline to the screen.
*
*****
c_return:
        movem.l d0-d2/a0-a2,-(sp)       save registers
        move.w  #$0d,-(sp)              carriage return
        move.w  #2,-(sp)                code=print character
        trap    #1                      do it
        addq.l  #4,sp                   pop args
        move.w  #$0a,-(sp)              line feed
        move.w  #2,-(sp)                code=print character
```

```
        trap    #1                      do it
        addq.l  #4,sp                   pop args
        movem.l (sp)+,d0-d2/a0-a2       restore registers
        rts                             and return

*****
*
* div64-- performs stupid 64-bit division.
*
* at entry:
*               d0.l contains high longword of dividend
*               d1.l contains low longword of dividend
*               d2.l contains high longword of divisor
*               d3.l contains low longword of divisor
* at exit:
*               d0.l contains high longword of quotient
*               d1.l contains low longword of quotient
*               d2.l contains high longword of remainder
*               d3.l contains low longword of remainder
*               d4.l contains BAD if divide by zero, else 0
*                 (not checked by cv64)
*               all other registers preserved
*
*****
div64:
        movem.l d5-d7,-(sp)             save registers
        clr.l   d4                      clear subtract counter high
        clr.l   d5                      clear subtract counter low
        tst.l   d2                      divisor high zero?
        bne.s   div_loop                if not, start
        tst.l   d3                      else: divisor low zero?
        beq.s   div_zero                if so
div_loop:
        cmp.l   d2,d0                   compare high longwords
        beq.s   div_lp0                 if equal, continue comparison
        bcc.s   div_count               else, subtract if dividend >=
        bra.s   div_out                 else we're done
div_lp0:
        cmp.l   d3,d1                   compare low longwords
        bcc.s   div_count               subtract is ok
        bra.s   div_out                 else, finished
div_count:
        andi    #$0f,ccr                clear extend bit
        subx.l  d3,d1                   subtract low longwords with extend
        subx.l  d2,d0                   subtract high longwords with extend
        addi.l  #1,d5                   count subtract (low)
        bcc.s   div_c0                  if no carry
        addi.l  #1,d4                   else add carry
div_c0:
        bra.s   div_loop                and continue
div_out:
        move.l  d0,d2                   return remainder high
        move.l  d1,d3                   return remainder low
        move.l  d4,d0                   return quotient high
        move.l  d5,d1                   return quotient low
        move.l  #0,d4                   no error
div_exit:
        movem.l (sp)+,d5-d7             restore registers
        rts                             and return

div_zero:
        move.l  d0,d2                   return remainder high
        move.l  d1,d3                   return remainder low
        move.l  #0,d1
        move.l  #0,d0
        move.l  #BAD,d4                 error code
        bra.s   div_exit                and leave


*****
*
* cv64-- converts an unsigned 64-bit binary number to ascii.
*        The largest possible unsigned 32-bit number is 4294967295.
* at entry:
*               d0.l contains number to be converted, high longword
*               d1.l contains number to be converted, low longword
*               (a6 + 8) -> leading zeros flag (word):
*                           TRUE => do print leading zeros
*                           FALSE => do not print leading zeros
* at exit:
*               'dgts' (in data segment) contains converted number.
*               all registers preserved.
*
*****
cv64:
```

```
        link    a6,#0                   link register
        movem.l d0-d7/a0-a1,-(sp)       save registers
        movea.l #dgts_end,a0            point to end of string space
        move.b  #0,(a0)                 insert null terminator
        moveq.l #d_len,d5               get length of string space
        bra.s   cv_test                 clear string space
cv_iloop:
        move.b  #BLANK,-(a0)            clear a byte
cv_test:
        dbra    d5,cv_iloop             go till end
        tst.w   8(a6)                   check leading zero flag
        beq.s   cv_nxt0                 if FALSE
        move.l  #FALSE,d6               otherwise, don't check
        bra.s   cv_nxt1
cv_nxt0:
        move.l  #TRUE,d6                leading zeros = true
cv_nxt1:
        movea.l #divisors,a1            point to start of array
        move.l  (a1)+,d2                get first divisor high longword
        move.l  (a1)+,d3                get first divisor low longword
        moveq.l #d_len,d5               get place index
        neg.l   d5                      index from back to front
        movea.l #dgts_end,a0            point to end of string space
cv_loop:

        bsr     div64                   do 64-bit division

        cmp.l   #TRUE,d6                leading zeros?
        bne.s   cv_nxt2                 if not
        tst.b   d1                      else: is this a zero?
        beq.s   cv_nxt3                 if so
        move.l  #FALSE,d6               else set flag = not leading zeros
        bra.s   cv_nxt2                 and go insert digit
cv_nxt3:
        move.b  #BLANK,d1               use space vice leading zero
        bra.s   cv_nxt4                 and insert it
cv_nxt2:
        addi.b  #ZERO,d1                convert to ascii
cv_nxt4:
        move.b  d1,0(a0,d5.l)           insert into string space
        move.l  d3,d1                   recover remainder low
        move.l  d2,d0                   recover remainder high
        move.l  (a1)+,d2                get next divisor high
        move.l  (a1)+,d3                get next divisor low
        addq.l  #1,d5                   decrement place index
        bne.s   cv_loop                 go till end

        cmp.l   #TRUE,d6                was number a zero?
        bne.s   cv_exit                 if not
        subq.l  #1,d5                   else find last digit
        move.b  #ZERO,0(a0,d5.l)        and insert a zero
cv_exit:
        movem.l (sp)+,d0-d7/a0-a1       restore registers
        unlk    a6                      deallocate frame
        rts                             and return


*****
*
* cvnbr-- converts a string of ascii digits to a number.
*         the digits themselves are not checked for errors.
* at entry:
*               (a6 + 8) -> string to convert (null-terminated).
* at exit:
*               d0 contains converted number
*               d1 contains BAD if string is too long, else 0
*               all other registers preserved.
*
*****
cvnbr:
        link    a6,#0                   frame pointer
        movem.l d2/a0-a1,-(sp)          save registers
        movea.l 8(a6),a0                get string
        clr.l   d0                      clear counter
        move.w  #MAX_DIGITS+1,d1        max size of string
        bra     front_test              get least sig digit
front_loop:
        addq.l  #1,d0                   count one place
        tst.b   (a0)+                   null?
front_test:
        dbeq    d1,front_loop           if not and <= max length

        tst.w   d1                      too long?
        bmi     front_error             if so
        subq.l  #1,d0                   else uncount delimiter
```

```
        subq.l  #1,a0                   point back to null
        clr.l   d1                      clear digit holder
        clr.l   d2                      clear number holder
        movea.l #factors,a1             point to array of factors
        bra     cvn_test                start conversion
cvn_loop:
        clr.l   d1                      clear upper bytes
        move.b  -(a0),d1                get a digit
        subi.b  #ZERO,d1                convert it from ascii
        move.l  d1,-(sp)                push as first operand
        move.l  -(a1),-(sp)             current factor = 2nd operand
        suba.l  #4,a1                   skip divisor longword
        bsr     mul32                   multiply them
        move.l  (sp)+,d1                discard high longword
        move.l  (sp)+,d1                low longword = product
        add.l   d1,d2                   add to total
cvn_test:
        dbra    d0,cvn_loop             continue to end
        clr.l   d1                      no error
        move.l  d2,d0                   return number
cvn_exit:
        movem.l (sp)+,d2/a0-a1          restore registers
        unlk    a6                      deallocate frame
        rts
front_error:
        move.l  #BAD,d1                 error code
        bra     cvn_exit                now leave


*****
*
* mul32-- performs complicated 32-bit multiplication.
*
* at entry:
*               (a6 + 8)  -> first operand.
*               (a6 + 12) -> second operand.
* at exit:
*               after return to caller,
*               (a7) -> high longword of product.
*               (a7 + 4) -> low longword of product.
*
* all registers preserved.
*
*****
mul32:
        link    a6,#0                   frame pointer
        movem.l d0-d7/a0-a2,-(sp)       save registers

        clr.l   d0                      clear registers
        clr.l   d1
        clr.l   d4
        clr.l   d7
        movea.l #m_subs,a1              base address of swap routines
        move.l  8(a6),d5                get operands
        move.l  12(a6),d6
        lea     12(a6),a0               point to result area, low word
        clr.l   8(a6)                   clear product area
        clr.l   12(a6)
m_loop:
        move.w  d5,d2                   get partial operands
        move.w  d6,d3
        mulu    d2,d3                   multiply them

        tst.b   d4                      carry from last add?
        beq.s   m_nxt0                  if not
        addq.w  #1,0(a0,d0.l)           else add it in
m_nxt0:
        add.l   d3,0(a0,d0.l)           add partial product
        scs     d4                      save possible carry
        addq.l  #1,d1                   count loop
        btst.l  #0,d1                   odd iteration?
        beq     m_nxt1                  if not
        subq.l  #2,d0                   else index product up one word
m_nxt1:
        cmp.l   #12,d7                  finished?
        beq.s   m_out                   if so
        movea.l 0(a1,d7.l),a2           else get swap routine
        jsr     (a2)                    execute it
        addq.l  #4,d7                   index for next one
        bra     m_loop                  and continue
m_out:
        movem.l (sp)+,d0-d7/a0-a2       restore registers
        unlk    a6                      deallocate frame
        rts                             and return
```

```
m_swap_1:
        swap    d5
        rts
m_swap_2:
        swap    d5
        swap    d6
        rts
m_swap_3:
        swap    d5
        rts


*****
*
* data segment
*
*****
                data
                even
m_subs          equ     *       address table of subs for 'mul32'
                dc.l    m_swap_1
                dc.l    m_swap_2
                dc.l    m_swap_3

MAX_DIGITS      equ     10      max length of digit string
BAD             equ     -1      error code
ZERO            equ     48      ascii '0'
FALSE           equ     0       false constant
TRUE            equ     1       true constant
BLANK           equ     32      ascii ' '

oo_string       dc.b    $0d,$0a,'oops... number is too big',$0d,$0a,0
dgts            ds.b    20      string area for ascii number
d_len           equ     20      length of string space
dgts_end        dc.b    0       null at end of string

                even
divisors        dc.l    $8ac72304       divisors for ascii conversion:
                dc.l    $89e80000       highest divisor is 1 followed by
                dc.l    $0de0b6b3          19 zeros
                dc.l    $a7640000
                dc.l    $01634578
                dc.l    $5d8a0000
                dc.l    $002386f2
                dc.l    $6fc10000
                dc.l    $00038d7e
                dc.l    $a4c68000
                dc.l    $00005af3
                dc.l    $107a4000
                dc.l    $00000918
                dc.l    $4e72a000
                dc.l    $000000e8
                dc.l    $d4a51000
                dc.l    $00000017
                dc.l    $4876e800
                dc.l    $00000002
                dc.l    $540be400
                dc.l    0
                dc.l    1000000000
                dc.l    0
                dc.l    100000000
                dc.l    0
                dc.l    10000000
                dc.l    0
                dc.l    1000000
                dc.l    0
                dc.l    100000
                dc.l    0
                dc.l    10000
                dc.l    0
                dc.l    1000
                dc.l    0
                dc.l    100
                dc.l    0
                dc.l    10
                dc.l    0
                dc.l    1
factors         dc.l    0               factors for number conversion
                dc.l    0

str_top         dc.b    81              max nr of chars to input
str_cnt         ds.b    1               byte for count
string          ds.b    80              beginning of actual string
```

Assembly Line

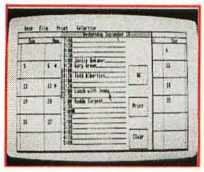
END

YOU NEED THE DISK!

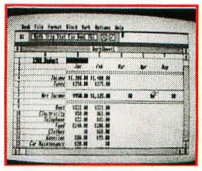
If you want to get the most out of ST-LOG, you're also going to want to get your copy of the disk. Each issue's disk contains all the exciting programs for that issue, including the programs' whole listings, which could not be included due to space considerations. The ST-LOG disk version is truly an excellent software value. Order yours today!

ONLY $9.95 EACH!

DECEMBER 1988
ST Date Planner, Mouse of Fortune, Inside ST Xformer II and more!

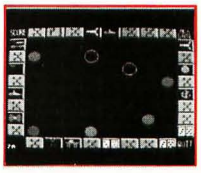


SEPTEMBER 1988
Opus DEGAS Fast Loader Double-column Printing And More!



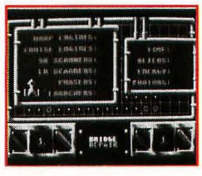
NOVEMBER 1988
ThetaTen Ultra-Graph Number Maze And More!

AUGUST 1988
Merlin's Box MID Mon GEMKit And More!



OCTOBER 1988
Monkeys & Balloons Spectral Sorcery And More!

JULY 1988
Moonlord ST Busy Buddy Battle Blips And More!



YES! I DO WANT THE DISK! ONLY $9.95 EACH

- ☐ ST-LOG July 1988 Disk
- ☐ ST-LOG August 1988 Disk
- ☐ ST-LOG September 1988 Disk
- ☐ ST-LOG October 1988 Disk
- ☐ ST-LOG November 1988 Disk
- ☐ ST-LOG December 1988 Disk

Name_____

Address_____

City_____ State_____ Zip_____

☐ Payment Enclosed—Charge My ☐ VISA ☐ MC

#_____ Exp._____

Signature_____

Add $1.50 postage and handling for each disk ordered.
Make check payable to: LFP, Inc. P.O. Box 67068, Los Angeles, CA 90067
California residents add 6.5%

ST LOG
The Atari ST Monthly Magazine

NEW NEWS IS GOOD NEWS: X*PRESS

X*Press Information Services
Regency Plaza One
4643 S. Ulster St.
Suite 340
Denver, CO 80237
1-800-7PC-NEWS

Reviewed by Andy Eddy

Welcome to the Information Revolution!

With the aid of your computer, you can access databases of all kinds to dig up reams of material—government records, business and financial info, airline reservation services and schedules—almost anything under the sun. The only downside, in the majority of cases, is the cost: many services charge an arm and a leg for this availability, breaking it down into "per minute" rates. And since virtually all of these services are laid out in a unique manner, often employing hard-to-learn command structures, many times you're left spending a great deal of time looking for the information you need.

An alternative system has started up, based in Colorado, that delivers the news and financial data in real-time right to your door—or more aptly put, through your serial port and right up to your computer screen. More importantly, it doesn't tie up your phone line, as is the case with other services of this genre. X*Press, as it is called, is delivered to your computer by way of a standard cable television line (provided your local system offers the service) or by satellite dish—at a whopping 9600 baud. We'll discuss the choices in the methods of transportation later; first, let's look at what X*Press is.

First there were newspapers, but now. . . .

X*Press is akin to having a news ticker in your home. Through it, you can scroll through story after story from a wealth of sources, both national and international. In addition, you have access to stock prices; all you need to do is set up a portfolio of up to 128 stocks by their symbols and the software automatically updates the current price, high and lows up to that time, and charts up and down arrows as "ticks" for monitoring the latest activity of the stock. This is invaluable to both the dabbling and the experienced investor.

For the news hound X*Press is equally powerful. With tons of

sources, you can keep up with current events easily. This not only includes national coverage, but imported stories from international news bureaus in many countries. X*Press Information Providers (IP) include such big names as McGraw-Hill, Associated Press, United Press International, and Cable News Network, as well as international sources such as TASS (Russia's news agency), Xinhua (China's news service), and The OPEC News Agency. New IPs are added frequently to keep the service expanding.

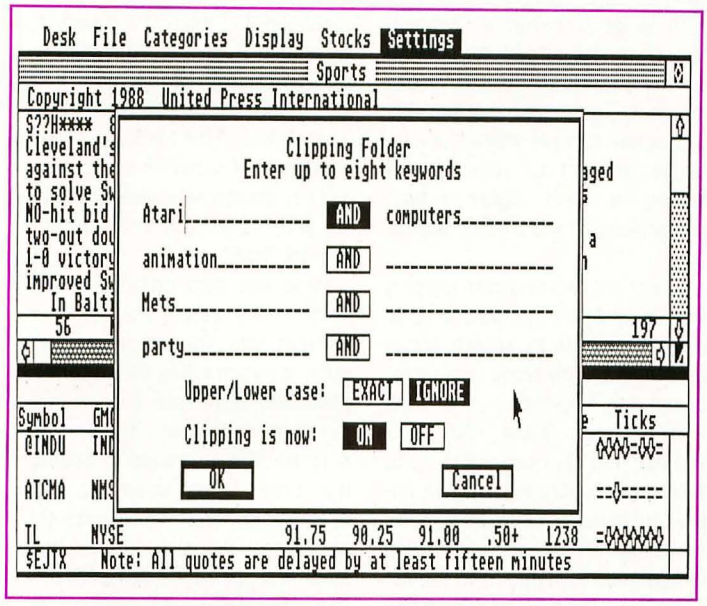
Of course, the X*Press service isn't restricted to such serious topics as financial information and current events coverage. You'll find other sections pertaining to sports, weather (both stateside and international reports), entertainment and shopping, among others. Within these areas you can find such diverse text as movie and book reviews, horoscopes and computer chat.

There are also two subjects that deal more directly with X*Press itself. The first, "Inside X*Press," has sub-categories that inform users of the latest features and additions to the service, hints and tips, and commentary on the system.

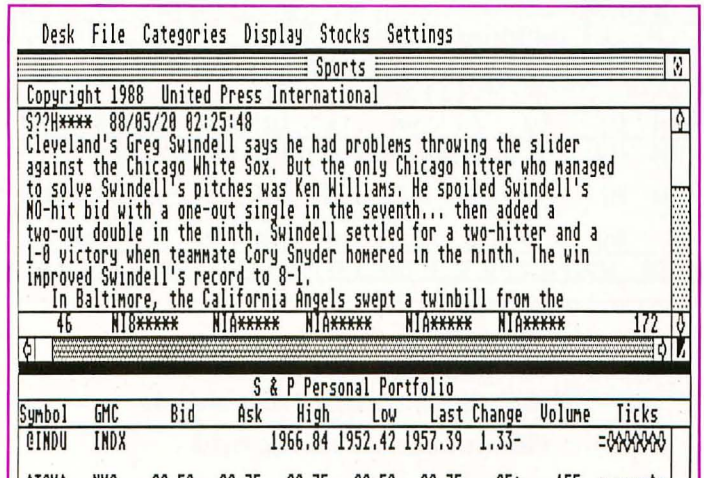
The other category is "Information X*Change." This is a two-way conferencing service, allowing X*Press users to speak their mind on topics of import under such topic headings as Inside Your Head, Teen Talk and Computers & You.

Now you might wonder how a seemingly one-way system, as X*Press is, can become a forum for discussion. X*Press has set up a number of methods for its subscribers to pass their comments along: 1) Comments can be sent by mail; 2) comments can be relayed through their toll-free phone number, after which they will then be transcribed and posted; 3) X*Press' "mailbox" can be called by computer and the text of your comment uploaded. (This is a toll-call to most of the nation); and 4) you can call X*Press' mini-BBS, which is accessible through a local phone connection via Tymnet.

In addition, Linda Nicholson, X*Press's conferencing coordinator, said that they plan on expanding the service to offer a toll-free data line that you can employ to upload your comments.



"The best feature of X*Press is the ability to tailor the software for what type of news you are most interested in."



Be your own editor

The best feature of X*Press is the ability to tailor the software for what type of news you are most interested in. Through the category selection process, you can move from menu to menu picking which topics and sources you want to receive data from, and the software will toss into memory all stories that meet those criteria. It's somewhat like having a custom-made newspaper delivered to you at whatever time you choose.

In the Category Selection menu, there are listings of major areas of interest: Finance, Business, News, Sports, Weather, Lifestyles, Entertainment, Tech Talk, Shopping, Information X*Change and Inside X*Press. If you click on a category, you'll also get a list of subcategories; for example, the Inside X*Press category has subcategories of Bulletins, News, What To Watch For, User Tips and New Services. Certain subcategories have indicators next to them that indicate a sub-subcategory, such as the state-by-state list under the USA State listing in the Weather category.

All this procedure does is to narrow down what your focus is, which serves to trigger the software to flag any stories within those specified sections and pop them into memory. Once you go through the entire selection process and have configured the software to the areas of interest to you, clicking on Save Settings will write those selections to a configuration file on the program disk. This will ensure that the same settings are loaded in each time you run X*Press, though you can change at anytime within a session.

The stories can be pulled up at any time for scanning by selecting the Display header from the menu bar. You can show all stories in the display window, or pick a specific category to view. You can also bounce back and forth between the window for story viewing and the window that holds the stock listings. As long as the pro-

gram is running, it will keep receiving stories and stock figures, refreshing the number of stories until the memory is filled to capacity.

Now the good news

I've had the opportunity to see the preliminary versions of the ST software, basically from the start of testing, and watching that evolution tells me that it could be the most feature-filled link to X*Press to date. The ST version programmed by Alan Page — who's best known for his work as co-author of the popular and powerful terminal software, *Flash*—has included many innovations. Thanks to the speed and memory capacity of the ST, Page had the room he needed to make the X*Press and ST pairing very versatile. Here's a glance at what is possible on the ST version of X*Press that isn't available to other PC versions.

In other versions, you can enact a keyword search for any or all of 16 keywords (from a list you set up) appearing in the stories in memory. As an example, when the Oscar nominations were announced, we activated the software (using an in-house IBM PC in this instance) to flag stories in the Movie subtopic in the Entertainment section. Later, we had the software scan through the existing stories in memory for any occurrences of the word "Oscar," exactly as you would do with the Search command in a word processor.

In the other versions of the software, any story you look at must have had its category previously selected, and you must be insightful in your choice of categories so your keyword search will be lucrative. In fact, the first showing of the Oscar listings we were clamoring for came down the line on the French press in the News category, not from the Entertainment area, as we expected. In a more important, and possibly timely, situation—such as hoping for speedy information on a possible business takeover of a company, which may drastically affect the price of a stock—not knowing what category a potential story will reside in could become a major stumbling block to your search.

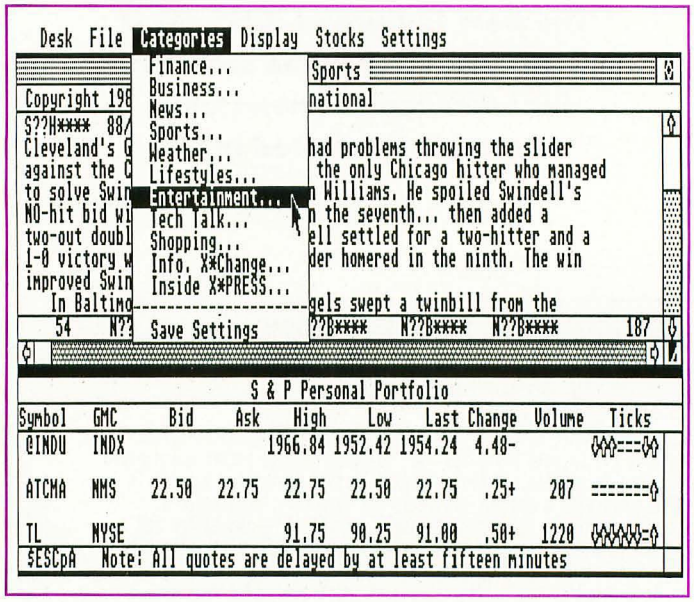
While the ST software has a similar keyword option as that mentioned previously, the ST's speed also permitted Page to expand on the keyword segment with what is called a "clipping folder," to poll all stories coming down the line for keyword matches. The main benefit to this is that you can get stories of particular interest to you, without having to first figure which categories that word might appear in.

There are two separate clipping folders, and you can choose up to eight keywords to search for in each folder, with some options for comparing keywords—like Time AND Inc., or Oscar AND NOT Grouch. You also can set it up to take into account matches in upper and lower case if you choose, so using "Time" as a keyword would more likely point to text on Time, Inc., as opposed to just any occurrence of the word "time."

Using the prior example, with the ST version you could plug the word "Oscar" into the keyword list of the clipping folder dialog box. Then the software would grab all stories that contained that word into a packet, separating them from the regular story acquisition, just as you would if you were clipping stories from a newspaper. This would occur regardless of whether or not you had any categories selected, focusing your viewing to the subjects you are interested in.

Page has also programmed a multitasking feature into X*Press ST that lets you run most any other program while the software continues to receive stories and put them in memory. The cost is a 15 to 20% decrease in speed, the time being used by the X*Press software to constantly poll the serial port for stories. You must first set up the software before running it, to allocate a smaller chunk of memory to X*Press' capture buffer. This will reserve some of the RAM to the other application(s) you plan to run, be it word processor, spreadsheet or game software.

Again, the X*Press program will continue to monitor the data feed in the background of the application you run, although the smaller storage area will limit your story acquisitions somewhat. There's no doubt that this won't be as much of a limitation once users expand the internal RAM of their STs to a point beyond one megabyte or when more Megas are purchased. To aid the ST user further, Page has added text compression to let more stories fit in memory.



"There's no doubt that this is the future of news and information retrieval, and the variety is immense."

X*press' two services and how to get them

Actually, X*Press is broken down into two different services: X*Change and Executive. The main difference between the two is in how much "white-collar" data you want to receive and how much you want to pay for it. Executive offers two other categories that X*Change doesn't have, Business and Finance, and lets the stock investor track a portfolio all day (with a 15-minute delay); X*Change updates the stocks for its users at the market's opening, midday and closing.

The other difference is in cost, with X*Change being offered as a "basic," no-monthly-charge service and Executive being charged at a recommended $19.95 a month. Each service requires the purchase of a kit which contains the software particular to your computer brand and hardware to convert the signal coming off the cable into a data stream readable by your computer and the X*press software.

The package even contains a file-transfer program, so you can keep your software current and receive the new categories X*Press offers. X*Press schedules occasional dates and times (done at night to minimize your loss of the service during prime times) for file transfer. You set up your computer with the file-transfer software running, and when X*Press starts the transfer, a coded packet is received, and the files are

saved to disk. Checksums are employed and error counts are tracked to verify an accurate download.

There's no doubt that this is the future of news and information retrieval, and the variety is immense. One other variation to the story is what you can do with this information once you get it. Some packages already exist for the IBM compatibles allowing you to export the data to other software, such as a spreadsheet or word processor. The ST version has a menu selection for loading in or saving out stories to disk in ASCII format.

But the largest potential is as an educational tool. If each school had X*Press, you'd have a real-time news source for current events, stock and finance information for accounting classes, and even a source of Spanish language text with Notimex, the Mexican news service. The Information X*Change is a great venue for keeping up with sentiments of others around the country.

They also have a program called X*Change In The Schools, that provides free software/hardware kits and support materials so that educational facilities may benefit from the resources the connection brings. Once X*Press, like the cable TV wires themselves, is exposed to a large volume of the country, we'll see them becoming a force in education—perhaps as much a part of the educational process as a textbook.

X*Press has just started marketing their service more heavily, so you may not have it in your area as of yet. For those with TVRO (TV Receive Only) satellite dishes, X*Press also offers a kit for getting their signal. The difference is in the additional hardware required and the subsequent added cost.

For cable systems, the addition of equipment to the already existing hardware is minimal and reasonably inexpensive in most cases. Therefore, X*Press is finding itself in more cable systems each day—they claim to be in over 480 systems, passing over 5,000,000 homes. If your cable company doesn't currently carry X*Press, give them a call, or contact X*Press directly at 1-800-7PC-NEWS (772-6397). You'll be getting in on a chunk of the future. ■

FirstCADD

Generic Software
Redmond, Washington
(800) 228-3601
Color or monochrome $50

Reviewed by Ian Chadwick

This is the first of a series of reviews of CAD—Computer Aided Design (often called Computer Aided Drafting and Design)—programs. These programs are used by such professions as circuit designers, architects, draftsmen and surveyors.

CAD programs differ from paint programs in that they deal not with pixels but with vectors and objects. Objects are not stored as a series of pixels (bits set in video memory), but rather as points in "space," usually connected by lines. CAD programs have defined spatial environments, sometimes in three dimensions, so objects can be positioned with exactitude and sized accurately. The results are generally line-drawings, as compared with the broad canvas paintings created by *DEGAS* or *Neochrome*.

Three-dimensional CAD programs deal in planes, lines and points. Two-dimensional CAD programs deal in points and lines, but usually allow a layering effect so that 2D images can be drawn over each other to create such images as a multi-layered circuit board. Using layers and perspective, 3D effects can be simulated in 2D programs.

CAD programs provide object primitives—that is, pre-defined object types (circle, ellipse, line, rectangle, arc, spline curves, etc.) that are used as the basis of a drawing. Custom objects can usually be created as "components" by connecting lines or defining points, then saved and loaded into other drawings. This object-vector orientation also allows drawings to be saved as ASCII files, since only the points and connections need to be defined in most cases.

Because of its graphic nature, the ST is well-suited for CAD applications. Over the next five or six issues, the major contenders in this field will be reviewed.

FirstCADD is one of those rare software ports from the PC/MS-DOS world, where it is known as *Generic CADD*. The PC/MS-DOS version is up to 3.0, while the current ST version is 1.02. In the PC/MS-DOS version there are approximately 140 commands, compared to roughly 80 in the ST version.

This is an entry-level, two-dimensional CAD program, with enough features to permit complex drawing and design, using real-world measurement, but too few to be a serious professional tool. However, it serves quite well as an inexpensive introduction to CAD principles, and you can produce high-quality drawings with a modest amount of effort.

Although it offers a menu structure for command entry, FirstCADD is a TOS, not a GEM, application. The "menu" is a vertical list of commands and sub-commands which can be selected with the mouse or entered manually through the keyboard. The menu is actually a convenience rather than a necessity, and can be hidden to enlarge the drawing-display area. This mirrors the PC/MS-DOS version, but ST users will find the system awkward and clumsy compared with the smoother and more recognizable GEM menu structure. On the plus side, the menu is simply an ASCII file and can easily be customized for your own use.

The major problem with this method is that the menu cursor moves simultaneously with the drawing cursor, and you may sometimes click the mouse, think-

ing to work on a drawing but instead selecting a menu choice! Pressing Escape clears a command, although there is no Undo feature. If you find yourself in a menu without the correct command to display the previous menu (the component list, for example), you must twice type DM followed by Return to re-display the root menu.

You can also define the function keys (alone, not with Shift or Alternate) to perform any basic FirstCADD command.

Because it's not a GEM application, FirstCADD also doesn't use the convenient file selector for loading or saving files, although it does, curiously, let you look at a directory list in order to find a file. Then you must type the path and name to load the file. It is unforgiving about such things as setting the default directory. This isn't a serious problem, merely annoying. You just have to remember to type the drive and pathname if you want to load a picture from a drive or path different than the one from which you loaded the program.

Drawings are made from the essential drawing entities, or primitives, the program supports (point, line, rectangle, two and three point circles, three and four point arcs, ellipses, regular polygons and b-spline, or complex, curves). Drawing is quite straightforward, using the mouse to define start and end points for most objects. The program takes care of the actual connections between points. Some objects, such as lines and rectangles, can be drawn with a "rubber band" effect so you can see where the boundaries are as you position them.

Once drawn, a line (most types, including curves and text) can be edited, recolored, moved, copied and changed. It's a good idea to turn on the display of construction points before starting to draw, because working with the cursor and the X/Y coordinates alone is difficult. The status display shows both the coordinates for the absolute X/Y and the delta X/Y (the relative distance from the last point) in the current unit of measurement.

Groups of lines within a specified window can be defined as a component, named, and saved in your own library. Components of any complexity can be loaded into any drawing on any layer. Components can also be scaled, rotated and removed from a drawing.

FirstCADD permits up to 256 transparent layers, which can be individually or collectively viewed. You can draw in and edit any individual layer, even draw in one layer with others displayed, but unlike the PC/MS-DOS version, layers cannot be saved individually. All layers must be saved together in a single drawing.

Lines and objects can be snapped to the definable grid, either at the grid point, the nearest drawing point or to a point within the definable tolerance level. Tolerance permits you to snap a line to the point nearest the cursor, within the defined limit.

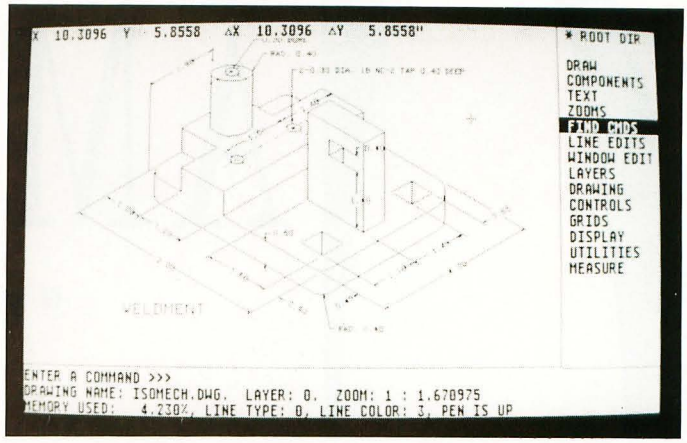
There are three forms of measurement available in the program. You can measure distance between any two points, the area of any number of connected, straight-line segments, and the angle between any two points.

You can zoom in on any area of the drawing using several options. You can specify a zoom value, define a zoom window, or zoom to predefined limits. You can also zoom backwards and forwards from any other zoom state.

Text is a special type of component in FirstCADD. Only one rather simple font is provided (the PC/MS-DOS version has seven), but a built-in font editor permits the user to define fonts by creating each letter, much the same way as one would create any other component—by drawing the lines and primitives for each character. Fonts, therefore, can be made of various components, not necessarily just characters. However, the font creation routine, while sounding simple in the manual, is much trickier when you attempt it. Experiment and practice a lot first. But be careful not to overwrite your "main" font!

If you choose Font Select after you've chosen Start Define, your disk will whirl away endlessly, and the program will hang, while

"CAD programs differ from paint programs in that they deal not with pixels but with vectors and objects."



seemingly writing to the drive all the while! You can select any zoom, component or drawing command, but if you select another text command other than Define Start or Define End, you may end up with a hung program. This is the only serious bug I've found in the program.

FirstCADD supports 105 printers, including a wide range of the most popular dot-matrix printers. The Atari laser printer isn't on the list, but the HP Laserjet series is. Only one printer at a time can be selected, using the configuration program (not from within the actual CAD program).

When you print ("plot" in the manual) a drawing, you have several controls over the result. You can print it to scale either according to the paper size or to your own defined scale. You can even rotate a drawing 90 degrees. However, you can print only what you see on the screen. You cannot print a large drawing in several sheets if it is too large for a single sheet. While a drawing is theoretically unlimited in size (as set by the configuration program), the output *is* limited to your sheet size. That can mean some cramped results on a single page.

FirstCADD is slow in three areas: printing, loading a file and screen redraws. The latter two are not excessive, albeit annoying at times, but when you start printing a complex drawing, be prepared to spend many long minutes waiting for completion.

The major failing of the program lies in its pitifully weak documentation, a mere 100 pages or so, lacking illustrations, a tutorial or even examples of commands. The manual reads as if it was hurriedly and unthinkingly dashed off, a sort of slash and burn editing of the PC/MS-DOS manual. This is unfortunate, because it makes the program appear much weaker and less capable than it really is.

There is also no description of error messages. The index is sparse. The commands are explained in as few words as possible, usually ignoring problems, reliable descriptions and side effects. For the novice user, it will take a lot of experimentation and careful reading to properly understand how the program operates and the caveats that come with it. The PC/MS-DOS version, in contrast, is more than 500 pages long, illustrated, explained and considerably clearer.

Once, however, you surmount the difficult slope the manual presents, FirstCADD is reasonably easy to use and understand. For the newcomer to CAD and the non-professional user, it is an inexpensive entry into a demanding field. While it lacks the ultra-slick features of a program like *DynaCADD*, it has sufficient power for the casual user at a considerably lower price.

THE GAMESHELF

By Clayton Walnum

Bubble Ghost
Accolade
550 S. Winchester Blvd.
Suite 200
San Jose, CA 95128
(408) 296-8400
Color only $34.95

Now here's a game with a hint of originality. You can forget blasting aliens at speeds beyond the tolerance of the human trigger finger; you can forget warping through galaxies on dangerous space-warring missions; you can even forget trying to find your way through a D&D-like scenario with a batch of role-playing characters.

In *Bubble Ghost*, there's this bubble—just a bubble.

Actually, there's also this little guy called Bubble Ghost, and his mission—with your help via the mouse—is to guide the bubble through each of 35 different rooms. Bubble Ghost can go anywhere, touch anything, and is completely indestructible—but the bubble, just like all bubbles, has to be handled with the greatest of care. If it touches anything, it pops. Each time you lose a bubble, Bubble Ghost will punish you with an impatient glare.

How, you may ask, can you move the bubble without touching it? You (or Bubble Ghost, if you want to get picky), must *blow* on the bubble, and it will then move in the direction you've sent it. Get the bubble to move across the room and to the exit and you've completed that screen and can move on to the next.

I suppose I should mention (though you've undoubtedly figured it out already) that each room has a set of obstacles that you must get the bubble by. Not only do you have to keep the bubble away from the room's walls, but you must also keep it away from dozens of contraptions—candles, fans, needles, and even some electronic gadgetry—each obstacle designed to make your travel through the room fraught with danger.

The first couple of rooms are easy, but the further you progress, the tougher things get. But don't fret. Once you play for a while, you'll learn a few secret tricks

that'll help you on the way. (For instance, if you blow on a candle, it'll go out.) Learning these tricks will be a painless process. Because *Bubble Ghost* requires precision rather than speed, you won't tire your mouse hand out. In addition, the game's habit-forming nature will keep you playing for hours; you'll pick up most of the tricks just by experimentation.

Bubble Ghost's manual is short and to the point, giving the player what he needs to know to play the game and nothing more. That's not a complaint, though. It's actually refreshing once in a while to pick up a game that can be played without reading a novel-length manual.

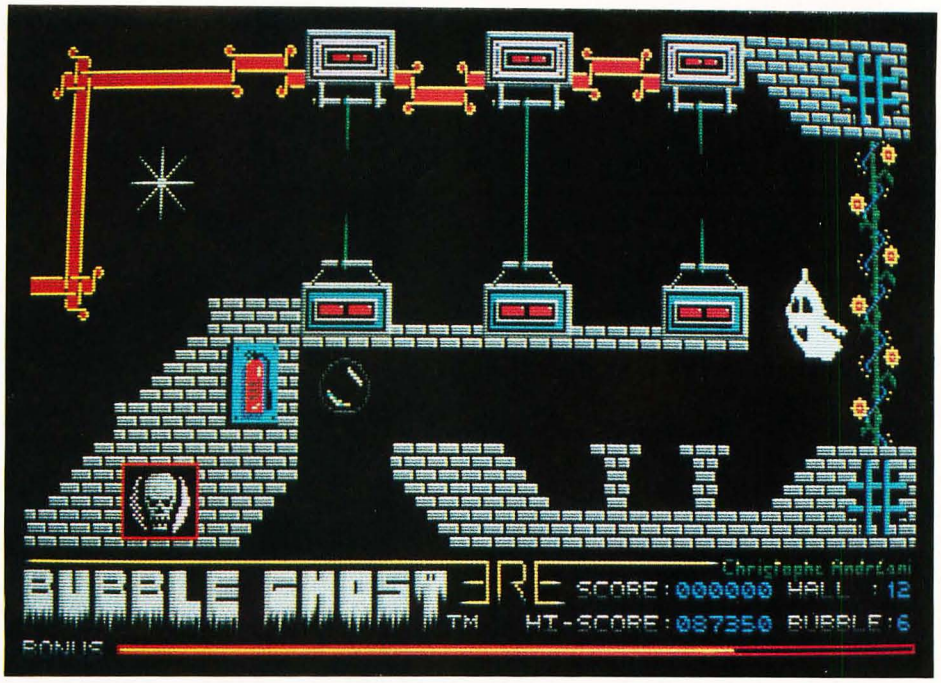
The game's graphics are fine—the usual high-quality ST graphics we've become accustomed to seeing in game programs.

If there's anything to complain about, it's the sound. I had to set my monitor to maximum volume in order to hear most of the sound effects. That in itself is not a major problem. Unfortunately, a few of the sound effects are significantly louder than the rest, which makes it difficult to get a good balance.

But minor problems like low-volume sound effects don't affect this game's charm and originality. Bubble Ghost is clearly a winner.

Recommendation: Buy it. ■

BUBBLE GHOST · from Accolade



"Bubble Ghost's mission, with your help via the mouse, is to guide bubble through each of 35 different rooms."

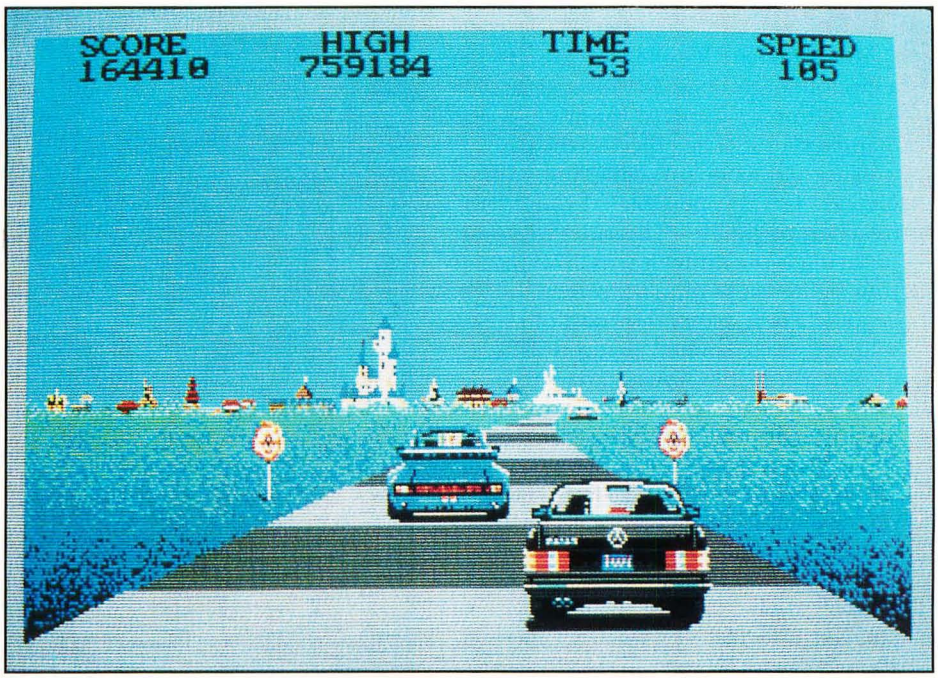
"In Crazy Cars, you race your car on a two-lane-wide road, accelerating, braking and steering your way past the other racers."

Crazy Cars
Titus Software
20432 Corisco Street
Chatsworth, CA 91311
(818) 709-3693
Color only $39.95

Way back when I saw my first home video-game system (yes, it was an Atari), one of the cartridges I saw demonstrated was a racing game called *Night Driver*. I won't say that Night Driver was the first game of its type, but I will say that since that time dozens of similar driving "simulators" have been released. They all look and play the same: the car remains "stationary" on the screen, while the road whips toward you, giving the illusion of high speed. As you manuever around corners, the background scenery shifts left or right, further enhancing the illusion of movement.

This type of game probably hit its peak of popularity with both the arcade and home versions of *Pole Position*, and the new versions have offered little new. *Crazy Cars* is yet another in this long chain of racing games and, I'm sorry to say, can't

(to page 93)



CRAZY CARS · from Titus Software

Paladin
Omnitrend Software
P.O. Box 3
West Simsbury, CT 06092
(203) 658-6917
Color only $39.95

Omnitrend software has been around for a while. My first experience with them was an immense game (four disks!) for the 8- bit Atari computer line called *Universe.* Universe was an extremely complicated game, creating within the confines of those four disks an entire galaxy to explore. It was the type of game you had to dedicate every waking moment to.

By the time Thomas Carbone and William Leslie, the two young men who were the sole staff of Omnitrend (doing both the programming and the marketing themselves) were ready for another project, they had decided that their ideas had already grown past the limits of the 8-bit computers and announced that the sequel to Universe, aptly titled *Universe II,* would run only on the new wave of home microcomputers, and luckily for us that new wave included the ST.

I say "lucky for us" not so much because I was particularly enamored of Universe II. That game was just as immense and complicated as its predecessor—not the type of game my *(to page 93)*

PALADIN · from Omnitrend Software

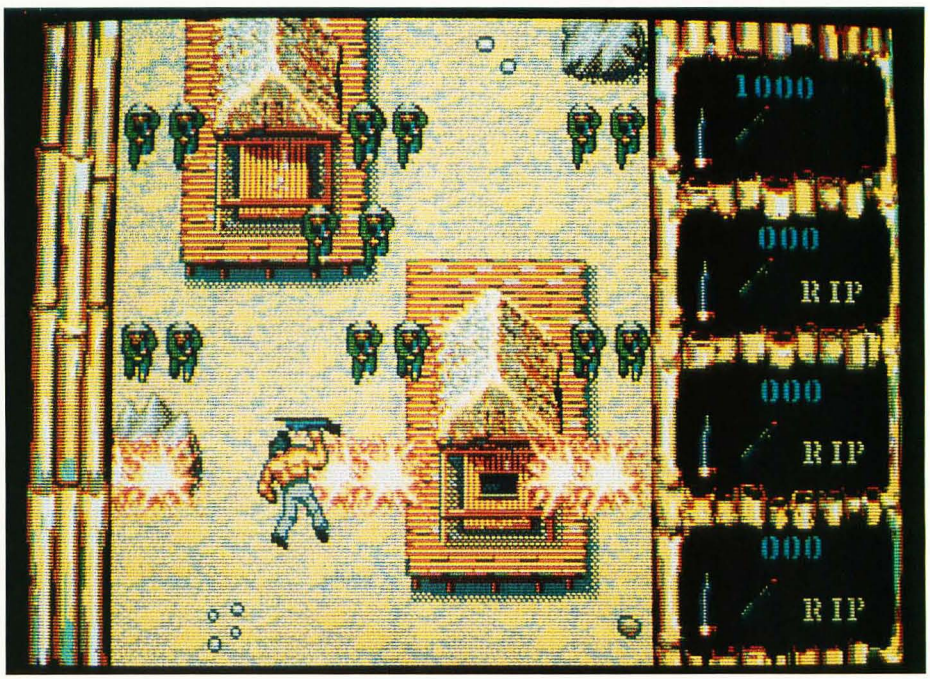
"Paladin is a hybrid of conventional war games and fantasy role-playing games. It's a game that requires thought and strategy."

Leatherneck
Microdeal
576 S. Telegraph
Pontiac, MI 48053
(313) 334-8729
Color only $39.95

You armchair Rambos are sure to get a kick out of this one. *Leatherneck* is arcade war action at its most frantic. As the manual states, "The object of the game is to score points by surviving and advancing as far as possible through the treacherous jungle." The rules are easy; surviving is the difficult part. The instant your fighter steps onto the screen, he will be attacked by wave after wave of enemy soldiers, each soldier firing a rifle or throwing grenades. You also have to make your way past large gun emplacements that fire with deadly accuracy.

Luckily the game can be played cooperatively by up to four players—and if you're planning on racking up a decent score, you'd better take advantage of this multi-player option. The first two players control their soldiers using joysticks plugged into the ST's joystick ports. To add players three and four to your battle force, you'll need to invest an additional *(to page 93)*

"The object of Leatherneck is to score points by surviving and advancing as far as possible through the treacherous jungle."



LEATHERNECK · from Microdeal

CRAZY CARS · CONTINUED

even keep up with its older competition.

You race your car (using the joystick or the arrow keys; either control method "feels" about the same) on a two-lane-wide road, accelerating, braking and steering your way past the other racers. If you manage to win the three stages of the race, you'll be awarded a snazzier car for your next competition. You start with a Mercedes 560. As you progress, you'll get to drive a Porsche 911, a Lamborghini Countach and a Ferrari GTO.

Unfortunately, Crazy Cars has an unfinished feel, as if the programmers got to the point where the game was just playable and then called it quits. For instance, unlike Pole Position, when your car crashes there is no explosion effect; you just slow down and lose some time. In fact, as far as animation goes, about the only "extra" is the way the car bounces when it goes over a bump at high speed.

The graphics are adequate, though not exceptional, and the sound effects are about what you would expect: a lot of squealing tires and shifting gears.

I don't know. Maybe I've just gotten tired of seeing the same old racing scheme over and over again. You'd think that, after all these years, someone would have come up with something new.

Recommendation: Get a demonstration before buying. ■

PALADIN · CONTINUED

schedule will tolerate. No, I say lucky because all those early projects (especially *Breach* which followed Universe II) led Mr. Carbone and Mr. Leslie to the creation of *Paladin*—and a fine creation it is.

Paladin is a hybrid of conventional war games and fantasy role-playing games. It's a game that requires thought and strategy—which means you get a break from battling frantically with a joystick. In addition, there are no time limits; the game operates on the "turns" principle, so the player can sick back and plan his moves leisurely.

Paladin is made up of a series of quests, each of which has a victory condition that must be met. The victory condition may be as simple as killing a certain percentage of the enemy and then getting all your men to the exit; or it may be much more complicated, requiring the completion of several tasks in order to win. Some quests lead immediately to others; while others are simple, one-quest games.

To play Paladin you move your characters around a game map, picking up items, exploring new territory and fighting the enemy. Each character is allowed a limited number of moves per turn, and each action the character enacts—moving, swinging a sword, picking up an item, etc.—uses up a certain number of those moves. When you've completed your turn, the enemy then makes its moves.

Unlike many war games, the characters and terrain in Paladin are not just simple icons. The graphics have been fleshed out and animated to add realism to the game. When you move a member of your party, he walks to the new location. When you attack with a sword, the weapon is actually swung at the enemy. Magical spells too have small bits of animation to keep the game interesting: fireballs explode, wiping out large areas of the terrain; and stun spells cause victims to glimmer momentarily.

▲

"Unlike many war games, the characters terrain in Paladin are not just simple icons. The graphics have been fleshed out and animated to add realism to the game."

▼

Winning a battle is always a treat; not only are you one step closer to fulfilling your victory condition, but you also get to hear a marvelous digitized death scream. Of course, you get to hear the scream when one of your own party members gets killed, too—not nearly as entertaining in that situation.

Your party is made up of various types of characters including a paladin (when you lose him, the game is over), swordsmen, rangers, thieves and mages. The enemy comes in various forms, as well: zombies, sorcerers, trolls, spirits and others, each with their own nasty way of inflicting damage on your poor group of adventurers.

Like most role-playing games, your party members have personal attributes, including vitality, health, encumbrance, accuracy of swordmanship and so on. Some of these attributes can be restored by resting or using potions; others require you to lighten your load or acquire special items hidden throughout the game.

Using the mouse to click on various icons along the bottom of the screen orders your characters to pick up objects, drop objects, open doors, climb stairs, cast spells and use items in various ways. These choices, in combination with the large number of objects available in Paladin, give the player plenty of moves to choose from and keeps the game challenging and fresh.

Paladin comes on two disks, one of which includes a number of quests for you to complete. Once you're familiar with the game (and that won't take too long), or have tired of the included quests, you can use the included quest builder program to create your own games. A map for a quest may consist of nothing more than a forest with a couple of rivers and pathways, or it may be as complicated as a large castle, filled with rooms and closets to explore.

Paladin is a habit-forming game. Once you get started you may find it difficult to turn off the computer and get back to the real world.

Recommendation: Buy it. ■

LEATHERNECK · CONTINUED

$14.95 to purchase Microdeal's four-player adapter.

Once you've flung yourself into combat, you have a choice of three different weapons: a heavy machine gun, which has long range, but slower firing; a light machine gun with shorter range but faster firing; and, of course, grenades.

The graphics are splendid and the action is hot—*very* hot. If you can survive for longer than two minutes on your first attempt, you'll be doing better than I did. Also, the first time you play, be prepared for some realistic sound effects. Each time a soldier dies (and due to this game's fast action, there's always someone with his face in the dirt), you'll hear a bloodcurdling scream that'll make your hair stand on end if you're not prepared for it.

Microdeal is certainly one of the most prolific ST game publishers (seems like every time I go out to the mailbox, there's another package from them), and Leatherneck's sizzling gameplay makes a good addition to their already large catalog.

Recommendation: For shoot 'em up fans. ■

DECEMBER 1988 · DISK LISTING:

The ST-LOG #26 diskette contains 9 magazine files. They are listed below.

FILENAME.EXT	FILE TYPE	COMMENTS
ASMLINE.ARC		
ASMLINE .S	ASSEMBLY	ASSEMBLY LINE
ASMLINE .PRG	RUN FILE	
DATEPLNR.ARC		
DATEPLNR.GFA	GFA BASIC	ST DATE PLANNER
GFABASRO.PRG	RUN FILE	GFA RUN-ONLY PROGRAM
FORTUNE.ARC		
MFORTUNE.GFA	GFA BASIC	MOUSE OF FORTUNE
MFORTUNE.PRG	RUN FILE	COMPILED VERSION
XFORMER1.ARC ¦		
XFORMER2.ARC ¦------->	ST XFORMER 2.1	
XFORMER3.ARC ¦		
UNARCHIV.DOC	TEXT	UNARCING INSTRUCTIONS
ARCX .PRG	RUN FILE	UNARCING PROGRAM
README .DOC	TEXT	DISK INSTRUSCTIONS

Disk instructions:
Only those files with PRG, TOS, or TTP extensions may be run from the GEM desktop. Other programs may require additional software as shown below. Due to space limitations, the files on this disk have been ARCed. Please read UNARCHIV.DOC for instructions on how to return the files to their unARCed state.

WARNING: Be sure to read the appropriate magazine article before attempting to run magazine files. Failure to do so may yield confusing results.

.EXT	DESCRIPTION
.BAS	Requires ST BASIC
.C	Requires C compiler
.PAS	Requires Pascal compiler
.S	Requires 68000 assembler
.GFA	Requires GFA BASIC or GFABASRO.PRG

INDEX TO ADVERTISERS

Atari Lifelines

by Clayton Walnum

Over the past year, we Atari owners have been much like blind men stumbling along the edge of a cliff. We don't know where we're going, we don't know how we'll get there, and we don't know when we might find ourselves falling into oblivion. We wonder what Atari is going to do next and speculate on how they will recover from their current slump. We wonder even if there *is* a slump. It's difficult to know whether the grim atmosphere of the Atari community is the result a few individuals' incessant griping—a matter of perception rather than reality—or whether Atari has truly betrayed our trust, has shaken out our wallets with a friendly grin then walked away covertly smirking, leaving us to listen to the I-told-you-so's of IBM, Commodore and Apple owners.

Atari products are continuing to disappear from store shelves. Major magazines are slashing Atari coverage from their pages. The lack of machines for the Christmas season has forced many once-loyal Atari dealers to bring other computer brands into their stores. We have to eat, they say. And to eat they must sell computers.

I suspect that the coming year will be a critical one for Atari, a year in which they will have to struggle not only to remain on the cutting edge of computer technology, but also to regain the respect and confidence of their customers and dealers.

Can they do it?

I don't know.

But I've got this feeling.

Probably this feeling's origin is in Atari's hints of dramatic announcements for the winter COMDEX. Atari representatives have indicated that there may be (coy, no?) some stunning new products represented at that show, and the rumors of Atari's reserving more than their usual square footage of display space seem to imply that *something* is waiting backstage ready to spring into the limelight. "Wait and see," Atari says, giggling and winking. "You're going to be surprised!"

As you've probably heard, the most likely of the surprises is an ST laptop. Atari, of course, has adherred to their new "closed-mouth" policy, so there's no official word, no photographs, no technical specifications. Just a rumor.

But whether an ST laptop will give Atari an "A" in the public's gradebook will depend to a great extent upon the price. Laptop computers, these days, are as ubiquitous as soiled diapers in a hospital nursery. Atari will have to differentiate its product from the others—and a low price may be the "feature" that will catch the public's eye.

A high-end laptop computer costs over $2,000 and includes such features as a built-in hard disk; long-lasting, rechargeable battery pack; online software (word processors and telecommunications programs are typical); a high-quality display; and, of course, lots of RAM. If Atari can come up with a high-end product at a low-end price, it may just get that "A," otherwise the ST laptop will be buried among its competitors.

Another product from the rumor mills (one that has in fact already been shown at computer shows around the world) is Atari's CD-ROM player. Just as the ST's MIDI compatibility attracted musicians by the score, so would the CD-ROM player draw the attention of schools, businesses and any type of organization that could take advantage of the huge storage space and fast data retrieval made possible by the CD technology.

CD-ROMs in the home? Many people scoff at that suggestion, saying that the average home can't take advantage of this new technology. The home CD-ROM market, they say, will be made up only of

computer "fanatics": people who will buy any computer-related gadget.

However, the fact is that, if Atari can bring the CD-ROM player to market at the price they'd like, the player along with the CD version of the Grolier's encyclopedia won't cost much more (maybe even less) than the encyclopedia alone in its conventional book form. Sounds like a bargain to me, especially when you consider that every significant word in the entire encyclopedia has been indexed so as to allow the almost instant location of any subject—and all this data on one thin CD.

Once CD-ROMs do get into the homes, can you even imagine the possibilities for game software? It almost staggers the imagination.

Yes, it'll take a while for CD-ROM to catch on, and there'll always be the skeptics. . .

But I've got this feeling.

Another possible "surprise" from Atari is a 68000-based game machine. What effect will this product have on the morale of the ST community? The answer to that is, I think, obvious—and strange. Because Atarians like to whine about Atari's game-machine image, it seems ironic that ST game software is usually at the top of the Atari best-seller lists.

Pssst. Listen up. We vote with our dollars, folks, and you can be sure that Atari, as well as all its third-party developers, have not overlooked the fact that more copies of *Dungeon Master* have been sold than any other piece of ST software. Couple that with the fact that the new age of video games is upon us (although many claim the new age will fade just as quickly as the old one did), and a state-of-the-art video-game system from Atari shouldn't come as a surprise to anyone. *Someone* needs to give Nintendo serious competition. And who else should that be but the deposed king of the video game, now back to reclaim its crown?

There is, ready and waiting to be burned into cartridges, a wealth of high-quality games for the ST. Of course, games like Dungeon Master will not, due to their large size, be able to make the transition from disk to cartridge. And you can't (even though Atari has tried it with the XE Game System) have a disk drive with a game machine. That makes the system a computer, and the average guy looking for a Christmas present for his kids is not going to buy a game machine unless it looks like a game machine.

Imagine that. People not buying Ataris because they make them think of computers. What a world.

That there is currently a good market for video-game machines there can be no doubt. Is the release, then, of a new, state-of-the-art game system a good move for Atari? The answer to that depends upon the longevity of the current video-game craze. Every company needs to look to the long-term. (Ask Coleco; they'll tell you.) If there is no future in video games, then perhaps Atari should focus its energies elsewhere.

But that's a big "if." We have to remember that game technology has improved immensely over the past few years. During the old video-game craze, people were buying game machines in an effort to bring the coin arcades into their home. However, machines like Atari's 2600, the top-selling game system at the time, looked pretty pathetic compared with the high-tech wonders that were sucking down quarters at the local arcade. People bought disappointment for their bucks and the industry suffered for it.

Today's video-game machines truly rival their arcade counterparts. If you haven't seen a Nintendo in action, you owe it to yourself to take a look. These are sophisticated graphics powerhouses, and the games are every bit as addicting and exciting as their coin-operated cousins.

The new game cartridges sport 128K of ROM—plenty of room for complex, long-lasting games. And greater game complexity has given rise to yet other advances. Many of the new video-game cartridges allow players to save their progress in a game at the end of a game-playing session, so they don't have to start over the next time they sit down to play. Some of the cartridges accomplish this with a built-in battery pack. Others use a clever password system: When the player has finished a session of gameplay, the values of the game's "variables" (the player's position, the items he may be carrying, his scores, etc.) are converted into a coded string of characters. When the player comes back to play again, all he has to do is enter this string of characters, and he's back where he left off.

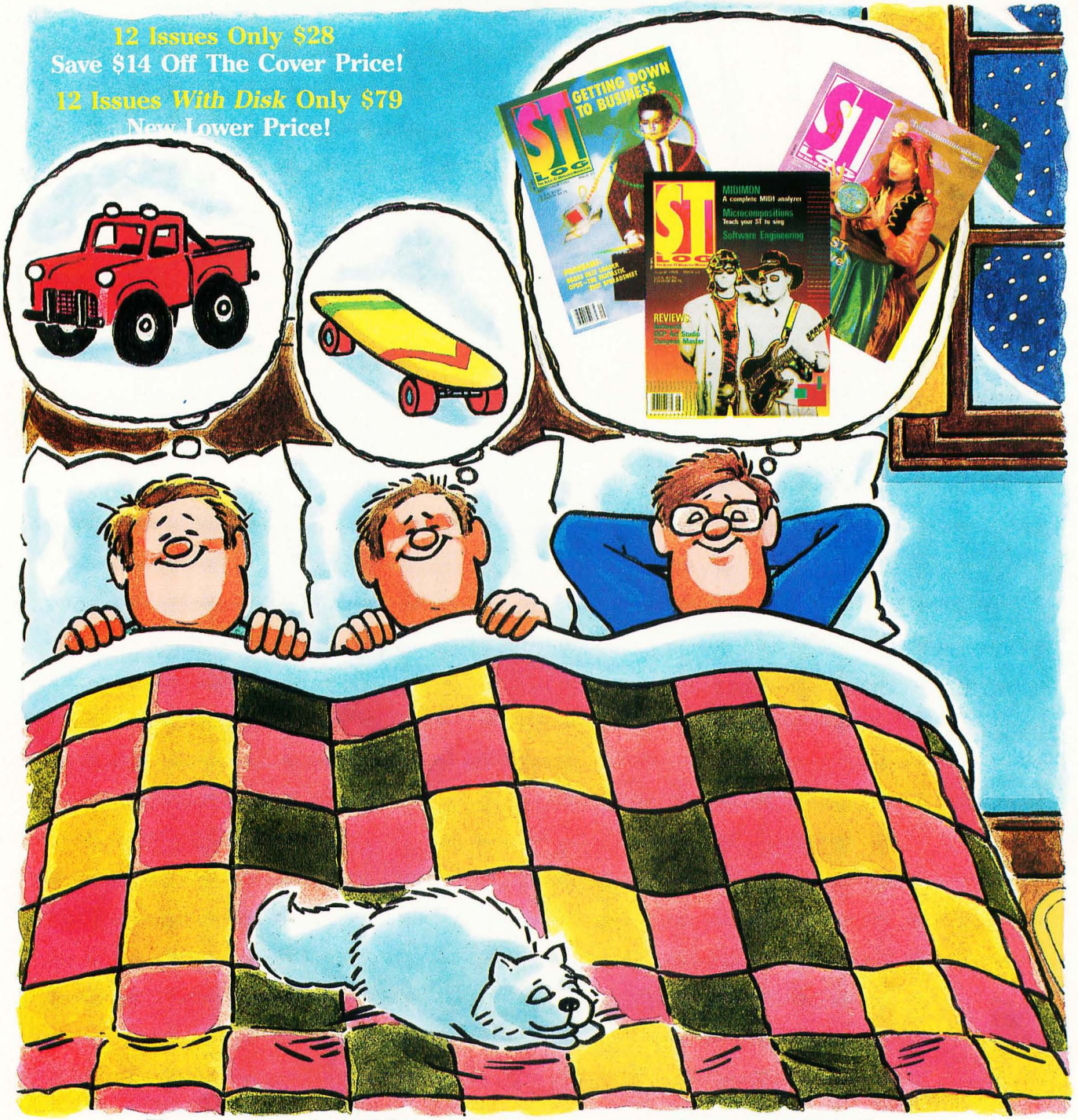
All of these advances—improved graphics, more complex games, game-saving features (and others)—mean that when you buy a video-game system today, you're not buying disappointment. That goes a long way toward assuring a good future for video-gamers.

Will an ST laptop, a CD-ROM player and a 6800-based game system be Atari's life-lines for 1989? Do these products even exist?

No one outside of Atari knows for sure. But I've got this feeling.■

MAY ALL YOUR CHRISTMAS DREAMS COME TRUE!

SUBSCRIBE TO ST-LOG AND SAVE!



The world of ATARI-ST continues to grow by leaps and bounds and ST-LOG is there every step of the way! SUBSCRIBE and SAVE today!

☐ **12 Issues $28**
MCLWW

☐ **12 Issues with Disk $79**
DCLWW

MAKE CHECK PAYABLE TO L.F.P., INC.,
P.O. Box 16928, N. Hollywood, CA 91615.
Offer expires March 1, 1989.

☐ PAYMENT ENCLOSED ☐ BILL ME
CHARGE MY: ☐ VISA ☐ MASTER CARD

CARD # EXP SIGNATURE

NAME

ADDRESS

CITY STATE ZIP

REVOLUTIONARY NEW PRODUCT

SWITCH BACK

REQUIRES at least 1 meg. of RAM
(or a Megadisk or Polydisk Cartridge)

- Imagine Saving almost any game at any point, then being able to return there as many times as you like.
- Imagine the Ultimate Back-up Utility that actually UNPROTECTS programs as it copies them. Lets protected programs be stored as files, run from a hard disk or even be transmitted over a modem.
- Imagine saving three or more protected single sided disks on just one double sided disk.
- Imagine Instantly switching back and forth between two different programs, games, utilities or business applications.

Now Stop Imagining and get Switch/Back. It can do all this and more.

Switch/Back is a revolutionary new hardware and software package that lets you get more from your ST, MUCH MORE.

Switch/Backs gaming features lets you instantly save most games then continue playing. If you get in trouble you can switch back to where you were as many times as you like.

BACK-UPS –Switch/Back can work with your favorite back-up program and allow you to save whole protected disks to files for archival purposes. It can also automatically unprotect a program and save it as standard file. This method works on hundreds of ST programs and it allows you to run the files directly. Its perfect for running protected programs off a hard disk. It creates standard TOS files, that can be stored together on disks or even transferred by modem.

SWAP – Switch back lets you load just about any two programs into your ST and switch instantly between them. It works with games, business programs, utilities, compilers, etc. Although only one program is running at a time, the other is available instantly, right where you left off.

The Switch/Back hardware plugs into your printer port for easy use (It has a pass through connection for your printer too.)

Switch/Back requires at least One Meg of memory (Or a Polydisk or Megadisk) **ONLY $69.95**

ST Protection Techniques

Finally ST Copy protection techniques are revealed. This complete book and disk package details the state of the art in ST Protection methods and much, much more.

The Software included with the book provides many powerful features like the AUTOMATIC PROGRAM PROTECTOR. This easy to use Utility allows you to protect just about any ST program. You can choose a combination of protection methods like encryption, checking custom disk formats, password protection or a limited use option that makes the program self-destruct after running a preset number of times.

The book includes topics such as Phreaking, Logic Bombs, Hardware data keys, the legal aspects of piracy and software protection, Custom disk formats, Pirate Bulletin boards and much more.

In addition it contains reviews of the popular ST back-up programs and detailed explanations of ST disks and drives.

ST Protection Techniques (Book and disk package) **Only $39.95**

The worlds most inexpensive clock cartridge. Finally its affordable to keep your time and date accurate. 3 year battery included. **ONLY $24.95**

WATCHCART ST

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)

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.

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**

DIGIPLAYER The High powered digisound software can now be obtained by those who already own a digitizer for the ST. Compatible all cartridge based digitizers. Extend the power of your digitizer with Digiplayer.

Only $49.95

ALPHA SYSTEMS

24 HOUR HOTLINE – VISA & MasterCard Welcome

216-374-7469

Customer Service line (216) 467-5665. Call or write for free catalog.

Order by phone or send check or money order to:
ALPHA SYSTEMS 1012 Skyland, Macedonia, OH 44056
Include $3.00 shp. & hdlg. (US & Canada). Ohio residents add 5½% sales tax. Foreign orders add $8.00

CIRCLE #127 ON READER SERVICE CARD.

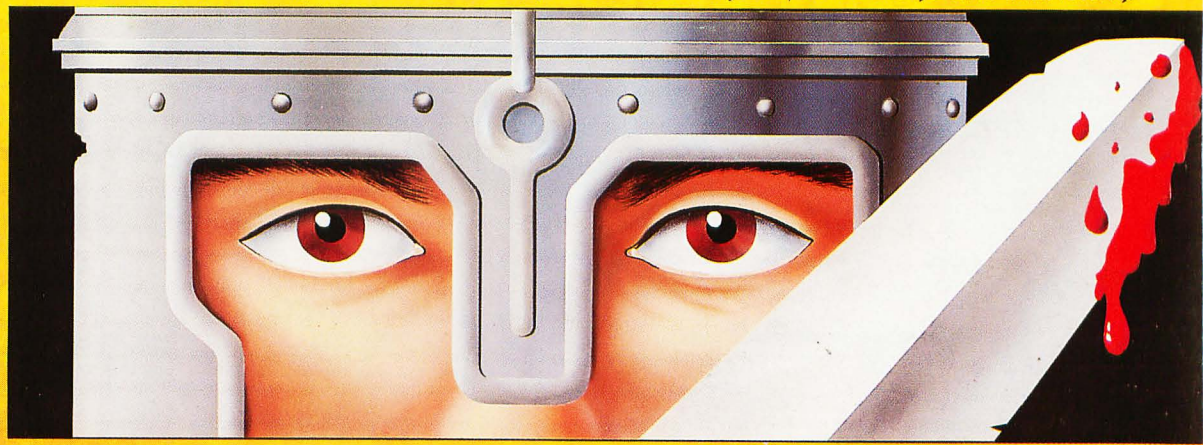
OMNITREND'S

PALADIN

In the Days of Legends, there was a young Paladin named Brandon. Brandon had Heard that the King of a Far Kingdom was Bestowing Knighthood upon those Souls who Proved themselves Worthy of the Title. Resolute upon the Goal of Knighthood, Brandon gathered a Fine Band of Nine to accompany him. He took Mages, with their Arcane Magics and Strange Potions.  He called upon Swordsmen,  with their Sharp Blades. With him came Rangers and even a Thief who had Reformed His Ways. Their journey took them across Fierce Deserts and Swift Rivers,  through Steamy Jungles and into Caverns  that Descended to the Very Heart of the Earth. On their Long and Perilous Journey, they Did Battle Nasty Trolls  who carried Great Axes. Undead Zombies  Plagued them and Dragons  Burned Their Hides. Sorcerers  Exchanged Orbs of Fire with their Wise Mage and Cast Mind Stuns on the Unwary.

All of this happened with **Fine Animation** and the Battles Did Ring with **Thrilling Digitized Sound** (excluding IBM version). They found that the **Program Includes Ten Challenging Quests.** They also discovered that an additional **Quest Disk with 16 Quests Is Available.** After a Time, when they wished to Change Their Fate, there was a **Quest Builder Program** which allowed them to Modify Existing Quests and Create New Ones to Please their Friend and Confound their Enemies. Even Valiant Warriors from the Far Future were not forgotten, for they could **Import Breach Squad Leaders as Paladins.**

Omnitrend's Paladin is available for the **Amiga** and the **Atari ST.** Versions for the **IBM PC** and **Macintosh** are coming soon. For those who wish to Order Directly (Visa/MC/COD) phone (203) 658-6917. For mail orders send Check, Money Order, Gold Pieces, or Credit Card Information to Omnitrend Software, Inc., PO Box 733, West Simsbury, CT 06092. Paladin is $39.95, the Quest disk is $24.95. Please add $3.00 for Postage and Handling.



OMNITREND SOFTWARE
P.O. BOX 733 - WEST SIMSBURY, CT 06092

CIRCLE #128 ON READER SERVICE CARD.