

ST Log

THE ATARI ST MONTHLY MAGAZINE

Telecommunications
Issue

JULY 1988
ISSUE 21

U.S.A. \$3.50
CANADA \$4.75



Moonlord ST
Frankendrive
Battle Blips



PALADIN



N 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

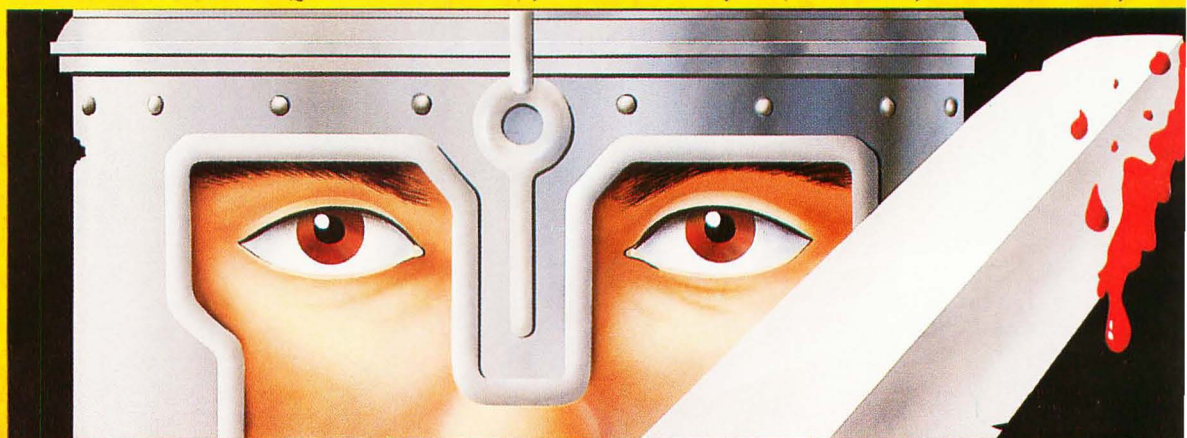


Exchanged Orbs

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 Friends 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.



EDITORIAL



While everyone was freaking out, I heard screams of "Quick, get me a fire extinguisher!" and "Get this on film!"

BOOM!

That's what it sounded like. Actually there was a nice shower of sparks and so much smoke it sent the photographer and staff running for cover. What I'm describing here is my SC1224 color monitor picking a most inopportune time to not only go on the fritz, but to do it in true 50's science fiction movie style. But let me regress a little here.

This is the city, Los Angeles, California. Over 465 square miles of constantly interfacing humanity representing computer users of every creed. . . . It was the morning of April 21st. I was busy in my downtown office when I received a call. They needed my ST to shoot the crystal ball inset for the cover of this issue. I didn't have the time to head down to Culver City, the location of the studio, so I asked ST-Log's art director Ed Herch whether he and the photographer could come in and take the photos at the Beverly Hills location.

After attempting several trial shots, the lighting just couldn't be adjusted properly, so we headed over to the photo studio. Oh, I forgot to mention, it was one of those rare days in Southern California when it was raining.

We arrived at the studio with my 1040 and monitor. While our photographer, Ladi Von Jansky, was positioning the camera and lighting, I proceeded to set up my equipment. But. . . when I pressed the power button on my monitor, it was Fourth of July time! While everyone was freaking out, I heard screams of "Quick, get a fire extinguisher!" and "Get this on film!"

So then someone asked if it's supposed to do that. My response was, "What do you think this is, a PC clone?"

Another recommended I try it again. What the hell. I turned it on again. This time just some crackling and buzzing sounds. So what happened? Did the rain get at it? (It wasn't raining when I carried it into the studio.) Was it plugged accidentally into 220V? (We checked, and it wasn't.) Was it the ride down in my rental car through the trials and tribulations of LA traffic? (Could be!)

Well, I had a couple of spare monitors. Trouble was, they were home. So someone asked "Why not go home and get one. . . ?" Fine. No problem. "I live in Massachusetts!" was my answer to that. "Fine! We'll wait for ya." Well, I called The Federated Group, which is run by Jack Tremiel's son, Gary. As you may recall, this chain numbers nearly 70 stores, and is fairly big throughout the west and parts of the midwest. They carry consumer electronics products such as televisions, audio components and computers. The chain was purchased by Atari last fall.

Gary was only too happy to help, and set me up with a monitor from their nearby Federated store on Olympic and Bundy (being new to LA, I knew where Olympic was—I had gotten lost around there a couple times. . . but never heard of Bundy.) ST-Log Art Director Ed Herch and I showed up at the store, with the manager expecting us. He quickly loaned us a store demo 1224, and we went on our way.

We set the monitor up, turned it on (carefully!) and shot the photo. All that time and effort—just for that little crystal ball on the cover! Gee. . . I can hardly wait 'till next month.

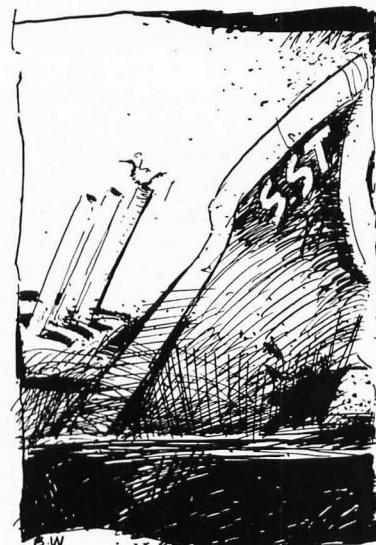
—Lee H. Pappas
Publisher/ST-Log



DIALXR...12



Frankendrive...30



Battle Blips...55



FEATURES

- DIALXR**.....Maloney 12
Use your ST's VT-100 emulator to repeat-dial a busy BBS, while you go back to your word processor, spreadsheet, or any other GEM application program.
- Moonlord ST**.....Clayton Walnum 18
This popular 8-bit game has now been translated to the ST. A commercial quality game, Moonlord includes stunning graphics by Maurice Molyneaux.
- The Software Evolution**.....Karl E. Wieggers 23
A close look at software engineering by a software professional.
- Frankendrive**.....Andy Eddy 30
Build your own hard disk drive? Yes, Igor! Andy tells of his hard disk construction experience, offering many tips on assembly and on locating the parts.
- Busy Buddy**.....Matthew J.W. Ratcliff 35
Don't let that BBS time-out on you. Keep your connection secure with this clever deck accessory.
- Read Any Good Docs?**.....David Coles 46
Program manuals have long been criticized. If you're planning on releasing a program in the future, these tips will help you create professional quality documentation.
- Anatomy of a Simulation**.....Bob Curtin 52
Flight Simulator II for the ST is an experience in itself. But can you get even more out of it?
- Battle Blips**.....Patric Dell'Era 55
The ever-famous game of Battleship is brought to a desk accessory that allows you to battle the computer or a human opponent via a modem.
- Errors in Abacus**.....Charles F. Johnson 72
The ST library from Abacus gets an updating by ST-Log's assembly language wizard, Charles Johnson.

REVIEWS

- Cyber Studio (The Catalog)**.....Charles F. Johnson 87
and Andy Eddy
- Microbyte disk drive (Paradox Enterprises)**.....Matthew J.W. Ratcliff 90
- Astra HD+ (Astra Systems)**.....David Plotkin 92
- Tacklebox (SRM Enterprises)**.....David Plotkin 93
- Bureaucracy (Infocom)**.....Andy Eddy 95
- Circuit Maker (Illiad Software)**.....Frank Cohen 95
- SupraModem 2400 (Supra Corp.)**.....Andy Eddy 97
- Buzzword (Buzzword Game Company)**.....Steve Panak 98

COLUMNS

- Editorial**.....Lee Pappas 3
- Reader Comment**.....6
- ST News**.....ST Gossip from Hollywood.....TG 17
- Database Delphi**.....Andy Eddy 76
- Step I**.....Maurice Molyneaux 79
- Ian's Quest**.....Ian Chadwick 84

MOVING?

DON'T MISS A SINGLE ISSUE

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

DO YOU HAVE A QUESTION ABOUT YOUR SUBSCRIPTION?

Check the appropriate boxes below:

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

P.O. BOX 16928, N. HOLLYWOOD, CA 91615

Name _____

Street Address _____

City _____ State _____ Zip _____

ATTACH LABEL HERE

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

ST-Log Staff

Publisher: Lee H. Pappas. *Executive Editor:* Clayton Walnum. *Art Director:* Eddy Herch. *Managing Editor:* Dean Brierly. *East Coast Editor:* Arthur Leyenberger. *Midwest Editor:* Matthew J.W. Ratcliff. *West Coast Editor:* Charles F. Johnson. *Contributing Editors:* Michael Banks, Ian Chadwick, Andy Eddy, Arnie Katz, Bill Kunkel, Maurice Molyneaux, Steve Panak, Douglas Weir, Joyce Worley. *Copy Chief:* Katrina Veit. *Copy Editors:* Anne Denbok, Sara Bellum, Pat Romero. *Typographers:* Judy Villanueva, David Buchanan, Klarissa Curtis. *Contributors:* Tom Castle, Frank Cohen, Betty D. DeMunn, Michael Donahue, Colin Faller, Kevin Kennedy, E.H. Wysocki. *Production Director:* Donna Hahner. *Production Assistant:* Steve Hopkins. *Advertising Production Director:* Janice Rosenblum. *Subscriptions Director:* Irene Gradstein. *Vice-President, Sales:* James Gustafson.

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

ST-LOG magazine (L.F.P., Inc.) is in no way affiliated with Atari. Atari is a trademark of Atari Corp.

Where to write

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, with the name of the column included in the address. We cannot reply to all letters in these pages, so if you would like an answer, please enclose a self-addressed, stamped envelope.

Advertising Sales

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: Janice Rosenblum — Advertising Production, **ST-LOG**, 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210.

Permissions

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

Due, however, to numerous requests from Atari club libraries and bulletin board systems, our policy does allow club libraries to individually-run BBSs to make certain programs from **ST-LOG** available during the month printed on that issue's cover. For example, software from the January issue can be made available January 1.

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

In addition, any programs used must state that they are taken from **ST-LOG** magazine. For further information, contact **ST-LOG** at (617) 797-4436.

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.

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 to **ST-LOG**, P.O. Box 1413-MO, Manchester, CT 06040-1413.

READER COMMENT

The Great ST Split

I'm a subscriber to Delphi, and I spend the majority of my on-line time in the ANALOG Atari SIG. It's been many months since the possibility of ST users having their own SIG was discussed, and still nothing has been done. The databases in the ANALOG SIG are greatly overcrowded due to the fact that programs for both 8- and 16-bit machines are vying for space there, and when I go to read messages in the forum, I don't want to be bothered with any messages except those that relate to my machine.

Certainly there are enough ST owners around to support their own SIG. (Judging by the messages in the forum, 90% are ST users already.) What's the holdup? We ST owners demand equal representation on Delphi!

—Mark Penfield
Atlanta, Georgia

It's already been done! ST users now have their own SIG on Delphi, sponsored of course by ST-Log Magazine. At the time of this writing, you could access the new ST SIG by entering the old ANALOG SIG and typing ST at the ANALOG > prompt. By the time you read this, though, the SIG may have gotten its own entry on the main system menu. (There was another entry there that started with "ST," and it's Delphi's policy to keep the menu entries as unambiguous as possible.)

The new ST SIG functions exactly the same as the ANALOG SIG, so you won't have to learn anything new to take advantage of its services. Though all the

What's the holdup? We ST owners demand equal representation on Delphi!

ST related files have been moved to the new SIG, there are some new database sections that weren't available in the ANALOG SIG (for instance, Desktop Publishing); so you may have to spend a little extra time figuring out what files got moved where. But in the long run having the extra database sections will greatly reduce the time it takes to locate a particular file, since the database sections are now more specific. See you on-line!

The Color of Things to Come

I have a 520ST with a monochrome monitor. I have a friend who has only a color monitor. We both read *ST-Log* regularly and both have a very similar complaint. Many of your programs run on either color systems or monochrome systems, but not both. It'd be nice if you could only publish programs that run properly on both

systems. That way, nobody has to miss out on a great program. For instance, I have to go over to my friend's house if I want to play "Crin's Castle" or "Strathello" from your May issue.

—Tom Reddy
Alton, Illinois

It's not untypical for game software to run only on color systems, since color can be very important to game graphics. And since the May issue had an adventure theme, it concentrated on games.

*However, there's more to consider than what type of software a program is. A magazine like *ST-Log* relies on its freelance contributors for the bulk of the material published. Frequently, these contributors do not own both a color and monochrome system, so can't develop their programs to work on both. *ST-Log's* programmers simply don't have the time to rewrite each program we choose to publish so that it's compatible with all systems, and we don't like to turn away a good program simply because it's limited to one type of system. We try very hard to please everybody—and it is certainly true that a program that runs on all systems has a better chance of being accepted for publication than one that has a more limited audience—but sometimes we must rely on our judgement and publish programs that we feel are exceptional enough to merit attention, even if they won't be usable by every reader.*

All the programs in the April issue ran on both color and monochrome systems. The June issue contained two programs for all systems (ST Font Printer and Mouse-ka-source) and one each for color or monochrome only (Decimal Destroyer and BASIC Draw, respectively). This issue contains only one program, Moonlord ST, that is limited to one system type.

The Support Dilemma

I am the owner of both an IBM PC and a 520ST, and I am continually amazed at the differences in the service offered by companies who provide software or hardware for these machines. I'm not even talking about factory service for the machines, which IBM has built its reputation upon. I'm just talking about a simple phone call to a software vendor. I've never had any trouble getting help with the products I've purchased for my IBM. If I have trouble with a software package, in most cases a quick phone call will

In most cases, a telephone call to the software vendor will result in a conversation with someone who hasn't the vaguest idea.

bring me a solution.

Not so with ST software. In most cases, a telephone call to the software vendor will result in a conversation with someone who hasn't even the vaguest idea of what he's doing. Letters usually go unanswered. I've discovered that when one buys a piece of ST software, he's almost certainly on his own. There have been exceptions: most notably WordPerfect Corporation's incredible customer-service department. These people stay in touch with their users. You don't have to beg for upgrade information or answers to questions.

I don't see why all providers of ST software can't treat their customers with the same kind of support WordPerfect offers. It seems to be that it would pay off in the long run.

—Anthony Pellman
Spokane, Washington

You're not alone in your complaints, and though there is no good excuse for a company to ignore its customers, there are some reasons why you experience better product support for IBM programs than for ST ones.

It all comes down to numbers. Customer support is an expensive and time-consuming task. Many publishers of ST software have only a few people on their staff, each person having to wear many different "hats" during the course of a day. This means that the staff's time is spread very thinly over a variety of tasks that must be completed to run a company. Since ST software doesn't sell anywhere near the number of packages an equivalent IBM product would, these ST software companies frequently can't afford a larger staff.

Your experience with WordPerfect Corporation proves the point. WordPerfect is a very large company that al-

ready had a good customer-support system in place before they released their word processor for the ST. For that reason, ST users get the benefit of a customer-support department that is, at this point, really being paid for by the users of other machines. The more popular the ST becomes, the more software will be sold, and the larger and more organized the software companies will become. Until then ST software purchasers will undoubtedly notice a lesser quality of customer support.

However, as we stated above, there is no good excuse for a software company to ignore its customers. ST software suppliers should note that their customers want and need their help. If anybody reading this is a software supplier not willing to support his product, ST-Log suggests you find a new line of work.

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



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

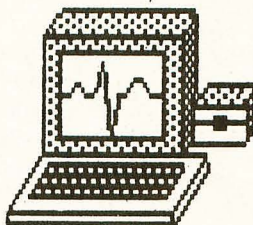
P.O. BOX 665, CHEPACHET, RHODE ISLAND 02814 USA
(401) 568-8459

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 #102 ON READER SERVICE CARD.



An incredible simulation

Cardiac Arrest!

with binder and manual, \$69.
See discounted package price.

Cardiac Arrest! is a unique product. In this mathematically-based simulator, you interpret the history, on-screen EKG, lab data, and vital signs, then give treatment orders in plain English. While many computer users enjoy Cardiac Arrest! as a challenging medical adventure game, it's a sophisticated product used world-wide for ACLS (Advanced Cardiac Life Support) education. IBM, Apple II +/c/e, Atari ST, Atari XL/E.

Antic: "impressive and amazingly complete"

ST World: "both highly educational and fun to play"

We support our products. Updates will be available to users for \$6 each when ACLS recommendations change. Our software is NOT copy-protected.

Cardiac Arrest!	\$69
ACLS Protocols	\$29
EKG Teaching	\$29
CardioQuiz	\$19
Blood Gases	\$24
QuizPlus	\$29
Demo	\$7

Ask about the four-disk ACLS Package (includes Cardiac Arrest!) for \$109. Order direct!

Mad Scientist Software

2063 N. 820 W., Pleasant Grove, UT 84062

Visa/MC orders call 801-785-3028

CIRCLE #103 ON READER SERVICE CARD.

ST NEWS



UPI

ST Babylon

Did you ever think that Hollywood stars such as Bob Hope, Sophia Loren, and Frank Sinatra use Atari ST computers? Well, Merrill Ward Software has indirectly brought these stars to the ST with their new product, *The Celebrity Cookbook*. The product is nicely packaged in what would look like a small cookbook and includes over 300 recipes that originated by the likes of Shirley Maclaine, Ronald Reagan and others.

After you get over the "hype" of the packaging, you will find a pretty decent assortment of recipes that can be made in just about any kitchen. Instead of favoring the more technically difficult dishes, *The Celebrity Cookbook* offers a common-sense approach to making breakfast, lunch and dinner.

The Celebrity Cookbook comes in six separate volumes. Each volume covers a different topic: Holiday cooking, weight-watching meals, nutrition, buying food, etc. The program runs on "any" ST, although we only tested it on a 520ST.

Merrill Ward Software

255 N. El Cielo Road, Suite 222
Palm Springs, CA 92262
(619) 320-5828

Atari Fun For Little Tots!

A new series of educational programs is being produced by Alohafonts Software. Uncle D's *Con-Sound-Tration* game tests and enhances memory concentration techniques for children four and up. The product comes with two disks, a general program disk and a data disk. Once the program has been loaded, everything is icon driven, so children will find it easy to work with once they get used to manipulating the mouse.

Uncle D is a cute character who appears to indicate the success or failure of a child to solve one of the puzzles. In play, the child selects a square from the screen and is shown a drawing of an everyday object. The child then flips a square from the right side of the screen and hears a digitized recording of the object's sound. If the child matches the drawing to the sound, the objects disappear and a happy face of Uncle D appears. Otherwise, the squares are flipped back over and the game continues.

Con-Sound-Tration has a list price of only \$24.95 and comes with four sets of games. Also available are several data disks that explore the environment,



Software Toolworks

One Toolworks Plaza
13557 Ventura Blvd
Sherman Oaks, CA 91423
(818) 907-6789

Datasoft Adds Some Fun

Datasoft has released three new games for the ST: Battledroidz, Global Commander, and The Rubicon Alliance. The games feature good graphics and animation and all are priced under \$30.

BattleDroidz is a combination of Marble Madness and Tron, two popular arcade games. A joystick is used to move your character around a three-dimensionally drawn screen. Hills, ramps, canyons and cliffs make up the playfield that is inhabited with Cyborites, little pill-shaped creatures that try to zap you. A game-grid map shows your movement through the playfield until you complete each level. If you were disappointed by the ST version of Marble Madness, you might find this one a challenge.

In *Global Commander* you must carefully negotiate peace between the Earth's United Nuclear Nations (UNN). The program is icon driven and gives you information about the UNN economies, resources, food supplies, etc. There are 16 UNN world powers that are nuclear-armed and aggressive. The goal of the game is to avoid nuclear war and save the world.

The *Rubicon Alliance* is a shoot-'em-up game where you pilot a starship that destroys just about everything in sight. Your home planet's defense shield has been breached and you must fight off the invaders until you find the protagonist of this game, Nono. Eight missions must be completed before you can attach Nono and save your planet.

Datasoft (Intellicreations)

19808 Nordhoff Place
Chatsworth, CA 91311
(818) 886-5922

What is SQL?

The Structured Query Language (SQL) is a new database query system that is just now making significant inroads into the mass business market. Microsoft, Apple, Lotus, Borland, Ashton Tate, and many other top software companies have announced SQL products for the IBM PC and Macintosh. SQL is now the accepted standard American database query language.

The SQL system has been available for the ST for the past year in Regent Base from Regent Software. The ST almost had a second SQL database when Oracle (a huge software company that specializes in databases) was considering porting its powerful SQL database onto the ST to run under Whitesmith's Unix system. Unfortunately, Oracle changed its mind.

Regent Software has now released the Regent Base Guide Book, a 100-page introduction to the SQL system as implemented on the ST in Regent Base. The Guide Book also comes with a demonstration version of Regent Base and all of the tutorial examples. At \$24.95, the Guide Book is a good introduction to the SQL language and overall database design principles.

Regent Software

7131 Owensmouth #45A
Canoga Park, CA 91303
(818) 882-2800

Delphi Makes Changes

ST/Log's official national bulletin board system can be found on Delphi. Since *ST/Log* used to be published as part of *ANALOG*, the *ANALOG* forum has been used to keep 8-Bit and ST information, news, and downloads. Unfortunately, this has made the one forum very crowded with information of both systems.

Delphi has now enhanced its system by adding a new ST Forum. Now you will find each month's *ST/Log* downloads in the new forum, as well as all the news, information and questions that used to be found in the *ANALOG* forum. To access the new forum, type GROUPS ST from the main prompt.

ST Xformer Enhancements

For those of you that are using Darek Mihocka's 8-bit emulator to run Atari XL/XE programs on your ST, Darek is working on a new version. The new version will include new features such as Player-Missile graphics support and a modem (R:) device handler. The more startling news is that the new version will also emulate an Apple II and Commodore 64.

If all this turns out to be true, the ST will be able to run IBM-PC software under PC-Ditto, Macintosh software under the Magic Sac, Apple II, Commodore 64, and Atari XL software under the Xformer. This just goes to show how versatile the ST really is.

Spanish / French / German languages, and more.

AlohaFonts

PO Box 2661
Fair Oaks, CA 95628-2661

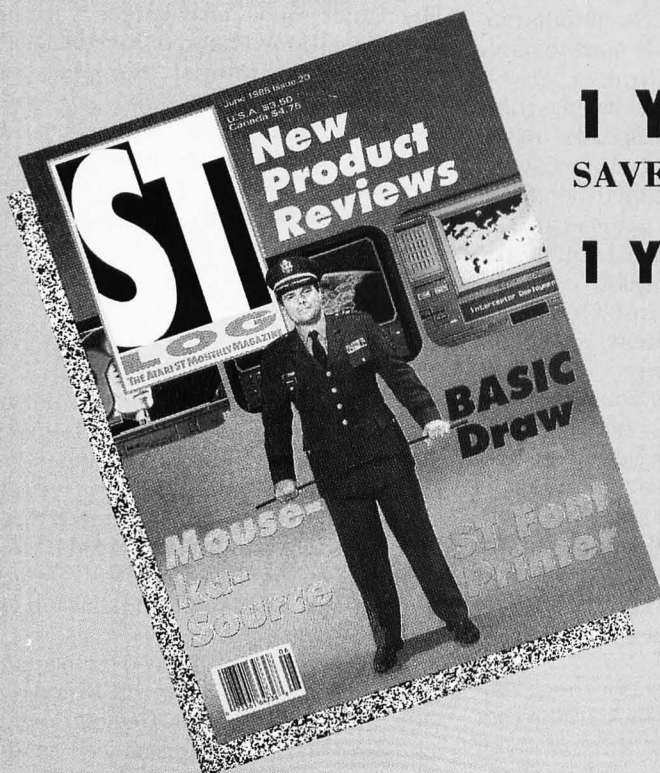
Typing With Mavis Who?

Mavis Beacon is just like the woman that might have tried to teach you typing skills when you were in high school. She is a good teacher, she puts on the pressure if you don't work hard enough and rewards you with some fun if you do well. This might not sound like your typical typing-tutor program, but *Mavis Beacon Teaches Typing* isn't your normal teacher.

Split between your normal classroom typing sessions and an interesting road racing game, Mavis has three levels of teaching: Never seen a typewriter, Somewhat Good, and Expert. Once selected, Mavis adjusts the difficulty according to how you progress through the 50 or so lessons. The program is completely icon/menu driven, so it is suitable for children learning how to type.

Mavis Beacon Teaches Typing is nicely packaged with a complete user manual and supplement for the Atari ST.

BOOT UP TO BIG SAVINGS!



1 YEAR FOR ONLY \$28

SAVE \$14 OFF THE COVER PRICE

1 YEAR WITH DISK ONLY \$105

SAVE TIME AND MONEY SUBSCRIBE TO ST-LOG

SAVE \$14 OFF THE
COVER PRICE WITH
THE CONVENIENCE
OF HAVING ST-LOG
DELIVERED DIRECT-
LY 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!

MCGWW

FOREIGN — ADD \$7 PER YEAR

☐ 1 YEAR WITH DISK @ \$105

DCGWW

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. Offer expires August 31, 1988. Your first issue will arrive in 6 to 8 weeks.

WATCH FOR IT!

HARDWARE ADD-ON's for ATARI ST

MEMORY upgrades:

Solderless "plug in" installation, 1 year warranty
520ST – expand to 1, 2.5 or 4 MB on ONE board
– prices start at \$129 for the 0K version – or go to
1 MB only, \$79 – without ram.

1040/520STfm – upgrade to 2.5 and 4 MB, fully socketed. OK board \$149.

For all our memory upgrades: on board CLOCK module only \$30 incl. software!

For more detailed information phone or write to:

tech-specialities Co.
1022 Hodgkins, Houston, TX 77032
☎ (713) 590-2068 and 590-3738

We ship COD or prepaid, sorry no credit cards!

S/H on memory upgrades	\$5
------------------------	-----

on Hard Drive Kits – \$10/no drive – \$20/w. dr.

Texas residents add 8% state sales tax.

CIRCLE #104 ON READER SERVICE CARD.

EXPANDABLE Hard Drive Kits:

1. 10" x 6.5" x 15" with full SCSI interface – 150 W
PC power supply with fan – room for up to 5 half
height drives – mounts on floor, under desk or on
desktop – can supply power to 520ST and disk
drives.

kit with 40 MB full height 30 ms drive	\$745
--	-------

2. MEGA footprint, 3.7" high, 1-port SCSI Interface
- room and power for 3 half height or 1 each full
and half height drive, comes with fan

kit with 10 MB half height drive	\$395
----------------------------------	-------

3. 4.5" wide x 6" high x 13" deep with 1 port SCSI host adapter – ready for 2 half height or 1 full height – 55 W power supply, fan optional

kit with 20 MB 1/2 height	\$485
---------------------------	-------

Atari 520ST, 1040St, 520STfm and MEGA are trademarks of ATARI Corp.

WHAT CAN **CRICIT**TM DO FOR YOUR BUSINESS?

CRICIT is an integrated **Cash Register & Inventory Control** package so complete that you can give your old cash register notice! CRICIT ties together many of the facets of running your business. **Here's what CRICIT can do for you:**

- **Complete cash register functions**
- **Flexible inventory control** for 65,500 products
- **Daily, period and yearly reporting**
- **Price/product labels** with optional bar code
- **Coupon issue and redemption**(fixed or % of sale)
- **Customized receipts, coupons, inventory and reporting**
- **Ready-to-mail purchase orders** with automatic re-order calculation
- **Commission calculation** for 15 sales clerks
- **Mailing lists** in list and label formats
- **Lay-aways, auto-discounts, stock searches**
- **Between-store reporting** via modem
- **User-friendly** error correction and training manual

\$219 U.S. / \$299 Can.

Contact your dealer or
send check/MO to:

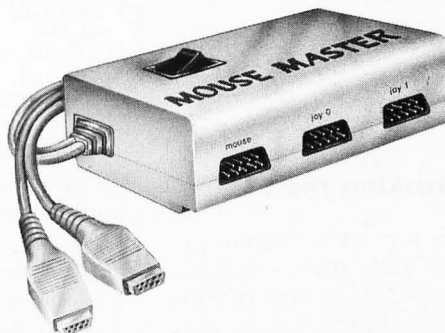
Nice & Software, Inc.
103 Queen Street S.
Kitchener, Ontario
Canada N2G 1W1
(519) 744-7380

Visa & MasterCard

Dealer and Distributor Inquiries Welcome

[illegible]

**New from the people who
brought you Monitor Master!**



Are you tired of fumbling under or behind your computer to swap your mouse and joystick cables? Are your cable and computer connectors worn out from all the plugging and unplugging? Then Mouse Master is a must for you!

\$39.95*



1930 E. Grant Rd.,
Tucson, AZ 85719

*Retail price does not include shipping & handling.

602-884-9612

MOUSE MASTER

CIRCLE #105 ON READER SERVICE CARD.

CIRCLE #101 ON READER SERVICE CARD.



**Any
Resolution**

Dial XR

by Maloney

Multi-tasking the ST

We all have our own dream programs. The two main things that I do with my ST are writing and BBSing. The problem is that BBSs, unlike commercial systems, are single-user. If someone else is on-line, then I have to keep re-dialing until the line is free. If I use an attack-dialing terminal program, then my computer is tied up. Frankly, I have better things to do with my time than to watch a screenful of "ATDT" lines slowly scrolling by.

What I need is a program that will allow me to write while simultaneously auto-redialing. Connecting with the BBS would signal me, and one quick point-and-click mouse operation would activate the terminal. After I had logged-off, a single keystroke would return me to my original text screen.

My file would be intact—the cursor would even still be right where I had left it.

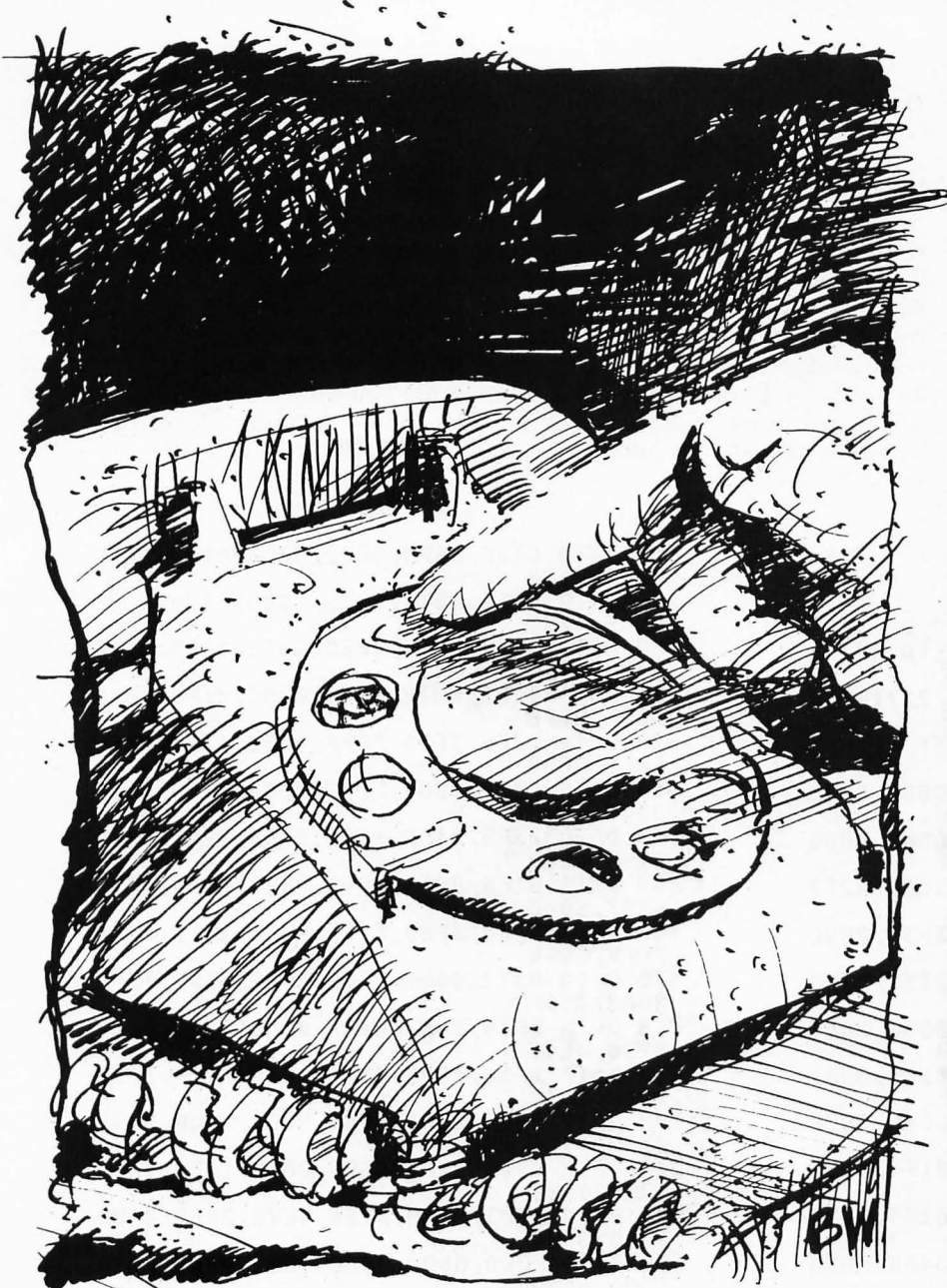
After about a year I realized that no one else was going to implement my idea; there just isn't a large enough market to justify the investment of time and money. After all, my dream program is not necessarily yours. You might want to attack-dial while forecasting with a spreadsheet, plotting your bio-rhythm cycle, or whatever. Besides that, we're all accustomed to using the programs that we already have. We don't want the hassle of learning new ones to do the same tasks.

The Power of the GEM Interface

GEM provides part of the answer for us—the ability to access programs as desk accessories. Most of these are con-

venience items like calculators and notepads. Others, such as the VT-52 emulator provided on the ST language disk, are much more powerful. Insofar as I usually only access message bases, I don't often need a terminal with upload/download capability. The emulator would normally serve my needs if somehow I could make it auto-redial.

This is what inspired me to write "**DIALXR.**" It functions as a shell, interfacing between the emulator and the operating system to turn the ST into an attack-dialing terminal. You can put it into the AUTO folder, and its operation is completely user-transparent. It is compatible with just about any GEM program (including the desktop), and it will also work (with some adjustments on your part) with many TOS programs. Best of all, it steals less than ¼ K of memory.



How It Works

second.

TOS was written to be "programmer friendly." By this I mean that its four major sections—GEMDOS, GEM, the BIOS, and the XBIOS—are accessed indirectly through memory vectors which point to the addresses at which they actually reside. Because these "hooks" are located in RAM, a programmer can modify their contents to point to custom routines.

I used this technique to intercept characters being sent from the keyboard to the modem on the RS-232 port. The BIOS vector is changed to the address of a routine which interprets keystrokes, setting and resetting flags and timers. The address of a second routine is placed in the vbl (vertical blanking) interrupt queue. This latter code is invoked about 60 times per

Magic Fingers

These two routines work together. After you enter the 'ATDT' (or 'ATDP') sequence to dial a number, the system waits for 30 seconds. If a connection has not been established, then the program forces the modem to disconnect from the line. It then waits for two seconds (to let the phone line settle), reconnects, and finally the "A" redial command is sent to the modem. The BIOS handles this as though you had the emulator on-screen and were manually typing in the command.

You can use any Hayes-compatible (or semi-compatible) modem connected to the RS-232 port. The only requirement is that it accept the "ATD" and the "A" commands.

So Let's Use It

If you're not familiar with the VT-52 emulator, you should check out the sections in your ST owner's manual about the emulator and changing the RS-232 configuration. Chances are that the only parameter that you will need to change is the baud rate. Be sure to save the desktop after doing so.

Enter and RUN the BASIC loader to create the file "DIALXR.PRG" on the disk in drive A. (Don't forget to check your typing with ST-Check.) Again I suggest that you place this file into the AUTO folder. Your boot disk will also require the emulator .ACC file.

Boot-up your ST. If your application program is GEM-based, then double-click on it, and after it is up, load in the file with which you will be working (if any). Turn on your modem and select "VT-52 Emulator" from the "Desk" menu. Enter the dialing sequence, and then return to your main program by pressing the "undo" key.

When the BBS answers your call the bell will sound, and the screen will change to reverse video. In about a quarter second the bell will again ring, and the screen will return to normal. Then just select the emulator and log-on. That's all there is to it!

After you log-off press "UNDO" to return to your application. You may, of course, elect to attack-dial another system before doing so. "DIALXR" is fully re-entrant.

Another option that you have is to ignore the on-line signal. Insofar as BBS programs will log you off if they do not receive a keystroke within a couple of minutes, if you've changed your mind about going on-line, you can continue with your main program. As an alternative you can switch your modem from "Data" to "Voice" or just turn it

off.

A Few Final User Notes

"DIALXR" plays by the rules of TOS, so you should not have compatibility problems with any application program—providing that it also follows proper programming procedures. Any program that insists on stealing the second vbl interrupt vector rather than searching the queue for a free one will incapacitate **"DIALXR."**

This also applies to other utilities that you might put into the AUTO folder. To prevent this problem, copy **"DIALXR"** into the folder last.

Another possible pitfall could arise if your main program intercepts the BIOS vector. If the program does not restore the vector when it exits, your ST will definitely crash. Also beware of video games—some of them can be exited only by resetting your machine.

The only other limitation was mentioned earlier. If your application is a

TOS program, you must dial through the emulator before loading and running the main program. Likewise, when you receive the on-line signal, you must exit before the BBS times out. As a rule of thumb do not initiate any unstoppable process which requires more than a couple of minutes.

I think that you will find that even if you have a full-featured terminal program you will be using it less and less as you become familiar with **"DIALXR."**

Listing 1:

ST Basic

```

100 OPEN "R",#1,"A:DIALXR.PRG",2
110 FIELD #1,2 AS B$: FOR I=1 TO 281:
READ X$
120 A$=CHR$(VAL('"&H"+LEFT$(X$,2)))+CHR
$(VAL('"&H"+RIGHT$(X$,2)))
130 LSET B$=A$: PUT #1,I: NEXT: CLOSE
#1: END
200 DATA 601A,0000,01F8,0000,0000,0000
,0000,0000
210 DATA 0000,0000,0000,0000,0000,0000
,4879,0000
220 DATA 001C,3F3C,0026,4E4E,5C8F,4267
,2F3C,0000
230 DATA 02F8,3F3C,0031,4E41,303C,0006
,207C,0000
240 DATA 04D2,4A98,57C8,FFFC,213C,0000
,00E6,23F8
250 DATA 00B4,0000,01EC,21FC,0000,0044
,00B4,4E75
260 DATA 0C6F,0001,0008,6660,0839,0001
,FFFF,FA01
270 DATA 6756,0C6F,0003,0006,664E,0C2F
,0044,000B
280 DATA 6608,08F9,0000,0000,01F4,0C2F
,000D,000B
290 DATA 6636,0839,0000,0000,01F4,6726
,0839,0001
300 DATA 0000,01F4,661C,23F8,04BA,0000
,01F0,06B9
310 DATA 0000,1770,0000,01F0,08F9,0001
,0000,01F4
320 DATA 6006,4239,0000,01F4,2079,0000
,01EC,4ED0
330 DATA 2F38,04A2,04B8,0000,002E,04A2
,3F3C,0007
340 DATA 2F3C,0003,0002,4E4D,5C8F,21DF
,04A2,2F39
350 DATA FFFF,8240,33DF,FFFF,8242,33DF
,FFFF,8240
360 DATA 4E75,0839,0001,0000,01F4,6738
,0839,0001
370 DATA FFFF,FA01,6630,0839,0002,0000
,01F4,6614

```

Listing 2:

Assembly

```

;*****
;*          DIALXR          *
;*      by Maloney         *
;*                         *
;*      Copyright 1988 by ST-Log      *
;*                         *
;*****
;
      text

```

```

380 DATA 61AE,08F9,0002,0000,01F4,13FC
,000F,0000
390 DATA 01F5,4E75,5339,0000,01F5,6608
,6192,4239
400 DATA 0000,01F4,4E75,2039,0000,01F0
,80B8,04BA
410 DATA 6AF2,0839,0003,0000,01F4,6640
,40E7,007C
420 DATA 0700,3F39,FFFF,8800,13FC,000E
,FFFF,8800
430 DATA 08F9,0004,FFFF,8802,33DF,FFFF
,8800,46DF
440 DATA 23F8,04BA,0000,01F0,06B9,0000
,0190,0000
450 DATA 01F0,08F9,0003,0000,01F4,4E75
,40E7,007C
460 DATA 0700,3F39,FFFF,8800,13FC,000E
,FFFF,8800
470 DATA 08B9,0004,FFFF,8802,33DF,FFFF
,8800,46DF
480 DATA 2F38,04A2,04B8,0000,002E,04A2
,3F3C,0041
490 DATA 2F3C,0003,0001,4E4D,5C8F,3F3C
,002F,2F3C
500 DATA 0003,0001,4E4D,5C8F,21DF,04A2
,23F8,04BA
510 DATA 0000,01F0,06B9,0000,1770,0000
,10F0,08B9
520 DATA 0003,0000,01F4,4E75,0000,0000
,0000,0000
530 DATA 0000,0000,0000,0002,2C08,062E
,100A,0A0A
540 DATA 0808,063C,140C,0808,0A08,0E2E
,0A08,5A0A
550 DATA 0800

```

Listing 1:

Checksums

100 data 790, 71, 953, 224, 820, 764
, 210, 16, 226, 30, 4104
260 data 160, 44, 84, 975, 91, 936,
995, 120, 252, 559, 4216
360 data 22, 95, 124, 66, 61, 30, 39
3, 454, 962, 65, 2272
460 data 405, 462, 143, 270, 197, 98
9, 867, 931, 159, 449, 4872

```

install_do    pea      super_do      ;address of subroutine
              move.w   #38,-(sp)      ;superexec funtion code
              trap     #14            ;call XBIOS
              addq.l   #6,sp          ;pop stack

              clr.w    -(sp)          ;no error code
              move.l   #760,-(sp)      ;bytes to keep
              move.w   #531,-(sp)      ;keep function code
              trap     #1             ;call GEMDOS

super_do      move.w   #6,d0          ;index value
              movea.l  #54d2,a0       ;_vbl_list + 4
loop          tst.l    (a0)+          ;is vector available?
              dbeq     d0,loop        ;if not take branch

              move.l   #vbl_do,-(a0)  ;store address in vbl vector

vector_change move.l   $b4,BIOS_default ;store old address
              move.l   #BIOS_mod_do,$b4 ;store new address

super_exit    rts                    ;return to continue

BIOS_mod_do   cmpi.w   #1,8(sp)       ;is device RS-232?
              bne.s    trap13_normal ;if not take branch

dcd_test      btst.b   #1,$fffffa01  ;are we on-line?
              beq.s    trap13_normal ;if so take branch

bconout_test  cmpi.w   #3,6(sp)       ;is it bconout?
              bne.s    trap13_normal ;if not take branch

D_test        cmpi.b   #68,11(sp)     ;is character "D"?
              bne.s    CR_test        ;if not take branch

              bset.b   #0,status_flag ;set prelim bit

CR_test       cmpi.b   #13,11(sp)     ;is character CR?
              bne.s    trap13_normal ;if not take branch

              btst.b   #0,status_flag ;is prelim bit set?
              beq.s    cancel         ;if not take branch

              btst.b   #1,status_flag ;is dial bit set?
              bne.s    cancel         ;if so take branch

              move.l   $4ba,timer      ;store current time
              addi.l   #6000,timer     ;add 30 seconds
              bset.b   #1,status_flag  ;set dial bit
              bra.s    trap13_normal  ;branch always

cancel        clr.b    status_flag    ;reset all bits

trap13_normal move.l   BIOS_default,a0 ;default trap 13
              jmp      (a0)           ;jump to it

alert_do      move.l   $4a2,-(sp)      ;push current address
              sub.l    #46,$4a2        ;make room

              move.w   #7,-(sp)        ;Control-g (ring bell)
              move.l   #500030002,-(sp) ;console and bconout
              trap     #13            ;call BIOS
              addq.l   #6,sp          ;pop stack

              move.l   (sp)+,$4a2      ;pop address

              move.l   $ffff8240,-(sp) ;push palette regs 0 & 1
              move.w   (sp)+,$ffff8242 ;pop 0 to palette reg 1
              move.w   (sp)+,$ffff8240 ;pop 1 to palette reg 0

              rts                    ;return to continue vbl_do

vbl_do        btst.b   #1,status_flag ;is dial bit on?
              beq.s    vbl_exit       ;if not take branch

              btst.b   #1,$fffffa01  ;are we on-line?
              bne.s    redial_test    ;if not take branch

              btst.b   #2,status_flag ;is connect bit set?

```

```

        bne.s      alert_test      ;if so take branch
        bsr.s      alert_do        ;ring bell and reverse screen

        bset.b     #2,status_flag  ;set connect bit
        move.b     #15,timer_A     ;set timer to 1/4 second
        rts        ;return to vbl int handler

alert_test  subq.b     #1,timer_A   ;time for alert_do?
        bne.s     vbl_exit        ;if not take branch

        bsr.s      alert_do        ;ring bell and reverse screen

        clr.b      status_flag     ;disable redial

vbl_exit    rts        ;return to vbl int handler

redial_test move.l     timer,d0     ;get target time
        cmp.l     $4ba,d0         ;is wait finished?
        bpl.s     vbl_exit        ;if not take branch

        btst.b     #3,status_flag  ;is redial bit set?
        bne.s     redial_do       ;if so take branch

        move       sr,-(sp)        ;push status register
        ori        #$0700,sr      ;disable interrupts
        move.w     $ffff8800,-(sp) ;push last register used
        move.b     #14,$ffff8800  ;select port A
        bset.b     #4,$ffff8802   ;set dtr bit
        move.w     (sp)+,$ffff8800 ;pop last register used
        move       (sp)+,sr        ;pop status register

        move.l     $4ba,timer      ;store current time
        addi.l     #400,timer      ;add two seconds
        bset.b     #3,status_flag  ;set redial bit
        rts        ;return to vbl int handler

redial_do   move       sr,-(sp)        ;push status register
        ori        #$0700,sr      ;disable interrupts
        move.w     $ffff8800,-(sp) ;push last register used
        move.b     #14,$ffff8800  ;select port A
        bclr.b     #4,$ffff8802   ;reset dtr bit
        move.w     (sp)+,$ffff8800 ;pop last register used
        move       (sp)+,sr        ;pop status register

        move.l     $4a2,-(sp)      ;push current address
        sub.l      #46,$4a2        ;make room

        move.w     #65,-(sp)       ;"A"
        move.l     #$00030001,-(sp) ;RS-232 & bconout
        trap       #13            ;call BIOS
        addq.l     #6,sp          ;pop stack

        move.w     #47,-(sp)       ;"/"
        move.l     #$00030001,-(sp) ;RS-232 & bconout
        trap       #13            ;call BIOS
        addq.l     #6,sp          ;pop stack

        move.l     (sp)+,$4a2      ;pop address

        move.l     $4ba,timer      ;store current time
        addi.l     #6000,timer     ;add 30 seconds
        bclr.b     #3,status_flag  ;reset redial bit

        rts        ;return to vbl int handler

        bss
BIOS_default ds.l      1
timer        ds.l      1
status_flag  ds.b       1
;status_flag bit assignments:
; 0  prelim
; 1  dial
; 2  connect
; 3  redial
timer_A      ds.b       1

        end

```

ST Gossip

from Hollywood, U.S.A.

by TG

Have You Heard That . . .

Some Atari executives have been working nights preparing an *advertising* campaign. Most at Atari now believe that the need for real advertising is crucial. Now if only they can get Jack T. to listen . . .

Strong rumors out of the West Coast indicate that if the cost of memory (RAM) chips doesn't fall soon, the cost of ST's in general, and Megas in particular, will be going *up*. This has already started happening in the PC world as the cost of clones and the various cards for them have started climbing for the first time in four years!

The Atari Super High Res Monitor has been sitting on a shelf in Sunnyvale for months waiting for the price of one meg chips to drop low enough to make the unit salable. What is this monitor? We hear it includes one meg of screen RAM built into the unit, with a 68000 chip for graphics processing. As well as maximum 1280 X 960 resolution and connection to the ST via the DMA port. This is supposed to be the monitor for the legendary EST (Enhanced ST), as well as the unit to be packaged with the ABAQ.

Timeworks has high hopes for their new product **Desktop Publisher**. Many who have seen it say the new GDOS-based desktop publishing package is much stronger than the old

Publishing Partner. Looks like the update from the guys at SoftLogic will come just in time to help them keep their position as Atari's number one DTP package. For those who haven't seen the package from Timeworks, it really is quite nice. The software uses GDOS to give a very readable screen representation of your final output. Among the many features that give this package super user-friendliness is the ability to take a graphic and drop it anywhere on your screen. Any text you may have on the screen will just move out of the way and automatically flow around the graphic.

Speaking of user friendliness, anyone who has tried to use the GDOS installed program provided with **Microsoft Write** has a clear lesson in what that term does *not* mean.

Still speaking of GDOS and **Publishing Partner** (and I know we spoke of them somewhere back there), a short conversation with the programmers at SoftLogic indicated that they think a lot more of the Atari laser printer now that they have had a chance to work with it a few months. According to them it will print from the *new Publishing Partner* much faster than the old trusty Apple laser printer they bought a few years ago. In fact, they now feel, all things considered, it's the laser printer of choice for the new

Publishing Partner.

Is it true that Atari is working on a new version of GDOS that will load re-scalable font descriptions rather than those archaic bit mapped memory hogs they are now using? That's what we hear.

How's the *new, new* operating system coming? For those of you who missed a few weeks somewhere along the line, Atari released a new "debugged" version of the operating system with the Megas some months ago. Then, just as the chips were starting to appear in the 1040s and 520s, Atari's Neil Harris, announced they were ready to button up the code on the *new, new* operating system. Atari was interested in any suggestions that user's might have which could be included to make it better. Hundreds of suggestions arrived (seems like half of them were asking for the Universal Item Selector to be included). Atari shut off suggestions after a reasonable period of time, and we were told they were busy figuring out what would and would not be included.

One small computer store in western Massachusetts, *The Computer Bug*, has decided not to wait, and has just written a new graphic interface for the STs. This new interface offers some of the

(continued on page 46)



ST JOHNS

GAME

Low Resolution Only

by Clayton Walnum

Moonlord Planetinsky was still a bitter man.

Even though he had succeeded in virtually single-handedly defeating last year's alien attack (the entire Titanian Territorial Guard had been stymied by the aliens' unusual strategies), even though he returned home a hero to the adulation of thousands, he found that, deep inside where it really counted, he was still as insecure as a newborn cub.

It was the name, you know.

It sounded so much like a title of office that people could rarely resist bowing when introduced. It was a matter of amusement for most, but Moonlord hated it.

His childhood had been no laughing matter, either. He had always been the one with the cootiumphaloids (imaginary creatures about the size of a temphibootawep; if the other kids said you had them, you were an outcast), and now as an adult, he still found that his unusual name was anything but an asset.

Why, he often thought, couldn't he have been given a normal name like Fredolotington Alnertopater or Eddyboperty Elnopilersop?

So he became tough—the toughest starfighter on the Saturnian moon of Titan. Nobody—*nobody*—dared cross him.

Now it seemed he had another job to do.

Moonlord stepped off the Slider-walkatron and crossed to the headquarters of the Titanian Territorial Guard, clutching the telegrammessagecard in his left hand. It was from Leeryup Coddleloop, Commander-in-Chief of the TTG. He snickered to himself as he remembered the last time he had seen Leeryup, tucked into a hospital bed, every part of his body except the bones swollen like over-filled cameladesertliquibags.

"Guess he won't bow to me again!" Moonlord said out loud. A few people glanced in his direction, but none let his gaze linger. Moonlord was a hero—and they loved him—but they knew better than to draw attention to his peculiarities. He drew a deep lungful of smoke from his smokyngstickolungolator, and exhaled a swirling blue tornado.

When he stepped into Commander Coddleloop's office, the gray-haired man behind the desk stood up and saluted. Even though Moonlord was a civilian, he now received the same respect as that awarded to an Admiral of the fleet. To say the least, the TTG were inordinately impressed by Moonlord's handling of the last alien invasion.

Moonlord sat down without returning the salute, and stared at the Commander, saying nothing.

The Commander sat slowly, fighting the urge to bow with all his soul. Heavens, but old habits died hard!

"Uh...ahem," he began eloquently. "...uh... To say the least, the TTG were inordinately impressed by your handling of the last alien invasion."

Wow, thought Moonlord, *Deja vu*. But he said nothing, just sat, waiting.

"We have a tiny problem," the Commander tried again, "one that requires your...er... delicate touch."

Moonlord's eyebrows climbed his forehead. "You wouldn't by any chance be referring to the new fleet the aliens have sent out, would you?"

"Well...it's a problem kind of...er... similar to that."

"Similar?"

"Um... very close to that, actually."

"How close?"

"Sort of... well... 'identical' would be the appropriate word, I guess."

Moonlord sighed. "Are you or are you not referring to the new alien threat?"

"I believe that would be an accurate paraphrase of my previous remarks."

"Have you ever considered politics?" Moonlord asked.

Why couldn't he have been given a normal name like Fredolotington Alnertopater?

"Well..."

"Never mind. It was a rhetorical question."

Moonlord stood up and crossed to the Commander's newly installed compudigibinotometer-ST, the one that had recently replaced the long-loved compudigibinotometer-XE, and called up the galactic map.

The aliens were everywhere.

"Let the good-times roll," Moonlord muttered.

"Excuse me?" said the Commander, standing to get a better look at the screen.

"I'll take the job," Moonlord said, turning toward the Commander. "I'll show those alien scum that they can't mess with Titan."

The Commander positively glowed. "Thank you, thank you, thank you!" He was so delighted that he forgot to control his inner impulses. Before he knew it, he was bending at the waist, performing an elegant bow. "Ohhhhhh, nooooo..." he muttered.

It was the Commander's opinion that hospital food hadn't improved much since his last stay.

So, where is it?

Moonlord ST is a translation of the original **Moonlord** published in issue 46 of *ANALOG Computing*. Those of you who are familiar with that game will find that this version, though greatly enhanced graphically and using the mouse instead of a joystick for input, is almost identical in game play. The only differences are the addition of a novice level and the ability to "ram" alien craft.

Actually, there is one other major difference: You don't have the option of typing in the program for **Moonlord ST**. Just like most full-scale ST programs, the source code is much too large to be published here. All game files and the source code are available on this month's disk version or on the Delphi ST SIG.

Note that the file MOONLORD.DAT must always be in the same directory as the main program file, MOONLORD.PRG, and the disk should remain in the drive during game play.

Playing Moonlord ST

When you run the program, the first thing you'll see is the title screen. After you've finished admiring Maurice Molyneaux's excellent work, click the left mouse button and select a level of

play. Those who've never played **Moonlord** before should select the novice level. **Moonlord** is a tough game to complete and, until you've managed to develop some good strategies, you won't have a chance on the expert level. The novice level gives you a higher energy allotment, and the aliens' attacks have a lesser chance of damaging your ship.

After selecting your level, you'll see the galactic map, represented on your screen by an 18 x 8 grid. Each square in the grid is one sector of the galactic milieu, and hidden within these 144 sectors are the 50 alien craft you must locate and destroy. Since aliens always travel in pairs, only 25 sectors actually contain the enemy.

To make your job a little easier, there are two starbases you can dock with, to stock up on supplies and make repairs. There's one at each end of the galaxy, and, of course, just like the aliens, they're randomly placed at the beginning of each game, forcing you to explore.

To win the game, you must locate and destroy all 50 alien craft. You have only 100 Galactic Standard Days in which to complete your mission. It'll take careful conservation of supplies and planned movement, so those who like to leap into the fray without a strategy will find failure a constant companion.

Though there's only one way to win the game, there's many ways to lose (can't make it too easy for you, now can we?). The first is to run out of time. You've got 100 days. No extensions. All begging will be ignored.

The second way to lose your hero status is to allow your energy to run out. Keep your eye on it; when it's gone, so are you. Don't forget to check the status of your weapons, either. If you should be in the heat of battle and find that both your weapons systems are down, you'll have to resort to ramming the aliens (more on that later). That means heavy damage. Also, every time you ram an enemy, you're taking a one-in-ten chance of destroying your own ship.

Finally, use of your ship's warp capabilities is a risky venture indeed. Each time you decide to utilize them, you're taking a one-in-ten chance of destroying your engines and ending the game.

The bridge

Below the galactic map, you'll find

"I'll show those alien scum that they can't mess with Titan."

the bridge. This is where you gain access to the ship's main functions. There are four systems available here: scanners, cruise engines, a status display, and warp engines (weapon systems are accessed from the scanner display). To select a system, place the mouse pointer over the system name and click the left mouse button.

Note that at times some of your systems will be damaged and thus unusable. You can tell at a glance which systems are down: their entries in the systems menu will be disabled. The only exception to this is the long-range scanners. They work automatically each time you move, so they have no menu entry. You can check them on the status display (see the section "Status" below).

Cruise

To move your ship from one galactic sector to another, select the "cruise" command. You are allowed to move in any of the eight compass directions, but you should note that diagonal moves are actually counted as two moves, and the required energy and time are deducted as if the move were completed with two non-diagonal moves. When you click on the cruise systems, you'll be asked to select your destination sector. Place the mouse pointer over the sector and click the left mouse button. Your ship will appear in the target sector.

Each sector of movement uses ten units of energy and one day of time.

Status

Throughout the game, it's important to keep close tabs on your ship's condition and supplies. You can't afford to be stuck far from a starbase when your energy is almost depleted, and it helps to know what weapons are functional before you spring into battle. All this information is available in the status display. To view the status display, select the "status" system from the bridge menu. The status display screen will then pop into view.

Your ship's six systems are displayed on the left, each followed by a number indicating how many days are needed to repair that system. A zero means the system is fully functional.

On the right, information on supplies, as well as the time remaining and the number of aliens remaining, can be found.

Damaged systems

Damaged systems must be repaired before they can be used. Damage is measured by the number of days the crew requires to complete repairs. If you don't need the damaged system right away, you need do nothing. The crew will automatically get to work, applying their best efforts to the restoration of your ship. Remember: One sector of movement on the galactic map consumes one day. A system that requires three days to repair will be operative after a move of three sectors.

If you find you must make repairs immediately, before continuing with the game, you may do so by selecting the "repair" system from the status subsystem menu. Use the mouse to tell the ship's computer how long you wish to wait for repairs by placing the mouse pointer over one of the arrows and clicking the left mouse button. When you're satisfied with your selection, press the right mouse button. The repairs will be made and the status screen updated.

If more than one system needs repair, the times are not added together. Each system has its own crew. For example, if your photon launchers require four days to repair, and your short-range scanner needs two days, it'll take only four days to fix both systems. Given the above circumstance, if you should select only two days of repair time, the short-range scanner will be operational, while the launchers will require two additional days of repair before you can use them.

Don't forget that the time you spend waiting for repairs will be subtracted from the time available to your mission. Sometimes it's better to continue crippled then to waste a lot of time waiting for repairs to be completed.

Warp

Should you find that you must move a long distance in a minimum amount of time, the warp engines may fill your need. Unfortunately, the warp engines are still experimental; their safety and reliability cannot be guaranteed. You have no control over where you'll end up, and each warp carries a one-in-ten chance of leaving you engineless, helplessly afloat in the timeless void of space. In other words, the game could come to an abrupt end.

Each warp consumes one day and 30 units of energy. Due to its undependability, you may have to jump several times before you get where you want (or at least in the general area).

"Let the good-times roll," Moonlord muttered.

Long-range scanners

You have two types of scanners on your ship: long-range and short-range. The long-range scanners fill in the galactic map as you move, and since they function automatically, you need do nothing except repair them when they become damaged.

The long-range scanners examine the sectors adjacent to your position and mark the galactic map appropriately. Empty sectors are indicated by a white dot, aliens are represented by a red circle, and starbases by a blue circle. Your current position is marked by a green circle. In most cases, your position marker will be outlined in yellow. However, if you should be in a sector containing aliens or a starbase, your marker will be outlined in red or blue, respectively.

Short-range scanners

The "scan" system on the bridge menu activates the short-range scanners. When you select this system, the short-range scanner display will pop up.

The short-range scan allows you to see your current sector in greater detail. Each sector of the galactic map is divided into 36 smaller sectors. Suns, aliens, starbases, as well as your own ship are all represented by icons on the short-range scan display. Four systems commands are available from the short-range scan sub-system (SRSSS) menu: bridge, cruise, phaser and photon.

To return to the bridge, select the "bridge" option.

Short-range cruise

You may move about in the short-range display in much the same manner as in the galactic map. Select the "cruise" system from the SRSSS menu, press the left mouse button, then use the mouse to select your destination.

Unlike the galactic map, your move-

ment here is somewhat restricted. You can't move through a sun, an alien or a starbase. If anything is in your way, you must maneuver around it. Also, "diagonal" moves are not allowed. This is because, as I mentioned before, diagonal moves are interpreted as two non-diagonal moves. Since the aliens will attack each time you move, only single moves are allowed.

Movement on the short-range display consumes no time, but uses three energy points per sector.

Phasers

The phasers are the first of your weapons systems, and your most powerful. When activated, they release a burst of electro-magnetic energy in every direction, damaging any alien craft on your scanners. Nothing can block their energy beams, not even a sun. The amount of damage done depends on the number of alien craft present and the distance they are from your ship. Damage is cumulative. You may have to fire more than once to get the job done.

To activate the phasers, select the phaser system from the SRSSS menu, then use your mouse to tell the ship computer the amount of power to allocate. (Click on the arrows to increase or decrease the amount, and then press the right mouse button.) Each power point will be subtracted from your remaining energy, so be stingy, allocating just enough to get the job done.

Photons

Photon torpedoes (globes of compacted light energy) can be used to fire on any alien craft that is in alignment (horizontally, vertically, or diagonally) with your ship. Their range is sufficient to strike any ship on your scanners, and a strike is always fatal. To fire a photon, select the photon system from the SRSSS menu, and then use the mouse to click on the appropriate point on the photon aiming compass.

Firing a photon consumes no energy, but nothing comes for free. In order to fire photons, your launchers must be in working order, and you must have photons on hand. At the start of the game, you are given ten photons. You'll be restocked only when you dock with a starbase. Obviously, you're going to have to use them judiciously.

Ramming

If you should find yourself in the midst of battle with all your weapon systems down, you can still defeat the

aliens by ramming them with your ship. Because your ship is much larger than the aliens', this will always be fatal to the enemy. However, by resorting to such desperate measures, excessive damage may be caused to your ship. Specifically, up to three systems may be damaged, and there's a one-in-ten chance that the damage will be sufficient to cripple your ship permanently, thus ending the game.

Starbases

When you set out from Titan Base, your ship will be carrying all the supplies it can hold. It'll be necessary at certain points in the game to stock up. For this reason, there are two starbases, one at each end of the galactic milieu.

The starbases move from game to game, and will not be marked on the galactic map until you locate them—one of your top mission priorities, obviously. Once you locate a starbase, you must—if you plan to restock your supplies and make repairs—go to the short-range scanners and dock with the base. Docking is accomplished by moving your ship on top of the base. All your supplies will be restocked, and all systems will be repaired.

Mission complete

All missions, regardless of success or failure, will be evaluated by the personnel at Titan Base. Your score is based on the number of aliens you destroyed, as well as the amount of time and energy you used (the less, the better). Also, you'll get much higher scores at the expert level than you will at the novice level.

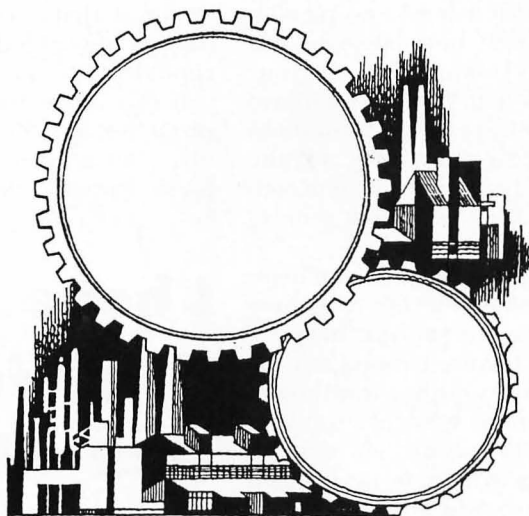
At the end of the game, if you want to play again, press the left mouse button. To exit back to the desktop, press the right mouse button.

Kudos

I wish I were an artist, but I'm not. Though I designed all the game screens and icons, it was *ST-Log* Contributing Editor Maurice Molyneaux who made them come to life. He took what were only barely acceptable displays and turned them into a professional-looking piece of work. Thanks, Maurice!

The sound effects for **Moonlord ST** were created using **G.I.S.T.** by Lee Actor and Gary Levenberg of Synthetic Software (G.I.S.T. is available through The Catalog). This is a sensational program that takes all pain out of generating sounds and makes implementing those sounds in your programs simple beyond belief.

FEATURE



Software Evolution

From Art to
Engineering

by Karl E. Wiegiers

Long ago, in a galaxy far, far away, people began to program computers. At that time, the computer itself was a room-filling monstrosity, and the programming challenges were quite different from those of today. Everything was done at a machine language level, which is dandy for computers but a bit taxing on the human eye.

Not so long ago, in a nearby galaxy, people started programming in assembly language, and then in third-generation ("high-level") languages such as FORTRAN, COBOL, and PL/I. The programs became larger as the machines shrank. Unfortunately, the software problems became larger, too, and some interesting trends began to emerge.

Large software projects were nearly always completed over budget and late, if at all. The danged users kept changing what they wanted, and the over-

taxed programmers could scarcely keep up with them. Hence, programs were forever being altered after they were declared complete. In fact, data-processing departments were spending several times the dollars and man-years that they anticipated for program maintenance (adding enhancements, fixing bugs, and changing to conform to current user requirements).

Just yesterday, in the galaxy right next door, the microcomputer was invented. This opened the world of computing to a whole new generation of amateurs: hackers, if you like. We were back to assembly language, although Bill Gates soon opened the floodgates of high-level languages by introducing the first BASIC interpreter for micros.

Now, hackers tend to be an independent lot. (For an informative and entertaining historical perspective, read *Hackers*, by Steven Levy.) In the early

days, clever microcomputer programmers taught their machines how to sing, dance, and show pretty pictures. They were more concerned with the latest animation techniques than with writing structured, maintainable code. And, hey, with only 16K or so to work with, you can live with that. But again, the computers got smaller, cheaper, and more powerful at an astounding rate, and the programmers only slowly figured out better ways to create the code that rapidly expanded to fill all available RAM.

Today, on our very own planet, there is a widely recognized "software crisis." The rate of progress in software development hasn't begun to keep pace with that of the rapidly evolving hardware on which the programs run. The expectations of users have gone up, and software customers often are dissatisfied with the final product they acquire. The quality of the software often leaves something to be desired. Maintenance has become a major issue: look how many different releases of commercial packages appear and the fresh problems that crop up with each new version. And the developers frequently don't have a good idea of how long it's really going to take to get the next product out the door. The result is vaporware.

In short, we can no longer afford the luxury of computer programming as art, except for those who still view programming mainly as an exceptionally intriguing hobby and learning experience. To make a buck with software now requires a more systematic, rigorous, and structured approach, akin to that used for producing the computers themselves. The old generation of computer programmers as independent artists is (slowly) giving rise to a new generation of "software engineers."

Is this really such a big deal? Well, consider that IBM estimated the worldwide costs of software development and maintenance in 1986 to be some \$135 billion. That's a fair piece of change. And while computing hardware has been growing in speed and power at a rate of around 30% per year, software development productivity seems to be increasing at only about 4% annually.

Yes. It is a big deal.

Wherefore Software Engineering?

Software is just like hardware, only



Evolution

completely different. Hardware (meaning machines of any kind) can wear out; software can't. (Of course, the floppy disk can wear out and the software can become obsolete, but these are hardware and business problems, respectively.) Hardware can be rigorously tested to make sure it works right under specified conditions—it can be stressed. There's no way to exhaustively stress-test most software. Once in place, hardware is difficult and expensive to update, whereas software can be changed much more readily, at least from the user's point of view (think of the contents of a book versus the contents of a floppy disk).

Here's a simple hardware/software comparison you can do in the privacy of your own mind. When was the last time you asked the Maytag man to come out and add a variable speed option to the spin cycle on your washing machine? Now, when was the last time you sat at a keyboard and said, "Gee, the next version of this program is supposed to be able to . . ."

Despite these differences, the software development sequence does have a well-defined life cycle, quite similar to that of hardware development. The historical view considered the software life cycle to consist of steps to analyze the problem, design a software solution for the problem, write the required programs, test the system, install it, and maintain it. The goal of software engineering is to provide a more controlled approach to the software life cycle that will facilitate the efficient development of higher quality products than was possible using earlier methods.

Here's an irony for you. The hardware business has been revolutionized by the techniques of computer-aided design (CAD), engineering (CAE), and manufacturing (CAM). Doesn't it seem strange that the software industry is still so dependent upon the individual efforts of individual human beings?

The software engineering philosophy does emphasize the use of automated tools to facilitate the program development process. Such tools include computer-aided software engineering (CASE) products, as well as programming environments and utilities optimized for productivity. The latter include such things as syntax checking

program text editors, powerful compilers, linkers and debuggers, and usability environments like GEM.

But tools of any kind are only useful if they are applied in the context of a systematic methodology. Several software engineering methodologies (or "paradigms," if you want to baffle your friends and alienate your in-laws) have evolved in recent years. Some of these involve graphical "languages" for the structured analysis and design of software systems. We'll talk about some of these later on.

Another aspect of software engineering is the development of procedures for managing and monitoring the technical activities involved in program development. For example, how do you measure the productivity of a software engineer? With hardware, it's easier to measure productivity in terms of quantity (how many widgets?) and quality (did they all do their little widget thing right?). What does software quality even mean? And if you can't measure your productivity, how can you realistically project how long it will take you to complete a particular project? Tough questions, with no easy answers.

Software Life Cycles: Before

Many of you began your computing career with an 8-bit Atari of some sort. What was the "life cycle" of a program you wrote back then like? Well, you began with an idea. That was your planning step. You probably had a rough idea of how to proceed. That was analysis. If you are a particularly systematic person, you might have drawn out a flowchart, at least for the trickier sections of the program. That was the design step. But most of the time you skipped design and went right into coding, which consisted of typing your lines of Atari BASIC code as fast as you could into a single source code file, with perhaps a subroutine or two.

Testing consisted of typing RUN and observing what happened, then changing the lines that blew up (BASIC interpreters are great for sloppy . . . er . . . I mean *casual* program development). Installation consisted of giving a copy to a friend. Maintenance didn't really exist, partly because you were on to the next project, and partly because you didn't have any documentation (huh?) to remind you of what you did.

Don't try to lie to me about this; I've been there.

This unstructured, evolutionary

programming approach works moderately well for smallish BASIC programs, but there are some problems, besides the obvious high chaos level. One is that your variables in Atari BASIC are all global; that is, they are known to all parts of the program. While this is convenient, it's also dangerous, because you can inadvertently use the same variable name for different purposes. Also, you are limited to single files of BASIC source code, since there aren't easy ways to join multiple files

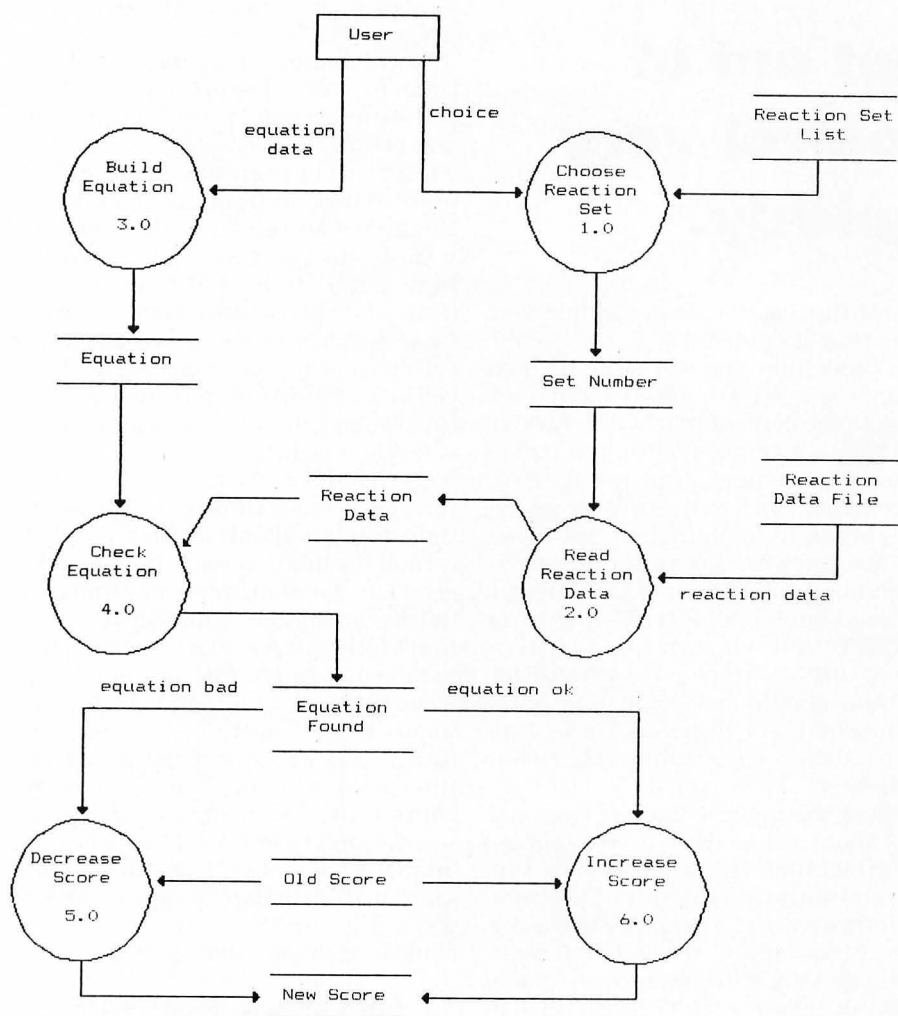
**I hope you
don't feel
that all this
talk about
engineering
is taking the
fun out of
computer
programming.**

together. You can, of course, run one program from another, but you can't really link two smaller programs together to create a large one.

BASIC is not a highly structured language, making it difficult to partition a program into discrete, bite-sized pieces that fit together nicely. The lack of good selection logic statements (like IF/THEN/ELSEIF) makes it hard to adhere to the structured programming precept of each program having exactly one entry point and one exit point. I've seen spaghetti code mazes of GOTOs and GOSUBs that would turn your hair white. And the need for line numbers makes it hard to create libraries of useful subroutines to merge into different programs you might write. Did you ever try to squeeze some code in between lines 1304 and 1305? 'Taint easy.

Things in the Atari world have changed, along with the rest of the computer world. Modern languages for

Figure 1. Sample Data Flow Diagram



**When was
the last time
you asked
the Maytag
man to come
out and add
a variable
speed option
to the spin
cycle on your
washing
machine?**

the ST, such as C, Pascal, and the newer BASICS like GFA and LDW, are more structured and modular. They let you write smaller program segments (called "modules") to perform specific functions, compile them separately, and link them together to create the runtime program. In the 8-bit Atari environment, Action! comes closest to being a modern, structured language.

As memory became cheap and abundant, programs became big and complex. It's much easier to create and modify a small program performing a single function than a large monolithic program. The trick, then, is to break a large program into a bunch of small modules, each of which can be designed, coded, and managed easily. You build a large software system by fitting together lots of small modules like bricks. The mortar that binds the modules together is the data interface between each pair of modules. By precisely defining these data interfaces (vari-

ables passed, etc.), it becomes easy to reuse an existing module. Also, the information within each module can remain local to that one module, minimizing inadvertent variable changes that wreak havoc elsewhere in the program.

Software Life Cycles: After

The sequence of events involved in creating a large system in the software engineering mode is somewhat different from that followed in the days of yore. The advantages become especially great if there are several people working on the same project. Let's see how you might do things in 1988, using a software engineering approach.

1. Perform a thorough, structured analysis of the problem, including writing a complete list of specifications describing what the system is supposed to do. I'll describe what I mean by "structured analysis" in the next section. Make sure you are solving the right problem, and that you under-

stand exactly what the program (or system of programs) must accomplish to be successful. The more complete the statement of requirements is, the more likely you are to come up with a satisfactory system.

2. Design the software system in detail *before ever writing any code*. Make sure the system you design is valid, complete, and can be implemented using the language(s) and hardware you have in mind. Believe me, it's much easier, faster, and cheaper to correct an error in the design than to fix the final product. Structured diagramming techniques can be used in the design step, and computer-aided software tools exist to help with this step (more about these later).

The hard part here is forcing yourself to hold off on coding until the design is complete. Face it, we all program computers because we love it. I've found it very difficult to discipline myself enough to avoid coding until I'm really sure of what I need to do, but it's

Evolution

usually worth the wait. Oh, all right, I admit it, I'll write snippets of code to try things out, but I call it "prototyping," so that's okay. (In fact, prototyping is an alternative software life cycle methodology that can be very effective, but that's a topic for another day.)

An important part of your design is to partition the problem into small enough pieces so that each program module is performing a single function. How you define "single" is up to you, but generally you want to try to limit module size to 100 lines of source code or less. Effective partitioning makes it easier to spread the work among different programmers. Also, you can identify modules that can be reused, either from a previous project or in the next project. This off-the-shelf-parts approach is another effective aspect of hardware engineering, and the payoff is just as great with software.

Each module should be designed to fit together with the others independently, essentially as a "black box" whose internal functioning isn't really important to other modules (or to people who use it as a part of another system, for that matter). Modules communicate through a well-defined data interface of passed variables and data structures. Try to avoid using global variables wherever you can, for the reasons mentioned earlier. Try to hide the

Things can get out of control very quickly.

data within each separate module from the other modules.

4. Okay, now you can write the program! Use good structured techniques, stick to the data interfaces described in the last paragraph, give your variables descriptive names, and avoid GOTO statements. Stick to the three main logical program building blocks of sequence (one statement after the other), selection (choosing one path from a choice, such as IF/THEN/ELSE or SELECT/CASE constructs), and iteration (FOR/NEXT type processing). Each module should have just one entry point and one exit point. These goals aren't always 100% achievable, but do your best.

There was an excellent series of articles about using structured programming techniques in BASIC way back in the May/July 1984 issues of the now-defunct *Creative Computing* magazine. The ideas apply to any language, although they are harder to implement in BASIC (or assembler—forget it!) than

in newer languages. Any modern computer programming textbook will also describe structured programming principles.

5. Integrate the modules and test them to assemble the final, completed software system. If you do this right, you can fit the pieces together a bit at a time, either from the top of the program down or from the bottom up. Whole books can be written on software testing, so I won't go into detail here. Suffice it to say that all of us can do a better job of testing our products, by systematically going through a wide selection of test case input. Don't forget to look at the output. Just because the program didn't crash doesn't mean it worked right!

6. Document what you did, both in the source code (internal documentation) and in separate written form (external documentation). If you did a good job of design, much of your external documentation is already complete. Don't forget things like lists of the files involved, data file formats, etc. It's always important to include some comments in the source code about what the module is for, the details of the data interface for the module, variable and subroutine names and functions, and so on. Don't base your documentation on just what you see in published program listings, either. Some of it is very good, but much is too scanty for commercial-scale development.

But I Like to PROGRAM!

I know. The software engineering philosophy involves a lot of front-end effort before we ever get to the really enjoyable part of writing code. And for the casual programmer, who does this just for fun, it isn't worth the time to really do all the systematic groundwork that I described. But it works!

The first time I tried using structured program design methods, I felt that I enjoyed about a tripling of my efficiency. The quality of the programs you write will go up if you take even a limited engineering approach. And you'll really become a believer if you ever have to do any serious maintenance programming. The software engineering approach greatly simplifies the task of correcting or enhancing an existing program, especially if someone else wrote the original. Take it from somebody who's been there. The key word is "structure."

Structured Analysis and Design

In the olden days, flowcharts were often used for diagramming the logic of a computer program. While flow-

Figure 2. Sample Structure Chart

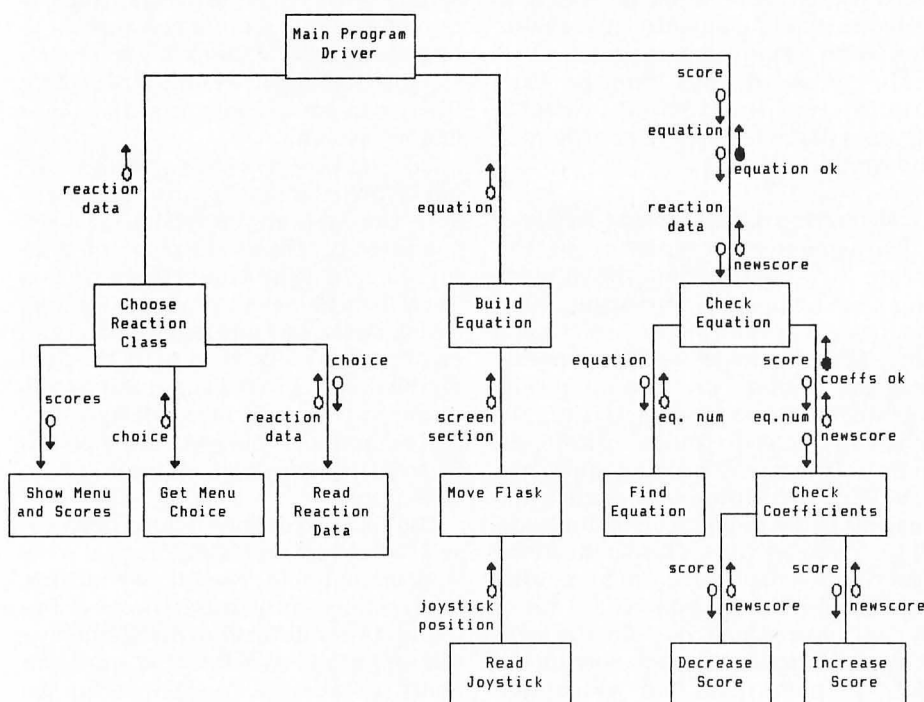


Figure 3. Sample Data Dictionary Entries

```

reaction-data-file = 7{reaction-data}

reaction-data = 16{chemical-formulas}
               + number-of-reactions
               + number-of-sets
               + 1{equation-description}16

equation-description = reaction-number
                    + 4{compound-numbers}
                    + 4{coefficients}

number-of-sets = [2:4]

```

English equivalent: The reaction data file consists of 7 blocks of reaction data. Each block of reaction data consists of 16 chemical formulas, the number of unique reactions in the set, the number of equivalent reaction sets, and from 1 to 16 blocks of descriptions of the possible equations that can be formed. Each equation description consists of a reaction sequence number, compound numbers for the four compounds in the equation, and coefficients for the four compounds in the balanced equation. There can be either 2 or 4 equivalent reaction sets in each reaction data block.

charts are useful tools, they have some limitations. For one thing, they imply a sequence of events, by indicating how control flows from one operation to another. This might be okay for diagramming the internals of a module, but it doesn't work well for describing an overall software system, in which a different sequence (or even set) of events might take place each time the program is executed.

In the 1970's, several alternative methods were devised for so-called structured systems analysis and design. Many of these concentrate on the flow of *data* within a software system. After all, that's what software is all about. The most important question to ask during the analysis phase is, "What is the output of this system supposed to be?" The next question is, "What data do you need to generate this output and where does it come from?" Now that we have defined the outputs and inputs of the system (in that order), everything else is just a matter of transforming input data into output data. What could be simpler?

The most popular diagram used for both structured systems analysis and design is the "data flow diagram" or DFD. A sample DFD is shown in Figure 1. (The figures are all based on portions of an educational chemistry game I wrote once upon a time.) A DFD simply illustrates the relationships between all the pieces of a system; you shouldn't try to infer anything about the sequence in which various processes may be executed.

There are really only four kinds of elements in a DFD, each of which must be given an appropriate label:

1. Processes—transform inputs to outputs. They are frequently represented as circles or rounded rectangles, called "bubbles." A process corresponds to a program module or group of modules.

2. Data Stores—repositories of data. They are shown as two parallel horizontal lines, and correspond to files, databases, or internal variable storage.

3. Externals—entities outside the software system that communicate

with the system, by passing data in or receiving data from the system. Externals (sometimes called terminators) are shown as rectangular boxes.

4. Data Flows—data that moves between stores, processes, and externals. They appear in the DFD as lines with arrowheads that indicate which way the data is flowing. The data flow label indicates exactly *what* data is flowing. An unlabeled flow is assumed to have the same label as that of the data store to which it is connected.

Data flow diagrams aren't limited to software applications. They can just as easily be used to represent a system in which actual objects, not data, are flowing from one place (process, external, or store) to another. An example is a warehouse system, with, say, computer parts being moved about. The DFD is just a general way to diagram some system, without regard to the hardware or language that might ultimately be used for implementing a software system.

A data flow diagram is a "model" of the software system you are building. That is, it's a visual representation of the system that can be used to show the purpose of the system, how it connects to the rest of the world, and what components it is made of. The great thing about a model is that it's an awful lot easier to change the model than to change the real thing. Consider a three-foot scale model of an aircraft carrier. If you wanted to move the ship's island from the starboard side of the ship to the port, wouldn't you rather do that on the model *before* you built the full-size ship and said "Uh, oh."? I would.

DFDs are great for facilitating communication between the people who request a software system and those who have to create it. In this case, a picture is worth at least 1024 words. By obtaining very early a complete picture of what the system is intended to do, you are much more likely to come up with a final product that makes the user feel that his money is well spent. Structured analysis and design greatly reduce the chance of errors and oversights.

Unless your software system is very small, you can't squeeze a complete description of it into a DFD that will fit on one piece of paper (or one computer screen). Whatever shall we do? We'll break the problem into several levels, with increased detail shown at each successive level.

First, draw a DFD containing only one process, representing your entire system. This DFD is typically called a "context diagram," because it shows

TEACHERS

Grade Book ST

- * Create and maintain student files, up to 200 students, 50 marks each, any num. of classes
- * Allows weighting and categories
- * Provides the following types of output:
 - Mark summaries with or without stud. names
 - Statistical analysis summaries with mean, median, ranking, breakdown by A's B's etc.
 - Attendance recording sheets
 - Individual student reports with comments
- * Uses dialog boxes to enter and update info, move easily through student records using up and down arrows.
- * Works with ST or MEGA in monochrome or color, any standard printer.

Send just \$34.95 to:

MICRO ELECTRONIC ARTS

10 Redwood Road,
Brantford, Ontario,
Canada. N3R 3M1

More information, dealer info, school licensing available on request, Ontario residents add 7% pst.

CIRCLE #107 ON
READER SERVICE CARD.

Computer Garden

Wilkes-Barre & Scranton's Favorite Computer Store

Accolade	Intersect	Soft Logik
Hardball.....\$25	Interlink.....\$25	Pub. Partner.....\$60
Pinball Wizard.....\$23	LC Tech.....\$39	Pub. Prtnr Pro.....\$120
Test Drive.....\$25	Stereotek kit.....\$139	Font Disk 1or 2.....\$20
Antic	Megamax	Springboard
Spectrum 512.....\$56	Laser C.....\$159	Certificate Mkr.....\$33
Cyberstudio.....\$70	Michtron	SSI
Atari	GFA Basic.....\$39	Wizard's Crown.....\$25
520STFM.....\$Call	GFA Compiler.....\$39	Rings of Zilfin.....\$25
1040ST.....\$Call	GFA Companion.....\$33	Phantasia.....\$25
Mega-2 ST.....\$Call	GFA Basic book.....\$25	Phantasia II.....\$25
Mega-4 ST.....\$Call	Microprose	Phantasia III.....\$25
Avant-Garde	F-15 Strike Egl.....\$25	War Game Con.....\$23
PC-Ditto.....\$Call	Gunship.....\$33	Star
Avatex	Silent Service.....\$25	NX1000 Printer.....\$169
1200e modem.....\$79	Microsoft	Color version.....\$229
2400 modem.....\$169	Microsoft Write.....\$89	Printer cable.....\$15
Modem cable.....\$15	Panasonic	Sublogic
Batt.Included	1080-II Printer.....\$175	Flight Simulator.....\$33
DEGAS.....\$49	1091-II Printer.....\$195	Scenery Disk 7.....\$15
DigitalVision	Printer cover.....\$9	Scenery Dsk 11.....\$15
Computereyes.....\$179	Printer cable.....\$15	Supra
Dr. T's	Practical Per.....\$15	2400 modem.....\$139
KCS.....\$189	Monitor Master.....\$44	Modem cable.....\$15
The Copyist.....\$189	Mouse Master.....\$35	Timeworks
MIDI Recording.....\$26	SeymourRadix	Wordwriter.....\$49
Studio.....\$26	IMG-Scan.....\$79	Datamanager.....\$49
Epyx	Sierra	Swiftcalc.....\$49
Impos. Mission II.....\$25	Space Quest.....\$33	Partner.....\$33
FTL	Space Quest II.....\$33	Desktop Publish.....\$79
Dungeonmaster.....\$25	King's Quest.....\$33	Trio Eng.
Sundog.....\$25	King's Quest II.....\$33	Digispec.....\$39
Future Sys.	King's Quest III.....\$33	Unispec.....\$49
GTS-100 drive.....\$199	Donald Duck.....\$16	Unison World
ICD	Police Quest.....\$33	Printmaster.....\$25
F20A-ST drive.....\$1	Liesure Suit.....\$25	Fantasy ArtGiry.....\$20
	Mother Goose.....\$20	Fonts & Borders.....\$23

Toll-free order line:

1-800-456-5689

For information call 1-717-823-4025

3% charge for VISA-MC-AMEX. Shipping, extra.
Computer Garden, 106 W. Carey St, Plains PA 18705

CIRCLE #108 ON
READER SERVICE CARD.

**Just
yesterday, in
the galaxy
right next
door, the
microcomput-
er was
invented.**

your system in the context of the rest of the world (all the externals that connect to the system). Then, use your microscope to peer inside that one process, and draw a second diagram containing several processes, representing the principal functions of your system. Each of those processes might be further expanded, showing even more detail, and so on.

This is the partitioning process I mentioned earlier. The ultimate goal is to subdivide the processes until each process on your lowest-level DFDs represents an individual program module that performs a single function. By drawing many DFDs at different levels, you don't have to try to comprehend the entire system by studying one mammoth diagram with many symbols and lines on it. A good rule of thumb is to limit each DFD to around seven or fewer processes.

Once you get your DFDs down to the module level, another diagramming method is used. "Structure charts" show the hierarchical relationships among modules, as illustrated in Figure 2. The structure chart also indicates the data interfaces between modules, by showing what information flows from a higher level module down to a lower level module and vice-versa.

Another vital part of the structured design is called the "data dictionary."

How many times have you found that you used the same variable name to mean two different things in a program, or used different names for the same piece of data? Imagine the problems that can arise when several people are working on parts of the same program, with no rules as to what to name variables or files, or how large arrays should be, or what type of data each variable represents. Things can get out of control very quickly. It has happened to me, folks, and it could happen to you.

The data dictionary is a repository of definitions of all of the elements in the design, including data flows, data stores, processes, and terminators. It can avoid the redundancy of using different names for the same piece of data, errors and inconsistencies in variable names and characteristics, communication problems among programmers, and other related bums. Of course, a data dictionary isn't much good if it's incomplete, inaccurate, or ignored. Figure 3 shows what some data dictionary entries might look like using one of the common notations (DeMarco's); the details aren't important here.

Now, we are almost ready to write the program modules themselves. But not quite. Our data flow diagrams tell us what information flows into each module and what flows out, but they don't tell us how to accomplish the transformation. The missing piece is a detailed "process specification," or "minispec." This design element describes exactly what goes on inside the module, in enough detail that now, finally, at long last, you can write a computer program in the language of your choice.

Process specifications can take many different forms. You have probably written them before, using some kind of "pseudocode" or "structured English" to write out in words exactly what you want the program to do. Figure 4 shows a process spec written in one flavor of pseudocode. Various other diagramming techniques can also be used, such as action diagrams, Warnier-Orr diagrams, and others. It's often a short journey from pseudocode or action diagram to actual code in a real computer language. In fact, some of the available computer-based diagramming tools will actually produce compilable source code from the diagrams, but such code generators are still pretty much a thing of the future.

This seems like an awful lot of junk to keep track of, just to write a computer program. Wait! Let's let the computer do the grunt work. Yeah, that's

the ticket!

The Case for CASE

The latest items for your software engineering toolbox are CASE, or computer-aided software engineering tools. On one level, these are simply graphic editors, usually intended to be used on powerful microcomputer workstations such as the IBM PC/AT or Apple Macintosh. (Did someone say "ST?" Nope, not yet. Sorry.) Using a mouse, data flow diagrams, structure charts, and other diagrams used in structured systems analysis and design can be drawn.

But CASE tools are much more than simple drawing packages. Any CASE program worth its salt can make sure that your DFDs and structure charts are valid, in that they conform to the rules of some established diagramming methodology. Of course, it's the user's responsibility to heed the error messages and correct the diagram. You could always ignore the errors, but your aircraft carrier will look pretty funny if it's missing the flight deck due to a design flaw. Does the word "unemployment" mean anything to you?

CASE tools can integrate the data dictionary with the diagrams, automatically creating entries in the data dictionary when the DFDs are validated. They can detect inconsistencies in the way items in the dictionary are used or defined. They can make sure that each

DFD is consistent with the related DFDs at higher and lower levels. They can verify that the data interfaces defined between modules in your structure charts are consistent. Some CASE packages can validate the structured English or action diagrams used in creating your process specifications. The overall goal is to reduce errors by ensuring that all the necessary parts of your design are present, properly defined, and properly connected. Doesn't this seem reasonable?

Summary

I hope you don't feel that all this talk about engineering is taking the fun out of computer programming; it doesn't have to. Most of us will always be hackers at heart, and we'll still spend time at the keyboard just typing things in and seeing what happens. That's the artist in each of us, fighting to get out and express himself electronically.

The main application for software engineering ideas, methodologies, and tools is in the world of larger-scale (usually commercial) software development, where projects tend to be increasingly complex and time-consuming. I think you'll agree that a free-form approach to serious program development isn't likely to be as successful as a more disciplined, systematic approach, particularly when several (or many) people are involved. Contemporary software products are ex-

pensive, they stick around for a long time, and they undergo continual revision and enhancement.

As an example, I'm writing this article using ST Writer version 2.0. How many versions of this popular program has the good Dr. Noonan released to the Atari world? If Dr. Noonan didn't receive a structured design for ST Writer from the original programmers, I suspect he wishes he had by now. Software engineering can help prevent your program from ultimately becoming a crazy quilt of patch on kluge on repair on modification on correction.

Bibliography

There are many excellent books available on software engineering, structured design and development, and related topics. Here are a few of the best.

1. R. S. Pressman, *Software Engineering: A Practitioner's Approach*, 2nd Edition, McGraw-Hill, 1987.
2. R. S. Pressman, *Making Software Engineering Happen*, Prentice-Hall, 1988.
3. T. DeMarco, *Structured Analysis and System Specification*, Prentice-Hall, 1979.
4. J. Martin and C. McClure, *Structured Techniques for Computing*, Prentice-Hall, 1985.
5. J. Martin and C. McClure, *Software Maintenance*, Prentice-Hall, 1983.

Figure 4. Sample Process Specification in Pseudocode

Procedure: Find Equation

```
Compare first compound number in equation with first compound number
  for all equations in set
  If no match
    then call Error-Routine
  else if match then do
    compare other compound numbers in equation with those in
    entry matched
    if no match
      then call Error-Routine
    else if match
      then do
        get reaction number for matched entry
        if reaction number already complete
          then do
            print "Already Done" message
            make error sound
            reset screen
          end do
        end if
      end do
    end if
  end do
end if
return
```

Procedure: Error-Routine

```
print "Incorrect Reaction"
make error sound
reduce score by 5 points
reset screen
return
```

Frankendrive

The Hard Drive Construction Project

by Andy Eddy

Since purchasing my ST, I've wanted to get a hard drive to make storage easier and neater, provide faster access to the growing list of software I use, and to help eliminate the desert of floppies that surrounds my computer "office." The latter threatens to spread into my family's precious living space, but, unfortunately, hard drive cost was a major limitation regardless of the overall benefits it would bring.

One thing in my favor—and we've said it before—is that as computers become more popular, the equipment that was once at a premium (like modems and hard drives) becomes more commonplace. The driving force behind this movement comes from mass production pushing the prices down; and that situation had taken hard drive prices lower, but not enough for my liking.

Still another method that people are using to lower costs is through do-it-yourself construction—an option that is becoming especially popular in the IBM-compatible world—giving the user the freedom to configure a system to his or her own specifications and budget.

Unfortunately, I never thought home construction of a hard drive system was a viable alternative for the ST; that is, until I heard about a company called Berkeley Microsystems (360 Oakland Avenue, Suite 5, Oakland CA 94611; (415) 465-6956). They've advertised in ST magazines and shared booths at some Atari Faires, demonstrating their method of using readily-available, IBM-compatible parts in union with their interface card to build an inexpensive alternative to the over-the-counter commercial drives.

I've done my share of Heathkit projects in the past and resolved that this might be the answer to my prayers: a low-cost way of getting that hard drive. So I contacted Berkeley for more information, and they sent me a copy of their manual, as well as some of the previous press they've received. With this glimmer of hope (and potential saving of greenbacks), the dream of the **Frankendrive** sprang into action.

Looking through the documentation, I figured it would be pretty easy to pull the job off. The manual lists (to assist in the configuration process once your construction is completed) various drives that match the parameters of their product, which is basically any ST506/ST412-compatible drive. If you

are at all confused by what I just said, then I recommend you take some time to shake hands with Tom Harker's **Hard Disk Primer**, from the May 1987 *ANALOG*. It'll be invaluable in providing answers to basic questions that may arise.

Igor, get me a screwdriver . . .

Now, on to gathering the parts for the **Frankendrive**. I decided to purchase new products in order to ease my fear of buying pre-owned materials; this strategy goes well for some, but my poor luck in the past with used cars backed up my plan. Scouring a couple of issues of *Infoworld* and *Computer Shopper*—the latter a magazine renowned for its wealth of hardware distributor ads—I came up with lots of

Spreading all the parts out in front of me, I figured it would take very little time to finish Frankendrive

pricing data, as well as some leads for the purchases themselves.

What I managed to do is purchase a Seagate ST238 (a 30MB RLL drive), an IBM-like case for housing the equipment, and an IBM-like 150-watt switched power supply to drive it all. The power supply mounts in the case perfectly (which comes with pre-drilled screw holes and a custom cutaway for the power switch), and also has a built-in fan for pulling the generated heat out of the enclosure. Some would say that a case that size and 150 watts are overkill, but I'll address those comments later.

Add to this the Berkeley interface card and Adaptec 4070 RLL hard disk controller, both provided by Berkeley (the RLL combo costs \$300, while the



MFM set-up goes for \$250). See Table 1 for the pricing breakdown, which doesn't include incidental shipping costs; unfortunately, there are no bargain computer shops near me, so mail-order was a necessity. And to make it all the more satisfying, the Berkeley unit comes with a built-in clock for setting the system clock. When you consider that you'd have to kick out an additional \$50 or so for that item elsewhere (excepting Mega owners, who have their clock built-in), you can call that a further savings over most commercial hard drives.

Spreading all the parts out in front of me, I figured it would take very little time to finish the **Frankendrive**. A bit hasty in my optimism, I followed the somewhat thin directions in the Berkeley manual, taking the interface and mounting it on top of the controller and combining them into one board set. Even though it isn't recommended—I found out later that this is for reasons of RF crosstalk, an interference which could cause excessive read errors—I also mounted the board combo on the hard disk to make one compact unit. My problem came from not being able to mount this lump of electronics inside the case properly. Placing it one way would interfere with the closing of the case cover; placing it upside-down would make mounting it with the existing screw holes nearly impossible. Time for a reassessment of my initial strategy.

As you might have noticed, I was winging it from the start, but with so much room in the case, I didn't figure it would be a problem (see The Lesson Learned). So I removed the board set from the drive and aimed for mounting them side-by-side. Everything fell together smoothly; the screw holes in the boards and drive matched the case, so I socked them down. When I attempted to plug all the required cables together, the data cable that ran from the boards to the drive came up a little short. Once again, I disassembled the whole thing and restarted.

Still reasonably clearheaded though, I followed that disappointment by switching the locations of the drive and board set, then placed the power supply in its proper location at the right rear where the switch cutaway is. Again, I tightened them all in place. And once again, The Lesson Learned came back to nudge me—but I hadn't picked up on it yet. On one hand, the

TABLE 1 -- COST BREAKDOWN

BERKELEY PACKAGE (RLL)	\$300
30 MB HARD DISK DRIVE	\$265
CASE	\$ 24
POWER SUPPLY	\$ 40
EXTRA POWER CABLE	\$ 7
<hr/>	
TOTAL COST	\$636

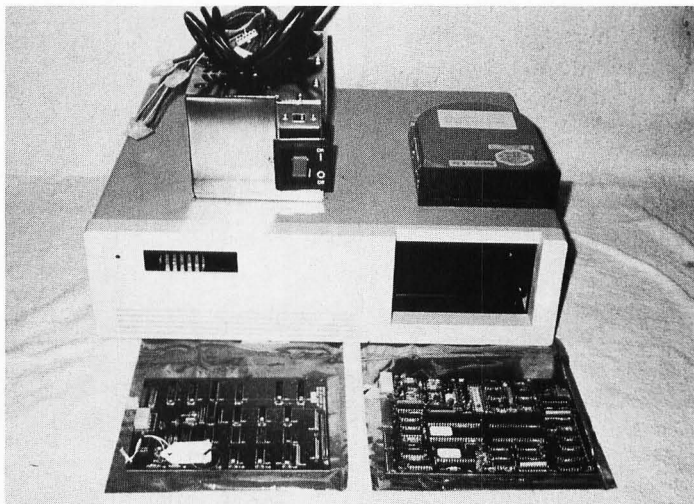


FIGURE 2

power supply cables that provide juice to the controller and drive reached fine; dishearteningly though, the cable to the Berkeley board came close enough to touch its mate, but wouldn't *s-t-r-e-t-c-h* sufficiently to connect.

In wimpy frustration, I gave a call to Berkeley for suggestions. The fellow I reached, Vance Chin, told me the best thing I could do at that point was to hook everything up and be sure it worked before going through the

sockets and insulating circuit boards from shorting on other metal parts. Then came the first crucial test: powering up the system, *apart* from the ST first...just in case. Flipping on the power switch didn't bring on a shower of sparks or flames, and rewarded me with the whirring of the drive coming up to speed. LEDs blinked and then dimmed as the **Frankendrive** signaled its readiness for action.

It's Alive...

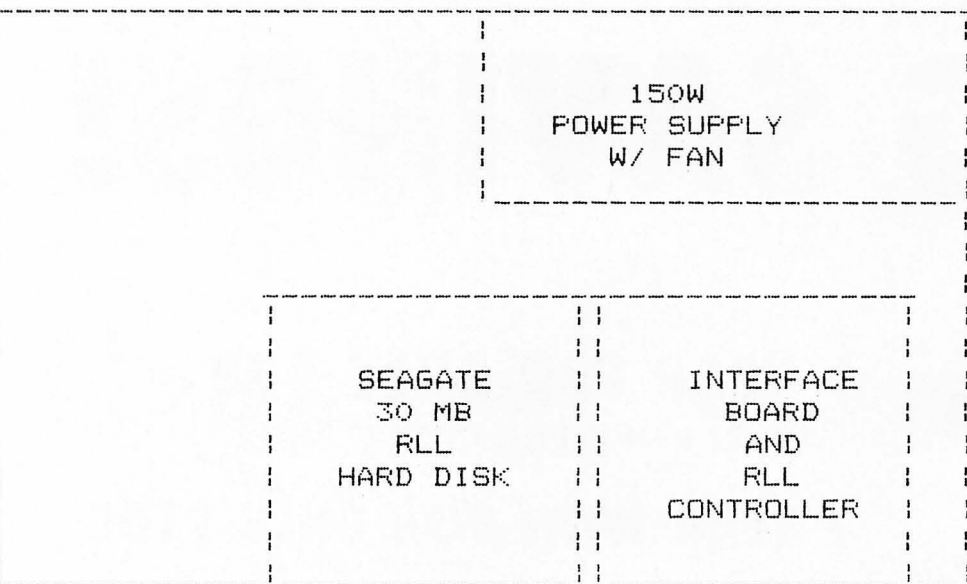
This brought on the *really* crucial test: hooking it up to the ST and seeing if the transplant would be successful or rejected. I first took the DMA cable (the length of which I will discuss shortly) and plugged it into the hard drive port on the rear panel of the ST, then repowered the **Frankendrive** as before. After it went through its routine, I gingerly placed my finger on the ST's switch and rocked it on. Like the expectant father who is told his wife has just given birth, I was blessed with a bouncing baby bundle of bytes—though similarly undiapered and untrained, so to speak.

For this reason, the next procedure was to configure the drive for interac-

Igor, get me a screwdriver!

process of anchoring the parts and risking further annoyance by having to remove them *again* should there be a problem (other than operator error). This is when The Lesson Learned finally sunk in and became reality: *Don't screw down the parts until you are sure that their location is permanent.*

After purchasing an extender power cable, I took Vance up on his idea, hooking all the cables in their proper



INTERNAL MOUNTING OF EQUIPMENT WITHIN THE CASE.

tion with the computer system, and the Berkeley package comes with a floppy for taking care of all the processes you need to undertake before putting your hard drive to work. This software lets you set up your system for the specific drive you are using, so it's prepared with such facts as number of cylinders, heads, step pulse rate, and other pertinent figures. Using a text editor that pumps out ASCII text, it's easy to alter their configuration file to match your disk's parameters (using the previously-mentioned table in the manual for assistance).

Once that simple process was completed, it was on to the formatting and partitioning process. Just like a floppy, the hard disk needs to be formatted to break it into smaller blocks for the ST to easily digest. Given that your drive is quite sizable, you must also partition it into several smaller drives to allow quick access time to all your data. Due to GEM limitations, the ST can address a partition no larger than 16MB; I have mine divided into four drives: two of 10MB each, and two of 5MB each. This process also marks any bad sectors you may have on your disk, so the system doesn't try to store data to them. It's a

lengthy procedure, but necessary for the sure use of the hard drive.

As you can see from Figure 2, my drive is together in the sense that I can move it without the worry of parts falling off, but I still need to make the **Frankendrive** more permanent. The main problem I face is with the routing of the DMA cable, currently snaking out the front disk drive opening. Berkeley provides a cable that is fairly short—they claim that this is to meet Atari specifications and assure quality data transfer over the DMA line—but for an installation of this nature, you'll wish there was a better way to handle it. If I can't get a longer cable (Supra is reportedly shipping longer cables with their drives), I'll drill a hole in the front of the case for the DMA cable, which will also give a certain amount of strain relief for the cable's connectors.

The Bottom Line

Again referring to Table 1, the final cost of my **Frankendrive** was about \$635, as opposed to the cheapest commercial unit that I've seen advertised in *ST-Log*, which was listed at just over \$750. I figure the money I saved was adequate payment for the aggravation I caused

myself and the phone calls I made. Also, the pleasure of doing it myself is something to keep in mind, but pride and accomplishment can't be measured in the same sense as cold hard cash.

The best thing about building the **Frankendrive** is how much is left open for expansion, as there is plenty of empty space remaining. Chin suggests that a second ST floppy drive can be mounted in the open space, making an especially compact unit for transport. Similarly, someone who uses an IBM-compatible, 5 1/4-inch drive (such as with PC-Ditto) could put that in instead, or as an added bonus.

My plan is to later add a second hard drive unit, as the Adaptec 4070 controller can address two drives. This way I can double my storage for very little expense: the cost of another drive and associated cables, or about \$290 to double my storage capacity to 60MB. This provides an even more substantial savings over the high-capacity commercial systems that are available; my 60MB **Frankendrive** would end up costing approximately \$925, far less than commercial units.

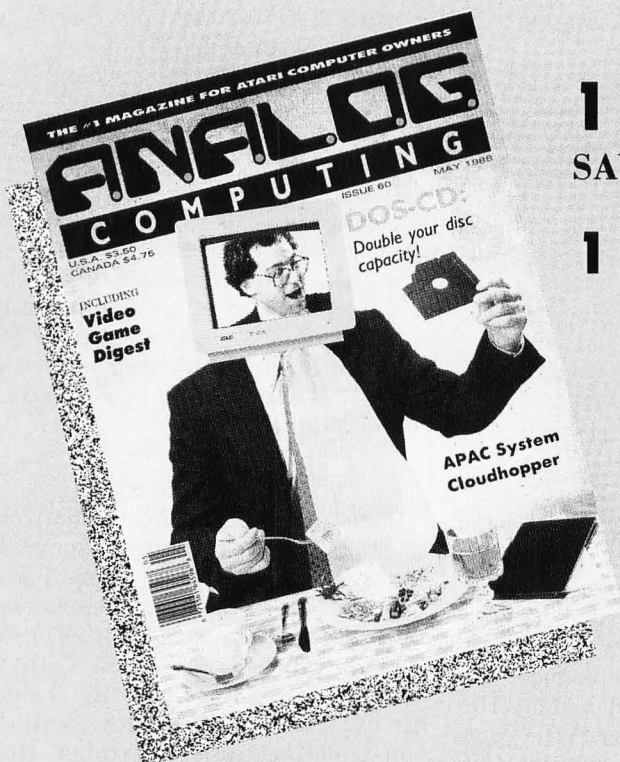
Final Note

Shortly after ordering all the parts for the **Frankendrive**, I spoke to some other ST users on Delphi who had realized even more savings over mine. Clarence Chang (CTCHANG) and Wayne Dunham (WAYNEDUNHAM) purchased Tulin 30MB RLL drives directly from the company for their construction projects, which they managed to find for about \$225. That brings up the importance of shopping around before you buy. I searched for some time to find what I found, and even then there were better ways to cut costs. Supra and ICD, two established providers to Atari users, also introduced their versions of the interface building block to give you some choice in creating your own **Frankendrive**.

Most importantly, be careful not to deal with companies that you or other users aren't too sure of. Mail order buying can be risky, so check into each firm that you plan to buy through, so you don't get burned.

The author resides in Connecticut, and is a frequent contributor to *ANALOG* and *ST-LOG*. He can be reached on Delphi and GENie as KIDX; on People/Link as KID X; and Compuserve at 72327,503.

BOOT UP TO BIG SAVINGS!



1 YEAR FOR ONLY \$28

SAVE \$14 OFF THE COVER PRICE

1 YEAR WITH DISK ONLY \$105

**SAVE TIME AND MONEY
SUBSCRIBE TO ANALOG**

**SAVE \$14 OFF THE
COVER PRICE WITH
THE CONVENIENCE
OF HAVING ANALOG
DELIVERED DIRECT-
LY TO YOUR DOOR
BEFORE IT EVEN HITS
THE NEWSSTANDS.
GET THE MOST OUT
OF YOUR COMPUTER.**

**SUBSCRIBE TO
ANALOG
TODAY**

☐ 1 YEAR @ \$28 — SAVE \$14!

MCGWY

FOREIGN — ADD \$7 PER YEAR

☐ 1 YEAR WITH DISK @ \$105

DCGWY

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 _____

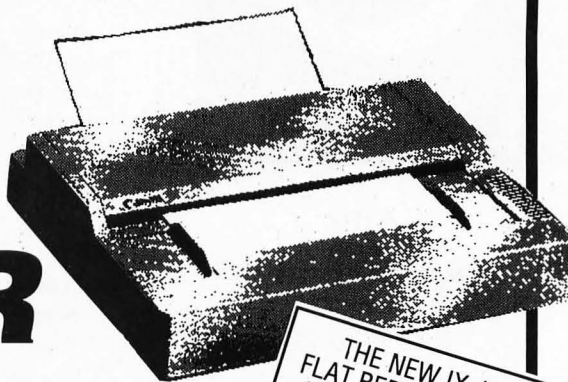
ZIP _____

MAKE CHECK PAYABLE TO L.F.P., INC., P.O. Box 16927, N. Hollywood, CA 91615. Offer expires August 31, 1988. Your first issue will arrive in 6 TO 8 weeks.

WATCH FOR IT!

ANOTHER SUPERIOR PRODUCT FROM NAVARONE

WHEN YOUR IMAGE IS AT STAKE, CHOOSE THE
**ST SCAN
IMAGE
SCANNER**



THE NEW IX-12F
FLAT BED SCANNER IS
FINALLY AVAILABLE.
CALL FOR DETAILS NOW!

FOR YOUR ATARI ST SYSTEM.

The flexibility
to introduce
art into desktop
publishing.



With the *ST SCAN Image Scanner* you can transfer your line art, photographs, logos, diagrams, text, and other graphics into your computer.

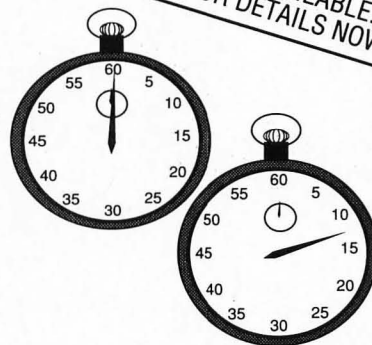
Capture your *image* sharp and clear with resolutions up to 300 dots per inch and with 32 shades of grey.

Navarone's high speed interface.

Navarone combines the Canon IX-12™ Image Scanner with it's own High Speed Interface that plugs into the cartridge port of the Atari ST or Mega™.

Sophisticated software is provided to allow scanning in both line art mode and halftone mode. The *ST SCAN Image Scanner* program is very easy to use and operates under GEM™ with simple click-on selections.

It takes less
than 15
seconds to scan
in your image.
Once digitized, you
can use graphic
programs like
DEGAS™ to edit,
crop, size or shape
your *image*.



You can put your image into final documents with Publishing Partner™ or save in Postscript to allow direct printing on postscript devices, such as the Linotronic 300™, Apple LaserWriter™, or QMS PS 800™.

Compatibility with graphic programs:

Fleet Street Publisher™ by Mirrorsoft, Publishing Partner™ by Softlogik, Easy Draw™ by Mi-graph, or DEGAS™ by Batteries Included.

The *ST SCAN Image Scanner* comes complete with scanner, interface, cable, software and manual for only:

\$ 1,239.00

To Order: Call our toll free number or send M.O. plus shipping (call for rates). VISA, MC, C.O.D. welcome. California residents add 7% sales tax.

NAVARONE

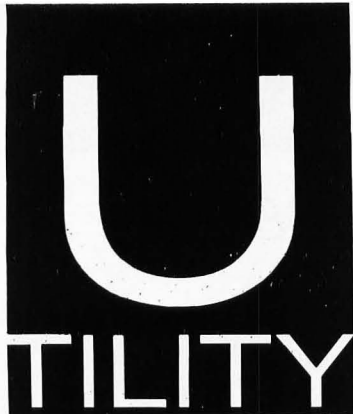


1-800-624-6545 (Nationwide)
Or (408) 378-8177 (California)

NAVARONE INDUSTRIES, INC. • 454 Kenneth Avenue • Campbell, CA 95008

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

CIRCLE #109 ON READER SERVICE CARD.



Any Resolution

by Matthew J.W. Ratcliff

Illum, let's log on to Gateway BBS and
see what's cooking.
ATDT 314-647-3290
CONNECT

Welcome to Gateway BBS.
PASSWORD: xxxx
Enter the last four digits of your phone
number: xxxx
Last time on: 7/20/87 3:00AM
Logon: 7/21/87 3:30AM

GO> z;2;r;n

Welcome to Ratty's Rap; you have 100
messages waiting.
MSG: 29567 Date: 7/21/87
From: Jim* Gateway
To: Mat*Rat
Subj: Where's the STUFF

Hey, Mat, Where are all the articles for the
August ACE Newslite? You'd better get to
work pronto, if you want to remain
Sysop. I made you, I can BREAK you!
<GRIN>

Big JIM. . . .

Oh, wow, I can see this is going to be
a long session: 100 new messages and
then I've got to get all those articles done.
Well, it's time for a Bud break before get-
ting too deeply into this. A quick dash to
the fridge, load up another video game for
Nathan, change Charlie, kiss Nancy and
assure her I'm still alive, turn the steaks
on the grill and take a restroom break.
Ok, back to the computer:

.....
Input too slow. Timed out. Goodbye,
Mat*Rat, thanks for calling. <CLICK>

CARRIER LOST

Oh, drat!
ATDT 314-647-3290
BUSY

If you are active in telecommunica-
tions, as I am, you may find the above a
rather typical access session. Most bulletin
board systems (BBS) have an automatic
timeout feature, and if you don't enter
something at your keyboard within one
minute or so you are automatically logged
off. Commercial systems such as Delphi
or CompuServe are more forgiving, but
you pay the price, of course.

Now all you need is the **Busy Buddy**
accessory. To create your copy of **Busy
Buddy**, type in Listing 1 (checking your
work with ST-check) and run it from ST
BASIC. Install BUSYBUD.ACC on your



telecommunications boot disk, and you
are in business. Whenever you need to
take a break, just pull down the desktop
menu (which is accessible from most
communications programs now) and
click on **Busy Buddy**. When activated
an alert box will come up, prompting you
for backspace type. Normally an ASCII 8
is required, but some systems expect a de-
lete character, ASCII 127. If you are call-
ing 8-bit Atari boards using ATASCII emu-
lation, then you should click on the mid-
dle selection, ASCII 126, what the 8-bit
uses for a backspace (the ST and other
computers use ASCII 126 as a tilde). Click
on the appropriate one (or just press
RETURN to accept ASCII 8).

The next alert box is a prompt for the
maximum time limit that **Busy Buddy**
should run unattended, five, 15 or 60
minutes. This feature will save your neck
if you put **Busy Buddy** to work on a pay
connect system, or a long-distance BBS.
Since Buddy will time out for you even-
tually, you'll get logged off the BBS soon-
er or later if you completely forget about
it. When the Buddy times out, another
alert box will pop up. It gives you the op-
tion of restarting (put in a new time limit),
or exit.

Once **Busy Buddy** is active, you will
see the send and receive lights on your
modem pulsate about once every two se-
conds. Buddy is sending a space and then
a backup (either backspace or delete)
character once every half second. This
will keep the system at the other end
from timing out and hanging up before
you've used all your access time.

With **Busy Buddy** running, you can go
to the Flash or Interlink edit buffer and
compose your messages, for example. Or
take a seventh-inning stretch and fetch
a cold brew. When it's time to get back
to modeming, just pull down the desktop
menu and click on **Busy Buddy** again.

u d d y

The timer routine will be shut off, and a reminder displayed to that effect.

I've been testing **Busy Buddy** thoroughly, with no problems, on Delphi, ST Forem (AURA in St. Louis) and 8-bit Forem (Gateway BBS). One real plus is that **Busy Buddy** is automatically disabled whenever you exit the menu screen to the terminal display. Buddy is active when GEM is active, in the edit windows, menus, and file selectors of these terminal programs. You should *never* attempt a file transfer without disabling **Busy Buddy** first, however. When you initiate the transfer, a file selector (GEM window) pops up, and Buddy is enabled once again. It truly confuses any Xmodem transfer. Simply go back to the **Busy Buddy** entry under the Desk menu and click it off, then all your file transfers will work smoothly.

This utility is extremely useful when it comes to message entry on BBS and Telecommunication services. While Buddy does his thing keeping the Forem board I'm connected to busy, I can use full screen editing in my Flash or Interlink buffer to generate a reply.

Bud takes a clicking and keeps on ticking too. I used Interlink to connect with

Gateway, began to enter a message, and then enabled **Busy Buddy**. Once on, I used Interlink's EXECUTE feature to load and run Flash. In Flash I created my reply, and then used ASCII upload to send

the message. All the while, except during some file I/O, **Busy Buddy** kept on running. You will find this utility helpful when you need to exit your terminal software to format disks (something Flash doesn't do), or perform other file maintenance functions.

This certainly isn't the most powerful accessory you will ever use, but it can come in quite handy if you spend a lot of time on the modem. Study of the heavily commented listing will show you how to write an accessory in C. Written with Megamax, the only other "special" thing required is linking this file with ACC.L (the ACC.L must be *first* in the link list), which comes with the compiler package. (Note the use of the external global variable **gl_apid**, vital information for your accessory.)

The **evnt_multi** function is used instead of the usual **evnt_msg** so that either the accessory open message or a two-second timed interrupt "event" can activate **Busy Buddy**. The state of the flag variable tells the program to activate, deactivate, or continue the busy signal process in **Busy Buddy**.

The next time you are on line, **Busy Buddy** will be there to help you out.

**Kiss Nancy
and assure
her I'm still
alive, turn
the steaks on
the grill and
take a
restroom
break.**

Listing 1: ST Basic

```
100 filename$="a:\BUSYBUD.ACC"
110 fullw 2:clearw 2:gotoxy 0,0:print
"creating file..."
120 option base 0
125 dim ax(16000):def seg=1:v$=""
130 p=varptr(ax(0)):bptr=p+1
140 for ix=1 to 4354
150 read v$:code%=val("&H"+v$)
160 poke p, code%:print ". ";
170 p=p+1
180 next
190 bsave filename$,bptr,4354
```

```
200 print "file written":end
1000 data 60,1A,00,00,10,BC,00,00,00,2
4,00,00,18,C4,00,00
1010 data 00,00,00,00,00,00,00,00,0
0,00,00,4E,F9,00,00
1020 data 00,0C,4E,F9,00,00,0C,86,4B,F
A,FF,FE,0C,6D,4E,F9
1030 data FF,FA,66,04,5D,8D,60,F4,9B,F
C,00,00,01,00,20,6D
1040 data 00,18,22,6D,00,18,D3,ED,00,1
C,20,2D,00,14,53,80
1050 data 6F,06,13,20,51,C8,FF,FC,20,6
D,00,10,20,2D,00,1C
1060 data 53,80,42,18,51,C8,FF,FC,28,6
D,00,10,D9,ED,00,1C
1070 data 29,4D,FF,FC,4F,EC,FF,FC,2A,6
D,00,08,4E,AD,00,06
```

1080 data 4E,BA,00,2A,3F,3C,00,00,4E,B
 A,06,C6,48,E7,FF,FE
 1090 data 3F,3C,00,41,3F,3C,00,02,4E,4
 1,58,8F,20,3C,00,0F
 1100 data 42,40,53,80,66,FC,4C,DF,7F,F
 F,4E,75,4E,56,FF,E6
 1110 data 4E,BA,05,0E,48,6C,00,00,3F,2
 C,EE,7A,4E,BA,06,02
 1120 data 5C,8F,3D,40,FF,FE,3D,7C,00,1
 0,FF,E8,3D,7C,00,01
 1130 data FF,E6,60,00,00,E8,42,6E,FF,E
 C,48,6E,FF,EC,48,6E
 1140 data FF,EC,48,6E,FF,EC,48,6E,FF,E
 C,48,6E,FF,EC,48,6E
 1150 data FF,EC,42,67,3F,3C,07,D0,48,6
 E,FF,EE,3F,2E,FF,EC
 1160 data 3F,2E,FF,EC,3F,2E,FF,EC,3F,2
 E,FF,EC,3F,2E,FF,EC
 1170 data 3F,2E,FF,EC,3F,2E,FF,EC,3F,2
 E,FF,EC,3F,2E,FF,EC
 1180 data 3F,2E,FF,EC,3F,3C,FF,FF,3F,3
 C,FF,FF,3F,3C,FF,FF
 1190 data 3F,2E,FF,E8,4E,BA,04,DE,DE,F
 C,00,3C,3D,40,FF,EA
 1200 data 30,2E,FF,EA,C0,7C,00,10,67,0
 0,00,44,0C,6E,00,28
 1210 data FF,EE,66,00,00,3A,30,2E,FF,F
 6,B0,6E,FF,FE,66,00
 1220 data 00,2E,30,2E,FF,E8,C0,7C,00,2
 0,67,00,00,14,3D,7C
 1230 data 00,10,FF,E8,4E,BA,00,E4,42,6
 E,FF,EA,60,00,00,10
 1240 data 3D,7C,00,30,FF,E8,4E,BA,00,4
 2,42,6E,FF,EA,30,2E
 1250 data FF,EA,C0,7C,00,20,67,00,00,2
 4,30,2E,FF,E8,C0,7C
 1260 data 00,20,67,00,00,18,4E,BA,00,E
 8,0C,6C,00,00,EF,08
 1270 data 6E,00,00,0A,4E,BA,01,30,3D,4
 0,FF,E8,30,2E,FF,E6
 1280 data 66,00,FF,14,4E,5E,4E,75,4E,7
 5,4E,56,FF,FE,48,6C
 1290 data EF,B4,3F,3C,00,01,4E,BA,05,0
 6,5C,8F,3D,40,FF,FE
 1300 data 30,2E,FF,FE,60,00,00,02,B0,7
 C,00,01,67,00,00,06
 1310 data 60,00,00,10,19,7C,00,08,EF,0
 A,60,00,00,46,60,00
 1320 data 00,0E,B0,7C,00,02,67,00,00,0
 6,60,00,00,10,19,7C
 1330 data 00,7E,EF,0A,60,00,00,2C,60,0
 0,00,0E,B0,7C,00,03
 1340 data 67,00,00,06,60,00,00,18,19,7
 C,00,7F,EF,0A,60,00
 1350 data 00,12,60,00,00,02,60,00,00,0
 A,60,00,00,06,60,00
 1360 data FF,F6,4E,BA,01,22,3F,3C,00,2
 0,4E,BA,00,7C,54,8F
 1370 data 19,6C,EF,0A,EF,0C,4E,5E,4E,7
 5,4E,56,00,00,10,2C
 1380 data EF,0C,12,2C,EF,0A,48,81,48,8
 0,80,41,66,00,00,10
 1390 data 10,2C,EF,0C,48,80,3F,00,4E,B
 A,00,4E,54,8F,48,6C
 1400 data EF,0C,3F,3C,00,01,4E,BA,04,5
 6,5C,8F,4E,5E,4E,75
 1410 data 4E,56,00,00,10,2C,EF,0C,48,8
 0,3F,00,4E,BA,00,2A
 1420 data 54,8F,10,2C,EF,0C,48,80,B0,7
 C,00,20,66,00,00,0C
 1430 data 19,6C,EF,0A,EF,0C,60,00,00,0
 8,19,7C,00,20,EF,0C
 1440 data 53,6C,EF,08,4E,5E,4E,75,4E,5
 6,00,00,10,2E,00,09
 1450 data 48,80,3F,00,3F,3C,00,01,3F,3
 C,00,03,4E,BA,07,5C
 1460 data 5C,8F,4E,5E,4E,75,4E,56,FF,F
 C,10,2C,EF,0C,12,2C

1470 data EF,0A,48,81,48,80,B0,41,66,0
 0,00,10,10,2C,EF,0C
 1480 data 48,80,3F,00,4E,BA,FF,C2,54,8
 F,3F,3C,00,07,3F,3C
 1490 data 00,02,3F,3C,00,03,4E,BA,07,2
 2,5C,8F,48,6C,EF,0E
 1500 data 3F,3C,00,01,4E,BA,03,B8,5C,8
 F,3D,40,FF,FE,0C,6E
 1510 data 00,01,FF,FE,66,00,00,1A,3D,7
 C,00,10,FF,FC,48,6C
 1520 data EF,0C,3F,3C,00,01,4E,BA,03,9
 6,5C,8F,60,00,00,0C
 1530 data 3D,7C,00,30,FF,FC,4E,BA,00,0
 E,30,2E,FF,FC,60,00
 1540 data 00,02,4E,5E,4E,75,4E,56,FF,F
 E,48,6C,EF,48,3F,3C
 1550 data 00,03,4E,BA,03,6A,5C,8F,3D,4
 0,FF,FE,30,2E,FF,FE
 1560 data 60,00,00,02,B0,7C,00,01,67,0
 0,00,06,60,00,00,10
 1570 data 39,7C,00,96,EF,08,60,00,00,4
 6,60,00,00,0E,B0,7C
 1580 data 00,02,67,00,00,06,60,00,00,1
 0,39,7C,01,C2,EF,08
 1590 data 60,00,00,2C,60,00,00,0E,B0,7
 C,00,03,67,00,00,06
 1600 data 60,00,00,18,39,7C,07,08,EF,0
 8,60,00,00,12,60,00
 1610 data 00,02,60,00,00,0A,60,00,00,0
 6,60,00,FF,F6,4E,5E
 1620 data 4E,75,29,49,E7,44,29,4A,E7,4
 0,22,2F,00,04,30,3C
 1630 data 00,C8,4E,42,22,6C,E7,44,24,6
 C,E7,40,4E,75,00,01
 1640 data 00,02,01,01,01,01,01,01,01,0
 1,02,01,01,01,01,01,01,01,01,01,0
 1650 data 00,00,00,00,00,00,00,00,00,0
 0,01,00,00,01,00,03
 1660 data 05,00,05,05,05,00,00,01,01,02,0
 1,00,10,07,01,02,01
 1670 data 00,00,00,00,00,00,00,00,00,0
 0,01,01,01,02,01,01
 1680 data 02,01,01,01,02,01,01,01,01,02,0
 1,01,01,00,00,00,00
 1690 data 00,00,00,00,00,00,00,00,02,0
 1,01,01,01,01,06,01
 1700 data 01,04,01,01,01,01,03,01,02,01,0
 1,04,02,01,08,01,01
 1710 data 00,00,00,00,00,00,01,01,01,0
 9,01,01,01,01,01,01
 1720 data 01,00,00,05,01,03,03,01,02,0
 2,01,00,00,00,00,00
 1730 data 00,00,00,00,00,00,00,00,00,0
 0,00,00,00,00,00,00
 1740 data 00,00,00,00,00,00,00,00,00,0
 0,00,00,00,00,00,00
 1750 data 00,00,04,03,00,08,03,00,06,0
 1,00,08,01,00,08,01
 1760 data 00,04,01,01,03,01,01,00,05,0
 0,01,01,01,00,05,00
 1770 data 00,01,01,00,01,01,00,00,00,0
 0,00,00,00,00,00,00
 1780 data 00,00,00,00,00,00,00,00,00,0
 0,00,00,00,00,00,02
 1790 data 02,00,00,00,00,00,00,00,00,0
 0,00,00,00,00,00,00
 1800 data 00,00,00,00,00,00,00,00,00,0
 0,00,00,05,01,00,05
 1810 data 01,00,01,01,00,01,01,00,02,0
 5,00,06,01,00,02,01
 1820 data 00,01,01,00,06,05,00,00,00,0
 0,00,01,01,00,01,00
 1830 data 02,01,00,02,01,01,01,01,01,0
 0,00,00,00,00,00,00
 1840 data 00,00,00,00,00,00,00,00,00,0
 1,02,03,01,02,01,01
 1850 data 01,01,01,01,00,01,01,00,01,0
 2,4E,56,FF,FA,39,6E

1860 data 00,08,EE,FE,30,2E,00,08,90,7
 C,00,0A,C1,FC,00,03
 1870 data 41,FA,FE,8C,22,08,20,41,D0,C
 0,41,D0,2D,48,FF,FA
 1880 data 3D,7C,00,01,FF,FE,60,1E,20,6
 E,FF,FA,52,AE,FF,FA
 1890 data 10,10,32,2E,FF,FE,E3,81,41,E
 C,EE,FE,D0,C1,48,80
 1900 data 30,80,52,6E,FF,FE,0C,6E,00,0
 4,FF,FE,6D,DA,2F,2C
 1910 data EE,7C,4E,BA,FE,2E,58,8F,30,2
 C,EE,AC,4E,5E,4E,75
 1920 data 4E,56,FF,FE,41,EC,EE,FE,29,4
 8,EE,80,41,EC,EE,DE
 1930 data 29,48,EE,84,41,EC,EE,BC,29,4
 8,EE,88,41,EC,EE,AC
 1940 data 29,48,EE,8C,41,EC,EE,A0,29,4
 8,EE,90,41,EC,EE,98
 1950 data 29,48,EE,94,41,EC,EE,80,29,4
 8,EE,7C,3F,3C,00,0A
 1960 data 4E,BA,FF,58,54,8F,39,6C,EE,A
 C,EE,7A,30,2C,EE,7A
 1970 data 4E,5E,4E,75,4E,56,00,00,39,6
 E,00,08,EE,BC,39,6E
 1980 data 00,0A,EE,BE,39,6E,00,0C,EE,C
 0,39,6E,00,0E,EE,C2
 1990 data 39,6E,00,10,EE,C4,39,6E,00,1
 2,EE,C6,39,6E,00,14
 2000 data EE,C8,39,6E,00,16,EE,CA,39,6
 E,00,18,EE,CC,39,6E
 2010 data 00,1A,EE,CE,39,6E,00,1C,EE,D
 0,39,6E,00,1E,EE,D2
 2020 data 39,6E,00,20,EE,D4,39,6E,00,2
 2,EE,D6,29,6E,00,24
 2030 data EE,A0,39,6E,00,28,EE,D8,39,6
 E,00,2A,EE,DA,3F,3C
 2040 data 00,19,4E,BA,FE,D6,54,8F,20,6
 E,00,2C,30,AC,EE,AE
 2050 data 20,6E,00,30,30,AC,EE,B0,20,6
 E,00,34,30,AC,EE,B2
 2060 data 20,6E,00,38,30,AC,EE,B4,20,6
 E,00,3C,30,AC,EE,B6
 2070 data 20,6E,00,40,30,AC,EE,B8,30,2
 C,EE,AC,4E,5E,4E,75
 2080 data 4E,56,00,00,39,6E,00,08,EE,B
 C,29,6E,00,0A,EE,A0
 2090 data 3F,3C,00,23,4E,BA,FE,84,54,8
 F,4E,5E,4E,75,4E,56
 2100 data 00,00,39,6E,00,08,EE,BC,29,6
 E,00,0A,EE,A0,3F,3C
 2110 data 00,34,4E,BA,FE,66,54,8F,4E,5
 E,4E,75,4E,56,00,00
 2120 data 3F,2E,00,08,4E,BA,02,CA,54,8
 F,0C,6E,83,00,00,08
 2130 data 6E,06,30,3C,00,00,60,1E,3F,2
 E,00,08,3F,3C,00,3E
 2140 data 4E,BA,03,34,58,8F,39,40,E8,7
 8,67,06,30,3C,FF,FF
 2150 data 60,04,30,3C,00,00,4E,5E,4E,7
 5,4E,56,00,00,3F,2E
 2160 data 00,08,3F,3C,00,4C,4E,BA,03,0
 E,58,8F,4E,5E,4E,75
 2170 data 4E,56,FF,FE,2F,0B,41,EC,E8,C
 6,26,48,60,16,30,2B
 2180 data 00,0A,C0,7C,00,03,67,08,2F,0
 B,4E,BA,00,2A,58,8F
 2190 data D6,FC,00,14,20,0B,41,EC,E8,C
 6,22,08,D2,BC,00,00
 2200 data 05,B4,B0,81,6D,D8,3F,2E,00,0
 8,4E,BA,FF,AE,54,8F
 2210 data 26,5F,4E,5E,4E,75,4E,56,00,0
 0,2F,0B,26,6E,00,08
 2220 data 2F,0B,4E,BA,00,44,58,8F,4A,4
 0,67,06,30,3C,FF,FF
 2230 data 60,30,30,2B,00,0A,C0,7C,00,1
 0,67,0A,2F,2B,00,06
 2240 data 4E,BA,01,F8,58,8F,42,6B,00,0
 A,3F,2B,00,0C,4E,BA

2250 data FF,2C,54,8F,4A,40,67,06,30,3
 C,FF,FF,60,04,30,3C
 2260 data 00,00,26,5F,4E,5E,4E,75,4E,5
 6,00,00,48,E7,01,10
 2270 data 26,6E,00,08,30,2B,00,0A,C0,7
 C,00,03,66,08,30,3C
 2280 data FF,FF,60,00,00,98,2E,13,9E,A
 B,00,06,30,2B,00,0A
 2290 data C0,7C,00,80,67,56,30,2B,00,0
 A,C0,7C,00,02,66,06
 2300 data 30,3C,FF,FF,60,76,30,2B,00,0
 A,C0,7C,00,04,67,10
 2310 data 3F,3C,00,02,42,A7,3F,2B,00,0
 C,4E,BA,00,68,50,8F
 2320 data 3F,07,2F,2B,00,06,3F,2B,00,0
 C,4E,BA,02,CC,50,8F
 2330 data 0C,40,FF,FF,66,06,30,3C,FF,F
 F,60,40,02,6B,FF,7F
 2340 data 00,0A,30,07,48,C0,D1,AB,00,0
 E,60,24,0C,6B,00,00
 2350 data 00,0C,6F,1C,3F,3C,00,01,30,2
 B,00,04,44,40,48,C0
 2360 data 2F,00,3F,2B,00,0C,4E,BA,00,1
 C,50,8F,27,40,00,0E
 2370 data 26,AB,00,06,42,6B,00,04,30,3
 C,00,00,4C,DF,08,80
 2380 data 4E,5E,4E,75,4E,56,FF,F2,0C,6
 E,00,00,00,08,6C,0A
 2390 data 20,3C,FF,FF,FF,FF,60,00,00,F
 E,3F,2E,00,0E,3F,2E
 2400 data 00,08,2F,2E,00,0A,3F,3C,00,4
 2,4E,BA,01,8A,DE,FC
 2410 data 00,0A,2D,40,FF,FC,0C,80,00,0
 0,00,00,6D,08,20,2E
 2420 data FF,FC,60,00,00,D2,3F,3C,00,0
 1,3F,2E,00,08,42,A7
 2430 data 3F,3C,00,42,4E,BA,01,60,DE,F
 C,00,0A,2D,40,FF,F8
 2440 data 3F,3C,00,02,3F,2E,00,08,42,A
 7,3F,3C,00,42,4E,BA
 2450 data 01,46,DE,FC,00,0A,2D,40,FF,F
 4,0C,6E,00,01,00,0E
 2460 data 66,0E,20,2E,FF,F8,D0,AE,00,0
 A,2D,40,00,0A,60,24
 2470 data 0C,6E,00,02,00,0E,66,0E,20,2
 E,FF,F4,D0,AE,00,0A
 2480 data 2D,40,00,0A,60,0E,30,2E,00,0
 E,67,08,20,3C,FF,FF
 2490 data FF,FF,60,62,20,2E,00,0A,B0,A
 E,FF,F4,6F,1E,48,6E
 2500 data FF,F2,20,2E,00,0A,90,AE,FF,F
 4,2F,00,3F,2E,00,08
 2510 data 3F,3C,00,40,4E,BA,00,E0,DE,F
 C,00,0C,42,67,3F,2E
 2520 data 00,08,2F,2E,00,0A,3F,3C,00,4
 2,4E,BA,00,CA,DE,FC
 2530 data 00,0A,0C,80,00,00,00,00,5D,C
 0,C0,7C,00,01,39,40
 2540 data E8,78,67,0A,20,3C,FF,FF,FF,F
 F,60,0A,60,08,42,6C
 2550 data E8,78,20,2E,00,0A,4E,5E,4E,7
 5,4E,56,00,00,2F,2E
 2560 data 00,08,3F,3C,00,49,4E,BA,00,8
 E,5C,8F,4E,5E,4E,75
 2570 data 4E,56,FF,FE,42,6E,FF,FE,60,2
 6,30,2E,FF,FE,E5,80
 2580 data 41,EC,E7,48,D0,C0,30,10,B0,6
 E,00,08,66,0E,30,2E
 2590 data FF,FE,E5,80,41,EC,E7,48,D0,C
 0,42,50,52,6E,FF,FE
 2600 data 0C,6E,00,4C,FF,FE,6D,D2,4E,5
 E,4E,75,4E,56,FF,FE
 2610 data 3F,07,3E,2E,00,08,3D,7C,01,3
 0,FF,FE,41,EC,E7,48
 2620 data 22,48,D2,EE,FF,FE,BE,50,67,0
 6,58,48,B3,C8,66,F6
 2630 data 30,28,00,02,3E,1F,4E,5E,4E,7
 5,29,49,E7,44,29,4A

2640 data E7, 40, 29, 5F, E7, 3C, 4E, 4D, 22, 6
 C, E7, 44, 24, 6C, E7, 40
 2650 data 2F, 2C, E7, 3C, 4E, 75, 29, 49, E7, 4
 4, 29, 4A, E7, 40, 29, 5F
 2660 data E7, 3C, 4E, 41, 22, 6C, E7, 44, 24, 6
 C, E7, 40, 2F, 2C, E7, 3C
 2670 data 4E, 75, 4E, 56, 00, 00, 60, 34, 20, 6
 E, 00, 08, 10, 10, 48, 80
 2680 data 80, 7C, 00, 0A, 66, 0E, 3F, 3C, 00, 0
 D, 3F, 3C, 00, 02, 4E, BA
 2690 data FF, C6, 58, 8F, 20, 6E, 00, 08, 52, A
 E, 00, 08, 10, 10, 48, 80
 2700 data 3F, 00, 3F, 3C, 00, 02, 4E, BA, FF, A
 E, 58, 8F, 30, 2E, 00, 0C
 2710 data 53, 6E, 00, 0C, 4A, 40, 66, C0, 4E, 5
 E, 4E, 75, 4E, 56, 00, 00
 2720 data 60, 18, 20, 6E, 00, 08, 52, AE, 00, 0
 8, 10, 10, 48, 80, 3F, 00
 2730 data 3F, 3C, 00, 04, 4E, BA, FF, 80, 58, 8
 F, 30, 2E, 00, 0C, 53, 6E
 2740 data 00, 0C, 4A, 40, 66, DC, 4E, 5E, 4E, 7
 5, 4E, 56, 00, 00, 60, 18
 2750 data 20, 6E, 00, 08, 52, AE, 00, 08, 10, 1
 0, 48, 80, 3F, 00, 3F, 3C
 2760 data 00, 05, 4E, BA, FF, 52, 58, 8F, 30, 2
 E, 00, 0C, 53, 6E, 00, 0C
 2770 data 4A, 40, 66, DC, 4E, 5E, 4E, 75, 4E, 5
 6, FF, F2, 48, E7, 00, 30
 2780 data 26, 6E, 00, 0A, 24, 4B, 41, EC, 00, 2
 0, 2D, 48, FF, F6, 0C, 6E
 2790 data 83, 00, 00, 08, 66, 16, 3F, 2E, 00, 0
 E, 2F, 0B, 4E, BA, FF, 34
 2800 data 5C, 8F, 3D, 6E, 00, 0E, FF, FE, 60, 0
 0, 01, 4C, 0C, 6E, 82, FF
 2810 data 00, 08, 66, 16, 3F, 2E, 00, 0E, 2F, 0
 B, 4E, BA, FF, 60, 5C, 8F
 2820 data 3D, 6E, 00, 0E, FF, FE, 60, 00, 01, 2
 E, 0C, 6E, 82, FE, 00, 08
 2830 data 66, 16, 3F, 2E, 00, 0E, 2F, 0B, 4E, B
 A, FF, 70, 5C, 8F, 3D, 6E
 2840 data 00, 0E, FF, FE, 60, 00, 01, 10, 2D, 4
 B, FF, FA, 42, 6E, FF, FE
 2850 data 3F, 2E, 00, 08, 4E, BA, FE, 76, 54, 8
 F, 4A, 40, 66, 00, 00, A6
 2860 data 60, 00, 00, 94, 10, 12, 48, 80, B0, 7
 C, 00, 0A, 66, 00, 00, 86
 2870 data 20, 0A, 90, AE, FF, FA, B0, 7C, 00, 0
 0, 6F, 40, 20, 0A, 90, AE
 2880 data FF, FA, 48, C0, 2D, 40, FF, F2, 2F, 2
 E, FF, FA, 2F, 2E, FF, F2
 2890 data 3F, 2E, 00, 08, 3F, 3C, 00, 40, 4E, B
 A, FE, 7C, DE, FC, 00, 0C
 2900 data 39, 40, E8, 78, 48, C0, B0, AE, FF, F
 2, 67, 08, 30, 3C, FF, FF
 2910 data 60, 00, 00, A8, 30, 2C, E8, 78, D1, 6
 E, FF, FE, 2F, 2E, FF, F6
 2920 data 2F, 3C, 00, 00, 00, 02, 3F, 2E, 00, 0
 8, 3F, 3C, 00, 40, 4E, BA
 2930 data FE, 46, DE, FC, 00, 0C, 39, 40, E8, 7
 8, 0C, 40, 00, 02, 67, 06
 2940 data 30, 3C, FF, FF, 60, 74, 52, 6E, FF, F
 E, 52, 8A, 20, 0A, 2D, 40
 2950 data FF, FA, 60, 02, 52, 8A, 20, 0A, 90, 8
 B, B0, 6E, 00, 0E, 65, 00
 2960 data FF, 64, 60, 10, 20, 0B, 36, 2E, 00, 0
 E, C6, BC, 00, 00, FF, FF
 2970 data 00, 83, 24, 40, 20, 0A, 90, AE, FF, F
 A, 48, C0, 2D, 40, FF, F2
 2980 data 2F, 2E, FF, FA, 2F, 2E, FF, F2, 3F, 2
 E, 00, 08, 3F, 3C, 00, 40
 2990 data 4E, BA, FD, E4, DE, FC, 00, 0C, 39, 4
 0, E8, 78, 48, C0, B0, AE
 3000 data FF, F2, 67, 06, 30, 3C, FF, FF, 60, 1
 0, 30, 2C, E8, 78, D1, 6E
 3010 data FF, FE, 42, 6C, E8, 78, 30, 2E, FF, F
 E, 4C, DF, 0C, 00, 4E, 5E
 3020 data 4E, 75, 43, EC, EF, B4, 12, FC, 00, 5
 B, 12, FC, 00, 32, 12, FC

3030 data 00, 5D, 12, FC, 00, 5B, 12, FC, 00, 2
 0, 12, FC, 00, 42, 12, FC
 3040 data 00, 75, 12, FC, 00, 73, 12, FC, 00, 7
 9, 12, FC, 00, 20, 12, FC
 3050 data 00, 42, 12, FC, 00, 75, 12, FC, 00, 6
 4, 12, FC, 00, 64, 12, FC
 3060 data 00, 79, 12, FC, 00, 20, 12, FC, 00, 6
 2, 12, FC, 00, 79, 12, FC
 3070 data 00, 20, 12, FC, 00, 4D, 12, FC, 00, 6
 1, 12, FC, 00, 74, 12, FC
 3080 data 00, 2A, 12, FC, 00, 52, 12, FC, 00, 6
 1, 12, FC, 00, 74, 12, FC
 3090 data 00, 7C, 12, FC, 00, 20, 12, FC, 00, 7
 C, 12, FC, 00, 20, 12, FC
 3100 data 00, 42, 12, FC, 00, 61, 12, FC, 00, 6
 3, 12, FC, 00, 6B, 12, FC
 3110 data 00, 73, 12, FC, 00, 70, 12, FC, 00, 6
 1, 12, FC, 00, 63, 12, FC
 3120 data 00, 65, 12, FC, 00, 20, 12, FC, 00, 7
 3, 12, FC, 00, 65, 12, FC
 3130 data 00, 6C, 12, FC, 00, 65, 12, FC, 00, 6
 3, 12, FC, 00, 74, 12, FC
 3140 data 00, 2C, 12, FC, 00, 20, 12, FC, 00, 4
 1, 12, FC, 00, 53, 12, FC
 3150 data 00, 43, 12, FC, 00, 49, 12, FC, 00, 4
 9, 12, FC, 00, 3F, 12, FC
 3160 data 00, 20, 12, FC, 00, 5D, 12, FC, 00, 5
 B, 12, FC, 00, 20, 12, FC
 3170 data 00, 38, 12, FC, 00, 7C, 12, FC, 00, 2
 0, 12, FC, 00, 31, 12, FC
 3180 data 00, 32, 12, FC, 00, 36, 12, FC, 00, 7
 C, 12, FC, 00, 20, 12, FC
 3190 data 00, 31, 12, FC, 00, 32, 12, FC, 00, 3
 7, 12, FC, 00, 5D, 12, FC
 3200 data 00, 00, 43, EC, EF, 8C, 12, FC, 00, 5
 B, 12, FC, 00, 31, 12, FC
 3210 data 00, 5D, 12, FC, 00, 5B, 12, FC, 00, 2
 0, 12, FC, 00, 7C, 12, FC
 3220 data 00, 20, 12, FC, 00, 42, 12, FC, 00, 7
 5, 12, FC, 00, 73, 12, FC
 3230 data 00, 79, 12, FC, 00, 20, 12, FC, 00, 4
 2, 12, FC, 00, 75, 12, FC
 3240 data 00, 64, 12, FC, 00, 64, 12, FC, 00, 7
 9, 12, FC, 00, 20, 12, FC
 3250 data 00, 69, 12, FC, 00, 73, 12, FC, 00, 2
 0, 12, FC, 00, 6E, 12, FC
 3260 data 00, 6F, 12, FC, 00, 77, 12, FC, 00, 2
 0, 12, FC, 00, 4F, 12, FC
 3270 data 00, 46, 12, FC, 00, 46, 12, FC, 00, 2
 0, 12, FC, 00, 7C, 12, FC
 3280 data 00, 20, 12, FC, 00, 5D, 12, FC, 00, 5
 B, 12, FC, 00, 20, 12, FC
 3290 data 00, 4F, 12, FC, 00, 4B, 12, FC, 00, 2
 0, 12, FC, 00, 5D, 12, FC
 3300 data 00, 00, 43, EC, EF, 48, 12, FC, 00, 5
 B, 12, FC, 00, 32, 12, FC
 3310 data 00, 5D, 12, FC, 00, 5B, 12, FC, 00, 2
 0, 12, FC, 00, 42, 12, FC
 3320 data 00, 75, 12, FC, 00, 73, 12, FC, 00, 7
 9, 12, FC, 00, 20, 12, FC
 3330 data 00, 42, 12, FC, 00, 75, 12, FC, 00, 6
 4, 12, FC, 00, 64, 12, FC
 3340 data 00, 79, 12, FC, 00, 20, 12, FC, 00, 4
 D, 12, FC, 00, 61, 12, FC
 3350 data 00, 78, 12, FC, 00, 69, 12, FC, 00, 6
 D, 12, FC, 00, 75, 12, FC
 3360 data 00, 6D, 12, FC, 00, 20, 12, FC, 00, 7
 C, 12, FC, 00, 20, 12, FC
 3370 data 00, 54, 12, FC, 00, 69, 12, FC, 00, 6
 D, 12, FC, 00, 65, 12, FC
 3380 data 00, 20, 12, FC, 00, 4C, 12, FC, 00, 6
 9, 12, FC, 00, 6D, 12, FC
 3390 data 00, 69, 12, FC, 00, 74, 12, FC, 00, 3
 F, 12, FC, 00, 20, 12, FC
 3400 data 00, 7C, 12, FC, 00, 20, 12, FC, 00, 7
 C, 12, FC, 00, 20, 12, FC
 3410 data 00, 4D, 12, FC, 00, 69, 12, FC, 00, 6
 E, 12, FC, 00, 75, 12, FC

```

3420 data 00,74,12,FC,00,65,12,FC,00,7
3,12,FC,00,20,12,FC
3430 data 00,5D,12,FC,00,5B,12,FC,00,2
0,12,FC,00,20,12,FC
3440 data 00,35,12,FC,00,20,12,FC,00,7
C,12,FC,00,20,12,FC
3450 data 00,31,12,FC,00,35,12,FC,00,2
0,12,FC,00,7C,12,FC
3460 data 00,30,12,FC,00,36,12,FC,00,3
0,12,FC,00,20,12,FC
3470 data 00,5D,12,FC,00,00,43,EC,EF,0
E,12,FC,00,5B,12,FC
3480 data 00,32,12,FC,00,5D,12,FC,00,5
B,12,FC,00,20,12,FC
3490 data 00,42,12,FC,00,75,12,FC,00,7
3,12,FC,00,79,12,FC
3500 data 00,20,12,FC,00,42,12,FC,00,7
5,12,FC,00,64,12,FC
3510 data 00,64,12,FC,00,79,12,FC,00,2
0,12,FC,00,54,12,FC
3520 data 00,49,12,FC,00,4D,12,FC,00,4
5,12,FC,00,4F,12,FC
3530 data 00,55,12,FC,00,54,12,FC,00,2
0,12,FC,00,7C,12,FC
3540 data 00,20,12,FC,00,7C,12,FC,00,2
0,12,FC,00,53,12,FC
3550 data 00,68,12,FC,00,61,12,FC,00,6
C,12,FC,00,6C,12,FC
3560 data 00,20,12,FC,00,49,12,FC,00,2
0,12,FC,00,3F,12,FC
3570 data 00,20,12,FC,00,5D,12,FC,00,5
B,12,FC,00,20,12,FC
3580 data 00,45,12,FC,00,78,12,FC,00,6
9,12,FC,00,74,12,FC
3590 data 00,20,12,FC,00,7C,12,FC,00,2
0,12,FC,00,52,12,FC
3600 data 00,65,12,FC,00,73,12,FC,00,7
4,12,FC,00,61,12,FC
3610 data 00,72,12,FC,00,74,12,FC,00,2
0,12,FC,00,5D,12,FC
3620 data 00,00,43,EC,E8,C6,22,FC,00,0
0,00,00,32,FC,00,00
3630 data 22,FC,00,00,00,00,32,FC,00,0
9,32,FC,83,00,22,FC
3640 data 00,00,00,00,32,FC,00,00,22,F
C,00,00,00,00,32,FC
3650 data 00,00,22,FC,00,00,00,00,32,F
C,01,02,32,FC,83,00
3660 data 22,FC,00,00,00,00,32,FC,02,0
0,22,FC,00,00,00,00
3670 data 32,FC,00,00,22,FC,00,00,00,0
0,32,FC,01,02,32,FC
3680 data 83,00,22,FC,00,00,00,00,32,F
C,02,00,D2,FC,05,78
3690 data 39,7C,02,00,E8,C4,4E,75,20,2
0,42,75,73,79,20,42
3700 data 75,64,64,79,00,00,43,4F,4E,3
A,00,00,41,55,58,3A
3710 data 00,00,50,52,54,3A,00,00,0D,0
A,00,00,00,00,00,02
3720 data 06,00
3730 data *

```

Listing 2:

C

```

/*****/
/* Busy Buddy accessory */
/* Copyright 1988 by ST-Log */
/* RatWare by */
/* Matthew J. W. Ratcliff */
/* AKA MatRat, All Rights */
/* Reserved. */
/*****/

```

```

/*****/
/* EXTERNALS */
/*****/

```

Listing 1: Checksums

```

100 data 892, 948, 117, 614, 503, 22
2, 410, 427, 14, 109, 4256
190 data 654, 357, 649, 562, 84, 997
, 774, 871, 9, 148, 5105
1080 data 85, 766, 147, 934, 55, 216
, 376, 255, 374, 375, 3583
1180 data 397, 274, 876, 170, 887, 3
7, 75, 62, 831, 29, 3638
1280 data 59, 82, 819, 700, 685, 788
, 741, 569, 15, 934, 5392
1380 data 889, 937, 990, 854, 851, 9
11, 838, 812, 68, 889, 8039
1480 data 989, 900, 51, 991, 915, 74
, 52, 125, 641, 828, 5566
1580 data 741, 687, 707, 780, 857, 9
06, 492, 481, 511, 485, 6647
1680 data 494, 494, 507, 493, 495, 4
77, 478, 522, 504, 485, 4949
1780 data 484, 485, 488, 500, 495, 4
90, 492, 773, 936, 152, 5295
1880 data 248, 224, 149, 208, 380, 2
82, 236, 73, 253, 948, 3001
1980 data 102, 967, 113, 88, 947, 11
6, 98, 8, 40, 60, 2539
2080 data 945, 998, 945, 934, 864, 7
40, 25, 762, 887, 61, 7161
2180 data 858, 29, 104, 798, 983, 74
1, 958, 988, 792, 746, 6997
2280 data 904, 816, 901, 863, 879, 1
16, 812, 752, 805, 794, 7642
2380 data 930, 101, 932, 818, 920, 4
9, 847, 950, 945, 940, 7432
2480 data 869, 147, 27, 991, 969, 75
3, 114, 862, 921, 267, 5920
2580 data 974, 308, 194, 16, 206, 85
0, 34, 988, 34, 727, 4331
2680 data 881, 902, 956, 857, 738, 9
39, 854, 771, 896, 64, 7858
2780 data 957, 883, 45, 936, 970, 1,
156, 955, 751, 47, 5701
2880 data 366, 41, 217, 184, 783, 96
9, 92, 965, 18, 162, 3797
2980 data 64, 266, 112, 196, 72, 908
, 907, 906, 909, 906, 5246
3080 data 909, 918, 910, 906, 898, 9
29, 886, 913, 906, 900, 9075
3180 data 900, 901, 1, 938, 895, 902
, 907, 924, 937, 912, 8217
3280 data 911, 937, 982, 915, 914, 9
13, 917, 951, 922, 940, 9302
3380 data 936, 924, 921, 954, 909, 9
12, 899, 904, 871, 17, 8247
3480 data 922, 932, 900, 910, 938, 9
20, 906, 956, 902, 919, 9205
3580 data 938, 910, 926, 925, 897, 8
92, 764, 795, 765, 866, 8678
3680 data 882, 861, 768, 595, 566, 2
15, 3887

```

```

extern int gl_apid;

/*****/
/* Include files */
/*****/

#include <stdio.h>
#include <osbind.h>

/*****/
/* Global variables */
/*****/

char Back_Type[] =
"[2][ Busy Buddy by Mat*Rat | | Backspace select, ASCII? ][ 8 | 126 | 127]";

char Buzzoff[] =
"[1][ | Busy Buddy is now OFF | ][ OK ]";

char Busy_Time[] =
"[2][ Busy Buddy Maximum | Time Limit? | | Minutes ][ 5 | 15 | 60 ]";

char Busy_Tout[] =
"[2][ Busy Buddy TIMEOUT | | Shall I ? ][ Exit | Restart ]";

char Buddy_Byte, Backup;
int Time_Limit;

/*****/
/* Constants */
/*****/

#define EVNT_MSG      0x0010
#define EVNT_TIM      0x0020
#define TIME_H        (int)0
#define TIME_L        (int)2000
#define AC_OPEN       40      /* Startup accessory cmd*/
#define ASCII_8        1
#define ASCII_126      2
#define ASCII_127      3
#define BS             8      /* Backspace character */
#define XE_DEL        126     /* Delete character-XE */
#define ST_DEL        127     /* Delete char-ST */
#define SPACE         32      /* Space character */
#define PRT            0      /* Standard IO devices */
#define AUX            1
#define CON            2
#define MIDI           3
#define KBD            4
#define FIVE_MIN       (int)150
#define FIFTEEN_MIN    (int)450
#define SIXTY_MIN      (int)1800
#define SEL_FIVE        1
#define SEL_FIFTEEN    2
#define SEL_SIXTY       3
#define BELL            7

/*****/
/* Program: Busy Buddy */
/* Type: Desk accessory */
/* Purpose: Busy Buddy, when activated, initiates */
/* a 2 second timed interval for sending alternating */
/* space and backspace characters over the modem. */
/* This feature will keep the system at the other */
/* end BUSY. This will prevent most BBS and */
/* telecommunication systems from TIMING OUT */
/* on you, so you have time for a Budweiser break. */
/*****/

main()
( /* Get busy BUDDY! */

int menu_id;

```

```

int mgin[8];
int dum,event,flag;
int forever;

/*****
/* Install accessory: */
*****/

appl_init();
menu_id = menu_register(gl_apid,"  Busy Buddy");

/*****
/* Event message only, until activated the first time */
*****/

flag  = EVNT_MSG;
forever = 1;

while (forever)
{ /* loop forever */

    dum  = 0;
    event = evnt_multi( flag, -1, -1, -1,
                        dum, dum, dum, dum, dum, /* de-daaa */
                        dum, dum, dum, dum, dum, /* de-dooo */
                        mgin, TIME_L, TIME_H,
                        &dum, &dum, &dum, &dum, &dum, &dum);

    if (event & EVNT_MSG)
        if (mgin[0] == AC_OPEN && mgin[4] == menu_id)
            if (flag & EVNT_TIM)
            {
                flag = EVNT_MSG;
                Buddy_Off();
                event = 0;
            }
            else
            {
                flag = EVNT_TIM | EVNT_MSG;
                Buddy_On();
                event = 0;
            }

        if (event & EVNT_TIM)
            if (flag & EVNT_TIM)
            {
                Buddy();
                if (Time_Limit <= 0)
                    flag = Buddy_Toff();
            }

    } /* end loop forever */

} /* end main() */

/*****
/* Function:      Buddy_On()
/* Description: Initialize Busy Buddy. Allow user to select
/* backspace or delete type of backup. Initialize Backup
/* variable and send first SPACE character to begin process.
*****/

Buddy_On()

{ /* begin Buddy_On() */

int bsel;

bsel = form_alert(1, Back_Type);          /* Get backspace or delete type */

switch (bsel)
{ /* begin switch */
    case (ASCII_8):
        Backup = BS;
        break;

    case (ASCII_126):
        Backup = XE_DEL;

```

```

    break;

case (ASCII_127):
    Backup = ST_DEL;
    break;

default:
    break;

} /* end switch */

Get_Limit();                /* Set time limit */
Send_Byte( SPACE );         /* Start the busy signal */
Buddy_Byte = Backup;        /* setup for backspace next time*/

} /* end Buddy_On() */

/*****
/* Function:   Buddy_Off()
/* Description: Make certain that a backup character was the
/* last thing sent (spaces must be matched with backups),
/* then inform the user that Busy Buddy is dormant.
*****/

Buddy_Off ()

{ /* begin Buddy_Off() */

    if (Buddy_Byte == Backup)        /* Make certain that backup was */
        Send_Byte( Buddy_Byte );    /* last character sent */

    form_alert(1, Buzzoff );         /* Tell user busy is off now */

} /* end Buddy_Off() */

/*****
/* Function:   Buddy()
/* Description: Send the current character from Buddy_Byte.
/* Then toggle from SPACE to BACKUP, or BACKUP to SPACE for
/* next 2 second interrupt.
*****/

Buddy()

{ /* begin Buddy() */

    Send_Byte( Buddy_Byte );         /* Send current space or backup */

    if (Buddy_Byte == SPACE)        /* If space just sent, backup */
        Buddy_Byte = Backup;        /* next time. */
    else
        Buddy_Byte = SPACE;         /* If backup just sent, then */
                                    /* send space next time. */

    Time_Limit--;

} /* end Buddy() */

/*****
/* Function:   Send_Byte( char )
/* Description: Send the character received to the AUX, RS232*
/* port, using the Bconout function.
*****/

Send_Byte( dude )
char dude;

{ /* begin Send_Byte( char ) */

/*****
/* Send the caracter passed to the AUX, RS232 Port */
*****/

Bconout ( AUX, dude );

```

```

} /* end Send_Byte( char ) */

/*****
/* Function:   Buddy_Toff()   returns ( INT )
/* Description: Make certain that a backup character was the
/* last thing sent (spaces must be matched with backups),
/* then inform the user that Busy Buddy is timed out.
/* Return flag status for next event multi.
*****/

Buddy_Toff ()

{ /* begin Buddy_Toff() */

#define BEXIT    1
#define BRESTART 2

int sel, bflag;

if (Buddy_Byte == Backup)          /* Make certain that backup was */
    Send_Byte( Buddy_Byte );        /* last character sent          */

Bconout( CON, BELL);                /* WAKE UP CALL!                */

sel = form_alert(1, Busy_Tout );     /* Tell user busy timed out     */

if (sel == BEXIT)
{
    bflag = EVNT_MSG;
    form_alert(1, Buzzoff);
}
else
{
    bflag = EVNT_MSG | EVNT_TIM;
    Get_Limit();
}

return ( bflag );

} /* end Buddy_Toff() */

/*****
/* Function:   Get_Limit()
/* Description: Get maximum time limit for Busy Buddy to run
/* unattended.
*****/

Get_Limit()

{ /* begin Get_Limit */

int bsel;

bsel = form_alert (3, Busy_Time );

switch ( bsel )
{ /* begin switch */
case (SEL_FIVE):
    Time_Limit = FIVE_MIN;
    break;

case (SEL_FIFTEEN):
    Time_Limit = FIFTEEN_MIN;
    break;

case (SEL_SIXTY):
    Time_Limit = SIXTY_MIN;
    break;

default:
    break;
} /* end switch */

} /* end Get_Limit */

```

(continued from page 17)

best features of the MAC, ported over to the ST. How about a desktop that allows you to place a favorite or often used program as a permanent part of the desktop? No, not a directory entry on a disk, but part of the desktop itself, just like your disk drive and trash icons. Speaking of icons, they do of course support editable icons and about 40 other new features. Called **NeoDesk**, this package retails for \$29.95... and is available now.

Atari dealers are really feeling the pinch as Atari heads into the summer. 8-bit sales, other than the XEGS system, have slowed to a crawl for most retailers. Mail order seems to be getting by, but if you haven't been to your favorite 8-bit dealer lately, don't be surprised if the next time you go he is devoting more and more space to other products. Can it be that Atari 8-bit owners have all the software they want? If they went into stores carrying Atari 8-bit products more often, they would learn about great new products like **The Newsroom**—a good desktop publishing package for the 8-bit.

There are at least two companies developing 16 MHz boards for the ST. The ST 68000 CPU normally operates at 8 MHz, and it is this speed which limits how fast the ST can run a program. These new boards will replace the present 68000 with double speed units that will make your ST work at least 150 percent faster than the present unit. Cost? About \$200, not including installation. But be careful, the installation could be a major cost. The 68000 comes soldered onto the ST board and has to be desoldered and then resoldered on the new board. This is no job for a friend using a woodburning set as a soldering iron.

Another company has already developed a card cage for the Megs, but unfortunately, they are only planning on offering it as part of their vertical market package (their own specialty software package). Too bad, since it has the ability to take most standard PC and AT cards. Nifty stuff like hard disk, modem and Fax cards. You know, all that good stuff you see advertised in PC mags for the guys that paid *big bucks* for their systems?

If you haven't connected your ST to a stereo yet, *do it!* The sounds will floor

ST Gossip

you. Try it with something like **Blockbuster** if you want to see how a good sound system can turn a nice game into something really super.

Talk to Lou Swilling from Astra Systems. Seems he's on a switch lately. By the time you read this ("famous last words"), Astra should be offering the following switch boxes: printer switch, DMA switch, floppy drive switch, RS232 switch (with two RS232 outputs and built-in null modem). Their wonderful monitor switch has been out for several months. Having trouble making up your mind, Lou?

You might want to watch for a new advertising campaign sponsored by Astra and Hybrid Arts. The Ads will feature the Pointer Sisters standing in front of their 1040s and thousands of dollars worth of Hybrid Arts software and hardware. Clearly seen in the ad is the new rack mounted Astra 240 Meg hard disk.

Missing, but not forgotten dept.

Missing: one multi-tasking, multi-user operating system called IDRIS. This is the Unix work-a-like system that Atari was hyping before Christmas. Anyone heard anything about it lately? When asked about it a week ago, Atari said it was available, but not being marketed by Atari. Something about failing to reach an agreement on price. The product is finished and is marketed by the company that originally wrote it. In the meantime, Atari has been advertising for a full software team to adapt a version of Unix 3.2 for them. (An OS for the 68030 product?).

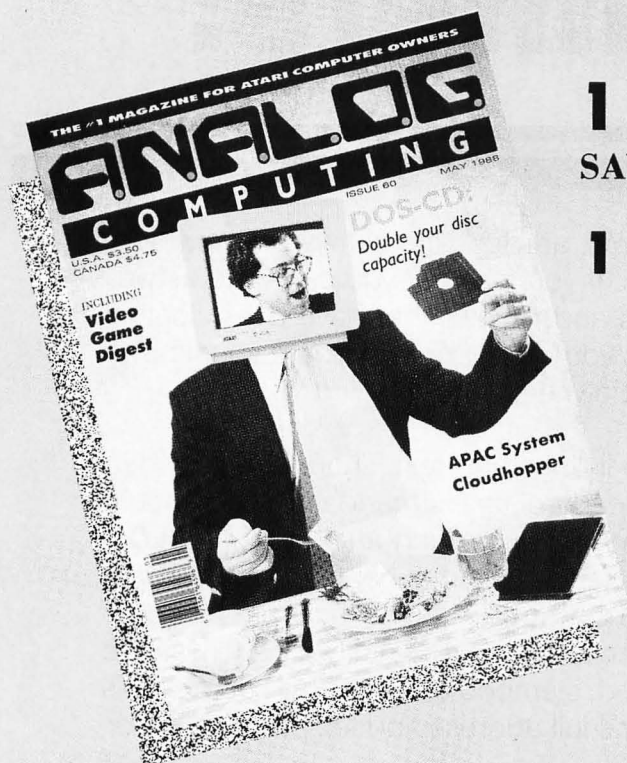
Did I say 68030? Sure the one that was shown in the back room at the Hanover Trade Fair this past spring (the CBit computer show in West Germany). You know, the one that will plug into the Mega and use it as a front end? You already own a keyboard attached to a box with a bunch of ports in it, right? Why buy another keyboard or more ports just to quadruple the power of your system? Just buy this box with a 68030, its memory and operating system and plug it into your mega via the expansion port. No, an upgraded 520 or 1040 does not equal a mega in this case.

Congratulations! Atari is now listed in the Fortune 500. Yes, the guys from Sunnyvale are number 486 with a bullet. P.S. Commodore was nowhere on the list.

Finally—Rupert Murdoch (of big media fame) recently ordered 7000 PC clones to be delivered within 30 days. Orders like this just don't happen in the ST world. But wouldn't it be nice...

TG can often be found skulking the turf around Hollywood and Vine where many a celebrity has been discovered. He is seen frequenting bars in the area, carrying what he calls the only true portable computer: an ST plugged into a heavy-duty truck battery. He writes this column to make a living until he breaks into film, and to provide the cash he needs to recharge his truck battery every month. Hear anything good? 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? Send TG mail to: ST-Log, 9171 Wilshire Bld., Suite 300, Beverly Hills, CA 90210)

BOOT UP TO BIG SAVINGS!



1 YEAR FOR ONLY \$28

SAVE \$14 OFF THE COVER PRICE

1 YEAR WITH DISK ONLY \$105

**SAVE TIME AND MONEY
SUBSCRIBE TO ANALOG**

**SAVE \$14 OFF THE
COVER PRICE WITH
THE CONVENIENCE
OF HAVING ANALOG
DELIVERED DIRECT-
LY TO YOUR DOOR
BEFORE IT EVEN HITS
THE NEWSSTANDS.
GET THE MOST OUT
OF YOUR COMPUTER.**

**SUBSCRIBE TO
ANALOG
TODAY**

- ☐ 1 YEAR @ \$28 — SAVE \$14!
FOREIGN — ADD \$7 PER YEAR
☐ 1 YEAR WITH DISK @ \$105
FOREIGN — ADD \$15 PER YEAR

MCGWY

DCGWY

☐ 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 16927, N. Hollywood, CA 91615. Offer expires August 31, 1988. Your first issue will arrive in 6 TO 8 weeks.

WATCH FOR IT!

Attention Programmers!

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

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

For those of you who are sincerely interested in the rules and regulations for publication, we've taken this opportunity to print our guidelines for authors. See page 128 of this book for everything you'll need to know.

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



TUTORIAL

Read Any Good Docs, Lately?

by David L. Coles

Only seconds remain until the completion of your sophisticated program. The debugging finished, you smile in the knowledge that shortly the mail will bring the announcement of your induction into the "Codiers Hall of Fame." But as you stagger off to relax it hits you—you're not through yet!

You've forgotten the little matter of documentation. No sweat, you reason; tomorrow you'll knock something out to guide the troops through the woods. Unfortunately, if you believe that will be sufficient, you're either too tired to think straight, or are grossly mistaken about the needs of computer users. Your software may well ensure your place in the Codiers Hall of Fame, but do you really want all those angry users burning the documentation in protest during the induction ceremonies?

Inadequate or confusing documentation dooms many otherwise excellent programs to collect dust after only a few uses, while it prevents others from being used to their fullest potential. Rather than just "knocking something out," you have the responsibility to complement your software with good quality documentation. But don't con-

fuse this with generating reams of paper. The task is to strike a balance between frustrating users with too little information and frightening them with too much. Ideally, you provide just the right amount of instruction to get the program installed, fully explain its operation, and provide ready references when a quick refresher is needed.

One very effective method of producing such quality documentation is for you to actually work through the installation and operation of the program yourself, noting each step as you go. These notes are then collated into an outline, and eventually become the backbone of your documentation. This article explains how to utilize this method, and presents some helpful tips to make your documentation more interesting. But the thought processes used for developing documentation are different than those used for programming. So, before you grab your favorite crayon to begin writing, set the project aside for several days, allowing sufficient time to clear your mind.

The logical way to start is to place yourself in exactly the same position that your users will be in when they

first lay hands on your program. Only by treating yourself as a first-time user will you be able to spot some of the more obvious items ("Oh, did my documentation neglect to mention that you had to have BASIC on the working disk . . . sorry 'bout that!"). To do this you will need to start with a working distribution copy of your software, one that will be identical to the one users will eventually have. This means it should contain only the programs you will be distributing, without any operating system, interpreters, compilers, or other proprietary commands.

Program installation is usually a one-time function, so it is best to treat it as a separate section in the documentation. Installation might be simply a matter of getting the command(s) onto a "working" system disk; or it could involve implementing specific peripherals, or special command and/or data file configurations. In any event, work through all the various installation functions, keeping a step-by-step log of exactly what you did (and in what order you did it) until you have completed installing your program.

The operating instructions are the real heart of any documentation. As

the program author, you have invested countless hours producing the program and are therefore familiar with every nuance of its operation. This section of the documentation has the task of successfully transferring all this operational expertise to the user.

Your program may be completely menu-driven, with the ability to walk users through its operation; or it may require that the user be familiar with special "commands." Either way, work through all the various operating functions, keeping a step-by-step log similar to the one you kept for the installation function. Once this phase is completed you will have compiled a set of notes which detail every step necessary for installing and operating your program.

Next, you'll need to formulate an outline for your documentation. A representative, full-blown outline might include the following:

- Cover Sheets
 - Title page
 - Copyright, disclaimers & warranty
- Table of Contents
- Introduction
 - Program features, results, etc.
 - Special notes (system configuration, etc.)
- Getting Started
 - System familiarization
 - System start-up
 - Creating backup & working disks
- Installation
 - (Part of your notes will be used here.)
- Operating Instructions
 - (The majority of your notes will be utilized here.)
- Reference section
- Index

With your notes and suitable outline, you're ready to begin writing the documentation. It may appear that the hardest part is yet to come. But remember, the majority of your documentation will come from the notes you have taken. What remains now is simply to transform these notes into an instructional text format and place them in the appropriate section.

To accomplish this, each function of installation and operation should be introduced with a function definition and an explanation of any associated options. The definition is followed by a narrative explaining how to execute

the function and an execution example.

To illustrate, let's see how this might work for a fictitious program entitled "DOIT." Assume that the program is not menu-driven, but instead requires that all optional arguments be included in the "command tail."

DOIT is a program which . . . (definition would be presented here).

DOIT allows you to select various options. You may choose:

- The *drive* containing the file you want DOIT to use.
- The *name* of the file to DOIT to
- The *echo* results to your printer
- The *number* of times to DOIT
- The individual *record* to start with
- The field number to *start* with
- The field number to *stop* with

These options are selected at the time DOIT is executed from the command (or ready) state.

To illustrate the operation of DOIT, let's assume:

- Drive A contains the file you want DOIT to use
- You want to DOIT to a file named FILENAME.EXT.
- You want the results echoed to the printer
- You want to DOIT three times
- You want to start with record #1
- You want to select field #3 through #4

Under these circumstances you would execute DOIT by entering:

The commas (,) are important, so be sure to include them. If you choose *not* to use a particular option, you must still include the comma. To demonstrate this important use of punctuation, the following example illustrates the execution of DOIT, first echoing the results to the printer and then again without echoing the results (as a training aid, an arrow () has been placed under the "P" which selects this option):

"P" selects echo option

DOIT A:FILENAME.EXT,P,3,1,3,4

Missing "P" eliminates echo option

DOIT A:FILENAME.EXT,,3,1,3,4

Note in both examples that the total number of commas remains the same, only the "P" is affected!

This example merely highlights the general nature of presenting operating instructions. In your documentation you would want to provide greater detail and cover additional items, such as defaults.

Providing a reference section in your documentation can be a real time saver for the experienced user. Rather than

DOIT A:FILENAME.EXT,P,3,1,3,4

Stop field
 Start field
 Starting record
 Number of times to DOIT
 Echo results to printer
 Input file name
 Drive containing the input file
 Main command

Docs, Lately?

having to wade through the operating instruction section again, they can simply turn to the reference section for a quick memory job. Items you may want to consider for inclusion would be:

- Abbreviated command summary listing with brief description of each command sequence or keystroke
- Description of special files needed or created by your program
- List of error messages and remedies.
- Glossary of unfamiliar words
- User patches for such things as changing I/O ports, peripherals, etc.

While not actually part of a reference section, an Index and Table of Contents are handy reference tools. Both are used to quickly direct the user to a specific location in the documentation, but they approach their task from opposite directions. The Index is used to locate the desired page when the appropriate keyword is known. The Table of Contents is used when the general function is known but the specific keyword is not known.

Testing is the final, and most critical, part of documentation preparation. During the programming phase you tested all the program's permutations to spot the errors and omissions. This same type of testing needs to be performed on your documentation. However, this testing is better done by someone else. Give a friend a "distribu-

tion diskette" and a copy of your documentation. Ask him to go through it just as if he had bought it off the shelf. Have your friend make notes of those areas that were confusing, needed more explanation, or were missing altogether. You might even consider doing this with several friends of differing backgrounds and experience levels. Once your documentation has been thoroughly tested (a day or two for small- to medium-sized programs), review their comments and make the appropriate changes.

Before ending this discussion, there are a few technical points to consider. The vocabulary you use affects overall understandability. Imagine how a new computer user would react to words such as argument, array, operand, string and I/O. Unfamiliar words force your reader to jump extra hurdles of intimidation to master the operation of your software. If you must use potentially unfamiliar words, thoroughly explain them in plain English before thrusting them on your reader. Sentence structure is another factor which has a decided effect on readability. Keep sentences short and use commas sparingly. When tempted to include parenthetical statements using commas, ask yourself if readability would be increased by using two separate sentences instead. And finally, the manner in which explanations are presented affects how quickly they are under-

stood. Consider the following two presentations of identical instructions. Which one seems easier to follow?

THIS?

Place the command disk in drive A and the data disk in drive B. Turn the computer on and allow it to initialize itself. When the computer signals it is ready, type the command: *DOIT* (and the return key).

OR THIS?

- Place the command disk in drive A
- Place the data disk in drive B
- Turn the computer on and allow it to initialize itself
- When the computer signals it is ready, type the command:

DOIT (and the return key).

Presenting instructions in checklist format makes them easier to follow for someone who might be executing them "by-the-book."

The type and quality of documentation you provide with your program can greatly enhance its overall attractiveness. Good quality documentation, like good quality programming, does not just happen. It requires thoughtful effort on your part. When approached in a step-by-step manner, producing quality documentation is not an insurmountable task. In fact, it can actually be an exciting part of your overall software production efforts. (P.S. Enjoy the induction ceremony, you deserve it.)

FEATURE

Anatomy of a Simulation

Getting The Most From Your Flight Simulator II

by Bob Curtin

To those who've not experienced the Atari ST version of subLOGIC's **Flight Simulator II**, I can only extend my heartfelt condolences. I've had the opportunity to use this program on a variety of machines, including an IBM PC-AT which, by the way, runs nearly as briskly as the ST version (Yes, I said *nearly*), but doesn't compare in the quality of graphics, features or ease of use. Regardless of the machine on which FS2 is run, however, the program has a depth which becomes quite obvious if you do some research into the real thing, or better still, do some actual flying. Since I bought the program in 1984 for my Atari 800, FS2 has gotten literally hundreds of hours of machine time; more than anything else in my software library, with the exception of programming and word processing software.

In that time I've discovered a few things which might just make your own flying time a bit more enjoyable. FS2 is, after all, a simulation, and a very accurate one at that. You can spend countless hours just motoring about the skies, but if you're anything like me, you can only do so much sightsee-

ing before the novelty wears off. Eventually you'll want a challenge, something to sharpen your flying and navigational skills.

Fortunately, FS2 has enough to provide you with challenges for years to come. The following paragraphs contain a few suggestions on how to go about taking advantage of the great depth of the simulation and above all, have some more fun with it.

Need a Co-pilot?

Before we get started, however, a word to those who are stark beginners or are just plain baffled by Flight Simulator II. This article assumes you have at least some familiarity with FS2, but if not, you might be interested in Charles Gulick's *Flying Flight Simulator*, published by Microsoft Press at a relatively inexpensive \$9.95. The book is an excellent tutorial for beginners, as well as a handy reference for experienced FS2 pilots. Mr. Gulick starts from the beginning and gradually leads the student pilot through ever more difficult lessons until, by the end of the book, the simulation is covered in detail. *Flying Flight Simulator* is a comprehensive compendium of useful in-

formation and it's a painless way to learn this complex program.

The same author has written a couple of books entitled *40 (and 40 More) Great Flight Simulator Adventures*, published by Compute! Books. Yes, they're exactly what you think they are: a series of situations, complete with objectives, which you can set up on FS2 and play out. Now, on with the show.

How about a section 8?

The single greatest improvement you can make in your simulation flying is to give up navigating by the little maps which come with FS2 and the scenery disks and invest in a set of standard aeronautical sectional charts. You can pick them up at just about any airport or order them through an aviation supply house such as Sporty's Pilot Shop in Batavia, Ohio. A sectional chart is simply a map used by pilots to navigate both by visual flight rules and, in conjunction with other charts, instrument flight rules. There are 37 sectional charts covering the continental United States (not counting Alaska), and the existing scenery disks supplied by subLOGIC conform so closely, you can navigate by them remarkably well. Navigating under VFR (Visual Flight

**If you kick
the left
rudder you
can follow
the pike
south to
Baltimore.**

Rules) becomes much easier with the sectional charts, especially when using the more detailed scenery disks. Major roads, rivers, lakes, mountains, cities, and even landmarks are accurately represented. (And I'll bet that all this time you thought those roads were randomly set there just for looks; to fill up empty space.)

For example, taking off from runway 26 at Lancaster, PA, you cross over route 322, pick up route 30, fly over the Susquehanna river and run into route 83. If you kick left rudder you can follow the pike south all the way to Baltimore. Each of these features is plainly visible on the Scenery Disk, but without intimate knowledge of the area, or without the use of a chart, you really can't tell what you're looking at. If you've been wondering just what it is you've been seeing down there, they're well worth the modest price.

Although the FS2 Scenery Disks don't contain all of the airports in the real world, each sectional has around 30 of them—all of the major airports and most of the secondary ones. When you're ready for that information, sectional charts will show you the transition areas for instrument approaches, the airport control zones, and federal VOR airways. Scenery Disk #11, for example, contains four sectionals (all of which conform to standard NOAA sectional charts) and sports a grand total of 140 airports. Radio navigational aids are also in abundance, and almost all real-world beacons are represented with their corresponding frequencies. ATIS frequencies (Automatic Terminal Information Service) are present to a limited extent, as well as ILS (Instrument Landing System) at selected airports.

Another thing the sectionals show you is prohibited airspace and restricted areas. The airspace around military bases, for instance, and around major airports have certain requirements which must be met before private aircraft may enter. For some areas it's as simple as contacting Air Traffic Control before you enter, while for others, entry is strictly prohibited. I grant you that the long arm of the FAA is not about to pop out of the screen in front of you and bend your floppy in two for violating prohibited airspace, but just for kicks, try doing some cross-country flying while observing the FAA altitude restrictions and avoiding restricted areas. You'll find it adds a whole new dimension to cross-country flying on FS2. No longer do you feel as if you're the only pilot in the sky, but part of an enormously intricate air traffic control system and one who's expected to observe the rules.

And speaking of rules, I'll bet there aren't too many of you out there who ever enter into a standard traffic pat-

tern when approaching an airport. Yes, I know, it's hard enough to get the beast onto the runway in one piece without having to worry about being neat about it. Guaranteed, however, that once you get the hang of it, it'll make landing much easier. Flying a traffic pattern correctly automatically lines you up with the runway at the

I do recommend that you buy a good pilot's training manual.

right altitude for a nice, safe, uneventful landing. When you think you've got it down pat, try the same approach with a good, stiff crosswind. Then add some turbulence. Keep making things difficult for yourself until you hone your skill to a fine edge.

What's a standard traffic pattern, you ask? Before you leave the pilots' supply house with that fist full of sectional charts, think about buying yourself a pilot's handbook. It'll tell you all you need to know about traffic patterns.

And I *do* recommend that you buy a good pilot's training manual; subLOG-IC provides a list of appropriate publications in the front of their documentation that will definitely fit the bill. A good manual contains information on aerodynamics, meteorology, communications, aeronautical charts, navigation, airports, air traffic control, instruments, ILS, radio navigation, using a flight computer, and more. Also, invariably, there's a section containing the Federal Aviation Regulations. A lot of the material you'll find in these manuals won't be applicable to the Flight Simulator II, but you'll be surprised at just how much is.

Weather to Fly or Not?

Chugging around the clear, still, friendly skies is one thing. Taking off, flying, and landing in stiff winds, limited visibility and teeth-crunching turbu-

lence is quite another.

FS2 gives you full control over your environment, and what better way to add challenge to your flying than to conjure up some nice, hostile weather? High winds are great for creating havoc with your instrument flights, especially long flights in which you're using radio beacons to triangulate your position and set your course.

For this kind of flying, you'll need to make another trip to your pilot's supply house to pick up a flight computer; either electronic (wimpy) or an inexpensive mechanical flight computer (which you should learn to use even if you do have an electronic computer). The mechanical computers are slip-sticklike affairs costing about five bucks for the cardboard and plastic kind. You'll need one of these computers for figuring true heading, (i.e. your actual heading taking into account airspeed, wind speed and direction, and compass heading), ground-speed, fuel consumption, and other useful information necessary to keeping your bearings in foul weather and avoiding becoming an unwilling and quite sudden part of the landscape. Again, the pilot's handbook which you dutifully purchased will instruct you in the use of your new computer. And yes, it really does work. You *can* compute true heading, ETA, groundspeed and so forth, and apply the numbers to your simulation flying. Believe me, it's an education in itself.

Crossing That Line.

Now, what about when you want to fly from one sectional to another? Well, as long as both sectionals are on the same disk, there's no problem; you get no sensation of crossing a boundary. However, if the sectionals reside on different disks, there's a couple of minor problems. I must specify that although I've used FS2 on other machines, I've not had the opportunity to test their reaction to scenery disk manipulation, so this section I can only guarantee to ST users.

First, the original scenery, that is, the scenery which comes supplied on the FS2 disk itself, is not as accurate as the scenery disks which are sold separately. I don't mean in terms of what they portray, but only in their scope.

For instance, the New York sectional is missing almost 40,000 square miles on the west side; a giant rectangle from just west of New York City north to Tupper Lake, west to Prince

You're likely to end up with the same problem which befalls a pilot flying from New York to Washington—a side trip to the Twilight Zone.

Edward Bay in Lake Ontario, and then south again to a point just south of Harrisburg, Pennsylvania. Obviously, flying west from the New York sectional to the Detroit sectional is impossible with any degree of realism. And all of the sectionals supplied with the FS2 disk have similar problems. It seems that we'll just have to wait for the revised scenery disks for those areas to arrive.

However, you *can* fly from the New York to the Washington sectionals (and Los Angeles to San Francisco) after spending about 40 miles or so in a featureless limbo. Just be sure you change the disks before the simulator goes looking for new data.

Going from one scenery disk to another is no problem if you know exactly where you are. Each NOAA sectional chart has an overlapping area which matches up with adjoining

charts. When you hit one of those areas on a sectional disk, change the disk to the one you're flying to. As soon as you leave the overlapping area, press "E" on your keyboard and the new scenery disk will take over. Of course, if you don't know exactly where you are, you're likely to end up with the same kind of problem which befalls a pilot flying from New York to Washington—a side trip to the Twilight Zone.

Can You Play An Instrument?

When you've mastered the nuts and bolts of your simulator; that is, takeoffs, flying and landing, VFR navigation, and so on, the next logical step is instrument flying, including using the Instrument Landing System at a number of airports on the FS2 disk and the scenery disks.

Before you tackle instrument flying in bad weather, I suggest that you practice radio navigation in fair weather until it becomes second nature. Start off flying directly from beacon to beacon, and then graduate to using two beacons to triangulate your position. Once you master that, you're freed from "beacon hopping," and you'll gain a lot more skill in instrument navigation. With practice, and using a combination of radio navigation, experience, and a little dead-reckoning thrown in for good measure, you should become quite proficient at droning around the countryside, knowing exactly where you are at all times. When you get to that point, try an instrument landing. First try it in clear, calm weather and just kill your main window. (It's a lot easier to reactivate your main window if you get into trouble than it is to reset the cloud cover.)

Instrument landing approaches are probably the most difficult aspect of the flight simulator to master. It requires a combination of "flying" skill, proficiency at reading and interpreting instruments, navigational skills, and nerves of steel. Once you do a number of successful approaches under ideal conditions, toughen the environment slightly and do it all over again. Keep trying to make things difficult for yourself, and don't give up. As with anything else, constant practice and diligence will beat raw talent any day. (You've got to watch out for the talented ones who constantly practice, though.)

Deja Vu City

For a long time, I've been nursing this nagging curiosity about FS2. Just

how close to reality is the simulator, and just how much of what I've learned by using FS2 would help me to fly and navigate a real, honest-to-goodness Cessna? Aside from a short stint at the stick of a Blanec Sailplane (I hope I've got the spelling right) back in the early 70's, I've never flown an aircraft before. So I took my curiosity to a nearby flight school and arranged for a short flight. I had originally planned on giving you a detailed comparison of FS2 and the real thing, but the flight, in it's essence, was simply a reflection of the simulator. VFR and radio navigation were identical, at least as far as the mechanics of it. Tuning in a VORTAC was a bit different than it's done in the simulator, in fact, most of the differences between the simulator and actual flying are either those imposed because of the limitations inherent in the medium, or in the stylized or simplified representation of parts of the world of flight.

The instrument panel of the Cessna 172 was virtually identical to the panel depicted in FS2. There were some minor variations, of course, such as the position of the fuel gauges and the compass, and the position indicators, which didn't exist at all. But the panel looked nice and familiar.

The actual flying of the plane was drastically different. Obviously, it would be, and I expected as much. In this respect the simulator is as close to reality as the driving simulators you find in an automobile driving school. The tactile feedback, and the "seat of the pants" feel for the aircraft is something that can't be experienced in the simulation. The results of physical forces acting on the aircraft are something which are seen in the simulation, and *felt* in the real thing. And finally, of course, the controls in the Cessna were infinitely more sensitive than the simulator's. Although my head knew what had to be done (and in that one respect the simulation did help), the actual execution of a few simple maneuvers bore no relationship at all to the computer simulation.

Also, there are a myriad of details which must be attended to in the course of a real flight which are not reflected in the simulator, such as radio communications and air traffic control.

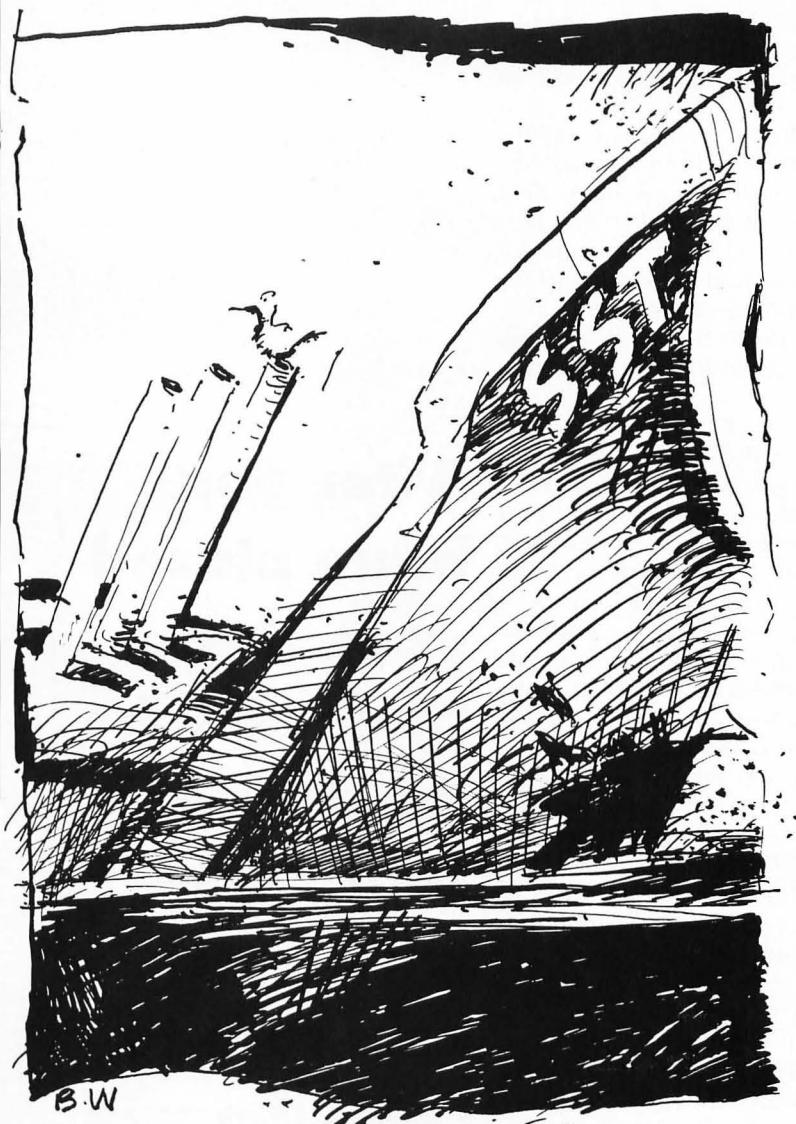
All in all, however, Flight Simulator II is a remarkable piece of work. With a little imagination and some homework, you can get a real close idea of what it's like to fly a small plane. But watch out. It's addictive.

GAME

Battle Blips!

by Patrick Dell'Era

**Low or
Medium
Resolution**



Even the stodgiest, "sophisticated" computer curmudgeon enjoys an occasional game against the computer. But oftentimes the computer becomes too predictable in its play to remain challenging. If only a computer could be programmed to have the irrational quirks of a real human!

Increasingly, gamers want to be able to play against other people instead of just the computer. With **Battle Blips!** (BB) you have just that option. But unlike most other person vs. person games, your human opponent doesn't even have to be in the same country as you!

Howz'at? Well, you see, BB lets you use your modem to play against someone else. What, you don't have a modem yet? That's okay, BB will still play you a pretty tough game against the computer!

BB is a strategy game based on the classic game of Battleship. You and your opponent each have a 10x10-inch grid on which to place your fleet of five ships. Then each tries to sink the other's fleet first. Each ship can take as many hits as it is long. The ships range in length from

two for a Destroyer to five for a Battleship.

BB was written as a desk accessory, so it is available to you at any time you are using a GEM program with the menu bar accessible. You can start up a game, interrupt it to go back to your main program, and then click it on again to continue from the exact place you left off.

To install BB, you must copy it to your boot disk with the extension .ACC. Then turn off your ST for ten seconds, and turn it back on. When you drop down the Desk menu, you should see the title "Battle Blips!"

To start playing, click it on. BB will work in either mono or medium resolution color.

After clicking BB, you will be presented with a menu that allows you to "Play against computer," "Play as HOST against modem," "Play as GUEST against modem," or "Exit Battle Blips! . . ." Move your mouse arrow to your choice and click.

Playing against the computer and exiting are pretty self-evident. They allow you to play the computer only or to return to the desktop.

When you start playing BB, you will see the play screen with two grids separated down the middle with a ships-sunk box. The grid on the left is your opponent's grid, the one on the right is yours. Next to the ship names are small empty boxes. When the named ship is sunk, a check mark will appear in the appropriate box. At the bottom of the screen is the message window. Here's where you will see whose turn it is, if the shot is a hit or a miss, etc.

Whether playing the modem or playing the computer, you must first place your ships on your grid. Move the dark cursor to the square that marks one end of your ship. Use the cursor keys to move around, or optionally, you may use the 1, 2, 3 and 5 keys of the keypad. Note that the keypad option is laid out and used exactly the same as the cursor keys, but places the Enter key right under your little finger so you don't have to move your hand to place a ship or make your shot.

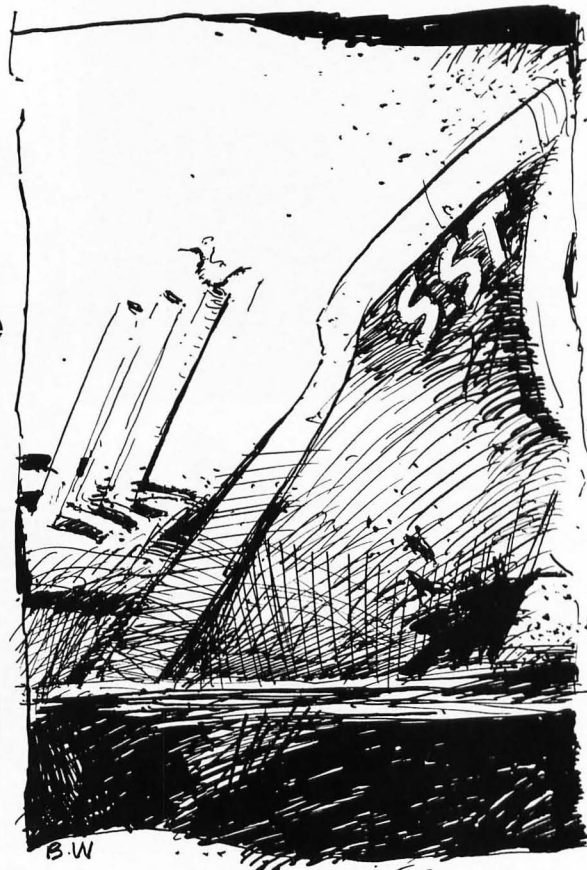
Your cursor will jump to the opposite side of the grid if you move it off an edge. When the cursor is at the desired position, press RETURN or ENTER. If the ship you are placing will fit only down or across, it will be placed in position. If it could fit in both directions, you will be asked to indicate which you want. Press the down arrow (or 2 on the keypad) to indicate down, or press the right arrow (or 3 on the keypad) to indicate across. Pressing RETURN or ENTER is not necessary to indicate the direction to lay the ship.

If the ship you are placing does not fit in either direction, you will be told. Note that another ship may be in the way, not just the edge of the grid. Simply move to another location and try again.

After you have placed your ships, there will be an invisible (but fair—trust me!) coin toss to see who will go first. If you are playing the modem as a Host (don't panic, you'll see what that means...), then your guest will make the call, and you will be told the outcome.

Once you and your opponent have placed your ships on your own grids, it is time to do your best to sink your opponent's fleet before he/she/it sinks yours. When it is your turn, your cursor will be placed in your opponent's grid. Move it to where you want to take a shot and press RETURN or ENTER. On your opponent's turn, you will see him/her/it move his/her/its (oh, the hell with it!) cursor to where she will take her shot. The message box will indicate whose turn it is, whether the shot taken is a hit or a miss, and, if necessary, if a hit has sunk

**After you
have placed
your ships,
there will be
an invisible
(but fair—
trust em!)
coin toss to
see who will
go first.**



the ship.

Squares that have not been shot at have an outlined cross in them. Shots that hit are marked with an oval with an "X" in it (you may recognize it!), and misses are shown with blank squares. If you take a shot at a square that you have already shot at, you will be told. Then you get another chance.

You may exit back to the desktop by pressing Undo. If you are playing the computer alone, when you come back to BB you will have the option of continuing the game or starting a new game. Continued games pick up right where you left off.

Playing the modem takes a little more explanation. First you boot up with BB installed as a desk accessory. Then you enter your terminal program, make a call to your waiting friend (or answer a call from your calling friend), and establish

your modem connection. You may have your terminal parameters set any way you want. BB doesn't care what baud, parity, etc. you choose, just so long as you can communicate with your opponent through your terminal program.

You and your friend need to agree before entering BB who will be the host and who will be the guest. Needless to say, you have to have one of each before you can get anywhere. It's probably a good idea to make the caller the host and the answerer the guest, but it doesn't matter to BB, so long as you do have one of each.

The reason BB was written as a desk accessory is so that you may use your favorite terminal program to make the connection with your modem.

(Let's face it, the terminal program you have and love is far better and more familiar than any kind of simple terminal program I would have put together.)

necting the line. Mi-Term and other terminal programs have the menu bar available at all times.

Once back at the menu bar, drop down the Desk menu and click on Battle Blips. When you see BB's options, click on the appropriate choice (dictated by whether you are the designated host or guest).

Then place your ships on your grid. When both people have completed that, the information will be passed back to the host computer and the guest will choose heads or tails to determine who will start first. Then the play begins!

You may at any time suspend the game and return to the desktop or the terminal program that you launched BB from. Then you and your friend may type messages back and forth, perhaps congratulating each other on a game well played.

You both may re-enter the game in the same way as you started it. Of course, you will be presented with different options from BB. You may continue the game, or exit. You don't have the option of starting a new game at this point, just so someone doesn't "accidentally" make a mistake and trash the game in progress. Re-entering the game puts you back at exactly the same point that you had suspended the game from.

A Few Technical Notes

Battle Blips! is written in Personal Pascal v.1.11 from O.S.S., Inc. It was developed first as a stand-alone program for ease of testing. When it was working well, it was converted to a desk accessory using built-in commands available in Pascal 1.11 or later versions.

You can re-compile the source into a stand-alone application by commenting out the first line:

```
($A+, D-, $20)
```

and commenting out the existing last block and installing the already commented out last block. If you do it right, it really takes only a few keystrokes!

While most of BB is written in fairly standard Pascal parlance (hey, what do I

know?), there are a couple of tricks worth looking at.

The message window used shows an interesting use of an undocumented procedure of Personal Pascal—**Obj_Draw**. With it, you can have the AES draw your dialogs just like in the call **Do_Dialog**. But after it draws the dialog, it returns to the calling program. Then you can manage the dialog yourself.

Check the procedure **Pause**. There you see how to enter Supervisor mode in Personal Pascal and access the timer for precise timing. From Supervisor mode there is no limit to the fun you can have crashing your system!

It was decided to use **BIOS** calls for modem input/output rather than the traditional Pascal system of Reset/Write. While the traditional Pascal means is an excellent learning tool, it just gets in the way (as far as I am concerned) for real applications. Besides, there is no provision for determining whether there is data waiting on the modem without using the **Bconstat** call.

BB should work just fine through networking systems such as CompuServe and Delphi. It doesn't use any but the standard readable ASCII characters for sending data, so the filtering common to such networks would not hinder BB.

To use a network, you and a friend would have to call it at a given time. Then enter into a private conference with each other. Then go for it!

Why would you want to use a network for such a thing? To keep Ma Bell from taking all your money when playing long-distance. Most cities have local access numbers to call the networks. The local call plus the network charges might be lower in some instances than long-distance direct calls.

Check it out.

My thanks to O.S.S. in general and Mark Rose in particular for the little tricks used to coerce Pascal to do what is needed.

After you and your friend have connected and established who will be the guest and who will be the host, you both need to access the menu bar by whatever means is necessary with your terminal software. Flash users will have to press the right mouse button. VT-52 Emulator users will have to press UNDO, dumping you back to the desktop but not discon-

Listing 1: Pascal

```
($A+, D-, $10)
( 06/21/87 20:51:28 )
PROGRAM Battle_Blips ;

CONST
  ($I \p_pascal\GEMCONST.PAS )
  AC_OPEN = 40 ;
  AC_CLOSE = 41 ;

  CR = 13 ;
  UNDO = $6100 ;
  LEFT = $4b00 ; ONE = $6d31 ;
  DOWN = $5000 ; TWO = $6e32 ;
  RIGHT = $4d00 ; THREE = $6f33 ;
  UP = $4800 ; FIVE = $6b35 ;
  RETURN = $1c0d ; ENTER = $720d ;
```

```

DESTRUCTOR = 1 ;
SUBMARINE = 2 ;
CRUISER = 3 ;
BATTLESHIP = 4 ;
A_CARRIER = 5 ;
HUMAN = 0 ;
COMPUTER = 1 ;
TOTAL = 0 ;
SHIPS = 0 ;
SHOTS = 1 ;
NOSHIP = '+' ;
NOSHOT = '+' ;
HIT = 5 ;
MISS = ' ' ;

TYPE
($I \p_pascal\GEMTYPE.PAS )
grid = Packed Array [ 0..9, 0..9, SHIPS..SHOTS ] of Char ;
Str5 = String[5] ;

VAR
msg : message_buffer ;
window, ap_id : Integer ;
title : Str255 ;
dummy : Integer ;
msg_box : Dialog_Ptr ;
msg_idx : Array[0..3] of Integer ;
cmp_grid, hmn_grid : grid ;
strength : Array[ HUMAN..COMPUTER, TOTAL..A_CARRIER ] OF Integer ;
ship_name : Array[ DESTROYER..A_CARRIER ] OF String[16] ;
human_turn : Boolean ;
unresolved_hits : Integer ;
pattern : Boolean ;
hmn_col, hmn_row : Integer ;
cmp_col, cmp_row : Integer ;
rez : Integer ;
cont_game, stop : Boolean ;
play_mode : Char ;

($I \p_pascal\GEMSUBS.PAS )

FUNCTION getrez : Integer ;
XBIO$ ( 4 ) ;

FUNCTION Menu_Register( id : Integer ; VAR name : Str255 ) : Integer ;
EXTERNAL ;

PROCEDURE Obj_Draw( dialog : Dialog_Ptr ;
start, depth,
x, y, w, h : Integer ) ;
EXTERNAL ;

FUNCTION event_key : Integer ;
VAR event, key : Integer ;

BEGIN
event := get_event( E_Keyboard | E_Timer, 0, 0, 0, 0,
FALSE, 0, 0, 0, 0, FALSE, 0, 0, 0, 0,
msg, key, dummy, dummy, dummy, dummy ) ;
IF event & E_Keyboard <> 0 THEN
event_key := key
ELSE
event_key := -1 ;
END ;

FUNCTION bconstat( device : Integer ) : Boolean ;
BIOS( $01 ) ;

FUNCTION bconin( device : Integer ) : Long_Integer ;
BIOS( $02 ) ;

PROCEDURE bconout( device, c : Integer ) ;
BIOS( $03 ) ;

FUNCTION super( x : Long_Integer ) : Long_Integer ;
GEMDOS( $20 ) ;

FUNCTION time : Long_Integer ;
( returns system timer ticks 200/second resolution )

TYPE
Long_pointer = ^long_integer ;

VAR
ssp : Long_Integer ; ( save old supervisor stack pointer )

hz_200 : RECORD
CASE Boolean OF
TRUE : ( l : Long_Integer ) ;
FALSE : ( p : long_pointer ) ;
END ;

BEGIN
ssp := Super( 0 ) ; ( save supervisor stack.. enter super mode )
hz_200.l := $4ba ; ( point at 200 hertz timer )
($p-) ; ( turn pointer checking off )
time := hz_200.p^ ; ( get the longword at that location )
($p-) ; ( restore old value of pointer checking )
ssp := Super( SSP ) ; ( restore supervisor stack... enter user mode )
END ; ( time )

PROCEDURE do_close ;
BEGIN
IF window <> no_window THEN BEGIN
close_window( window ) ;
delete_window( window ) ;
window := No_Window ;
stop := TRUE ;
show_mouse ;
END ;

```

```

END ;

PROCEDURE out_modem( trash : Str5 ) ;
VAR   ctr : Integer ;
BEGIN
  bconout( 1, ord( play_mode ) ) ;
  ctr := 1 ;
  WHILE ctr <= Length( trash ) DO BEGIN
    bconout( 1, ord(trash[ctr]) ) ;
    ctr := ctr+1 ;
  END ;
  bconout( 1, CR ) ;
END ; { out_modem }

PROCEDURE modem_error( error : Integer ) ;
VAR junk : Integer ;
    trash : Str255 ;
BEGIN
  trash := '[I] [Game suspended|by ' ;
  CASE error OF
    0 : trash := Concat( trash, 'opponent' ) ;
    1 : BEGIN
      out_modem( 'SS' ) ;
      trash := Concat( trash, 'you' ) ;
    END ;
    2 : BEGIN
      cont_game := FALSE ;
      trash := '[I] [Fatal data error|during modem input' ;
    END ;
  END ; { case }
  trash := Concat( trash, '.| ][ okay ]' ) ;
  junk := do_alert( trash, 1 ) ;
  do_close ;
END ; { modem_error }

PROCEDURE in_modem( VAR in_str : Str5 ) ;
VAR   ctr : Integer ;
    done : Boolean ;
    key : Integer ;
    s : Char ;
    trash : Str255 ;
    pace : Long_Integer ;
BEGIN
  REPEAT
    pace := time + 1200 ;
    trash := '' ;
    done := FALSE ;
    REPEAT
      key := event_key ;
      done := (key = UNDO) ;
      IF done THEN
        modem_error( 1 )
      ELSE IF bconstat(1) THEN BEGIN
        s := chr( bconin( 1 ) ) ;
        done := ( s = chr(CR) ) ;
        IF NOT done THEN
          IF ( s IN ['0'..'9', 'A'..'Z'] ) THEN
            IF ( Length( trash ) < 254 ) THEN
              trash := Concat( trash, s )
            ELSE
              trash := '' ;
          END
        ELSE IF time > pace THEN BEGIN
          IF in_str = 'OK' THEN BEGIN
            out_modem( in_str ) ;
            pace := time + 1200 ; { 6 seconds }
          END ;
        END ;
      UNTIL done ;
      IF NOT stop THEN BEGIN
        done := (( Length( trash ) > 1 ) AND ( Length( trash ) < 5 ) ) ;
        IF done THEN
          IF play_mode = 'H' THEN
            done := ( trash[1] = 'G' )
          ELSE IF play_mode = 'G' THEN
            done := ( trash[1] = 'H' ) ;
        IF done THEN
          IF in_str <> 'OK' THEN
            done := ( Pos( 'OK', trash ) = 0 ) ;
        END ;
      UNTIL done ;
      IF NOT stop THEN BEGIN
        IF in_str = 'OK' THEN
          out_modem( in_str ) ;
          in_str := Copy( trash, 2, Length(trash)-1 ) ;
        IF in_str = 'SS' THEN
          modem_error( 0 ) ;
        END ;
      END ; { in_modem }

PROCEDURE draw_msg_box( level : Integer ) ;
BEGIN
  Obj_Draw( msg_box, level, level, 0, 0, 640, 640 ) ;
END ;

PROCEDURE message( msg : Str255 ; level : Integer ) ;
BEGIN
  level := msg_idx[level] ;
  set_dtext( msg_box, level, msg, 3, TE_CENTER ) ;
  draw_msg_box( level ) ;
END ;

PROCEDURE wait_opponent ;
VAR   mdmstr : Str5 ;
BEGIN
  IF play_mode <> 'C' THEN BEGIN
    draw_msg_box( 0 ) ;

```

```

mdmstr := 'OK' ;
out_modem( mdmstr ) ;
message( 'Waiting for your opponent...', 2 ) ;
REPEAT
  mdmstr := 'OK' ;
  in_modem( mdmstr ) ;
UNTIL stop OR ( mdmstr = 'OK' ) ;
END ;
END ; ( wait_opponent )

PROCEDURE draw_bkgrnd ;
VAR x,y,w,h : Integer ;
BEGIN
  work_rect( window, x,y,w,h ) ;
  set_clip ( x,y,w,h ) ;
  Draw_mode ( 1 ) ;
  Paint_Style(14) ;
  Paint_Color( BLACK ) ;
  Paint_Rect( x, y, w, h ) ;
END ; ( draw_bkgrnd )

PROCEDURE do_open ;
VAR menu_screen : Dialog_Ptr ;
junk : Integer ;
trash : Str255 ;
menu_offset : Integer ;
ready : Boolean ;

BEGIN
  window := new_window( G_NAME,
    title,
    0, 0, 0, 0 ) ;
  open_window( window, 0, 0, 0, 0 ) ;
  hide_mouse ;
  stop := FALSE ;
  draw_bkgrnd ;

  REPEAT
    ready := TRUE ;
    menu_screen := new_dialog( 9, 0,0,45,20 ) ;

    junk := add_ditem(menu_screen, g_text, none, 0,1,44,1,0,256 ) ;
    set_dtext( menu_screen, junk, 'B a t t l e   B l i p s !', 3, TE_CENTER ) ;
    junk := add_ditem(menu_screen, g_text, none, 0,2,44,1,0,256 ) ;
    set_dtext( menu_screen, junk, 'version 1.5', 3, TE_CENTER ) ;
    trash := Concat( 'Copyright ', chr(189), ' 1987 ' ) ;
    trash := Concat( trash, 'P.L. Dell' Era' ) ;
    junk := add_ditem(menu_screen, g_text, none, 0,3,44,1,0,256 ) ;
    set_dtext( menu_screen, junk, trash, 3, TE_CENTER ) ;

    trash := Concat( 'Portions copyright ', chr(189), ' CCD and OSS, Inc.' ) ;
    junk := add_ditem(menu_screen, g_text, none, 0,4,44,1,0,256 ) ;
    set_dtext( menu_screen, junk, trash, 3, TE_CENTER ) ;

    IF NOT cont_game THEN BEGIN
      menu_offset := add_ditem(menu_screen, g_button,
        SELECTABLE | DEFAULT | TOUCH_EXIT, 8,6,28,2,0,256 ) ;
      set_dtext( menu_screen, menu_offset, 'Play against computer',
        3, TE_LEFT ) ;

      junk := add_ditem(menu_screen, g_button,
        SELECTABLE | TOUCH_EXIT, 8,9,28,2,0,256 ) ;
      set_dtext( menu_screen, junk, 'Play as HOST against modem',
        3, TE_LEFT ) ;
      junk := add_ditem(menu_screen, g_button,
        SELECTABLE | TOUCH_EXIT, 8,12,28,2,0,256 ) ;
      set_dtext( menu_screen, junk, 'Play as GUEST against modem',
        3, TE_LEFT ) ;
    END
  ELSE BEGIN
    menu_offset := add_ditem(menu_screen, g_button,
      SELECTABLE | DEFAULT | TOUCH_EXIT, 8,7,28,2,0,256 ) ;
    set_dtext( menu_screen, menu_offset, 'Continue Game',
      3, TE_LEFT ) ;
    junk := add_ditem(menu_screen, g_button,
      SELECTABLE | TOUCH_EXIT, 8,11,28,2,0,256 ) ;
    set_dtext( menu_screen, junk, 'Start New Game',
      3, TE_LEFT ) ;
    IF play_mode <> 'C' THEN
      Obj_Setstate( menu_screen, junk, DISABLED, FALSE ) ;
    END ;

    junk := add_ditem(menu_screen, g_button,
      SELECTABLE | TOUCH_EXIT, 8,15,28,2,0,256 ) ;

    set_dtext( menu_screen, junk, 'Exit Battle Blips! ...',
      3, TE_LEFT ) ;
    junk := add_ditem(menu_screen, g_text, none, 0,18,44,1,0,256 ) ;
    set_dtext( menu_screen, junk,
      'While playing, press [UNDO] to exit game.', 5, TE_CENTER ) ;

    center_dialog( menu_screen ) ;
    show_mouse ;
    junk := do_dialog( menu_screen, 0 ) ;
    hide_mouse ;
    delete_dialog( menu_screen ) ;

    IF NOT cont_game THEN
      CASE junk - menu_offset OF
        0 : play_mode := 'C' ;
        1 : play_mode := 'H' ;
        2 : play_mode := 'G' ;
        3 : do_close ;
      END ( case )
    ELSE
      CASE junk - menu_offset OF
        0 : ;
        1 : BEGIN
          IF play_mode = 'C' THEN BEGIN
            cont_game := FALSE ;
            ready := FALSE ;
          END ;
        END ;
      END ;
    END ;
  END ;
END ;

```

```

        END ;
    END ;
    2 : do_close ;
END ; ( case )
UNTIL ready ;
IF NOT STOP THEN
    draw_bkgnd ;
END ; ( do_open )

PROCEDURE pause( seconds : Integer ) ;
VAR dummy : Integer ;
    finish : Long_Integer ;
BEGIN ( pause )
    finish := seconds*250 ;
    dummy := get_event( E_Timer, 0, 0, 0, finish,
        FALSE, 0, 0, 0, 0, FALSE, 0, 0, 0, 0,
        msg, dummy, dummy, dummy, dummy, dummy, dummy ) ;
END ;

FUNCTION rand( max : integer ) : integer ;
    FUNCTION random : long_integer ;
        XBIOS ( 17 ) ;
    BEGIN
        IF max = 0 THEN
            rand := 0
        ELSE
            rand := int( random MOD max ) ;
        END ;
    END ;

PROCEDURE init_grids ;
VAR x, y, z : Integer ;
BEGIN
    IF NOT cont_game THEN BEGIN
        FOR x := 0 TO 9 DO
            FOR y := 0 TO 9 DO
                FOR z := SHIPS TO SHOTS DO BEGIN
                    cmp_grid[ x, y, z ] := NOSHOT ;
                    hnn_grid[ x, y, z ] := NOSHOT ;
                END ;
            FOR x := HUMAN TO COMPUTER DO
                FOR y := TOTAL TO A_CARRIER DO BEGIN
                    CASE y OF
                        TOTAL : strength[x,y] := 5 ;
                        DESTROYER : strength[x,y] := 2 ;
                        SUBMARINE : strength[x,y] := 3 ;
                        CRUISER : strength[x,y] := 3 ;
                        BATTLESHIP : strength[x,y] := 4 ;
                        A_CARRIER : strength[x,y] := 5 ;
                    END ; ( case )
                END ;
                unresolved_hits := 0 ;
                pattern := Odd( rand( 100 ) ) ;
                hnn_col := 0 ;
                hnn_row := 0 ;
                cmp_col := 0 ;
                cmp_row := 0 ;
            END ;
        END ; ( init_grids )

PROCEDURE show_sunked ;
VAR ctr, ctr2 : Integer ;
BEGIN
    text_style( THICKENED ) ;
    FOR ctr := HUMAN TO COMPUTER DO
        FOR ctr2 := DESTROYER TO A_CARRIER DO
            IF strength[ ctr, ctr2 ] = 0 THEN
                draw_string( 376-118*ctr, 42*rez+12*rez*(ctr2-1), chr(8) ) ;
            text_style( NORMAL ) ;
        END ; ( show_sunked )

FUNCTION name_rank( c : Char ) : Integer ;
VAR ctr : Integer ;
    trash : String[16] ;
BEGIN
    ctr := 0 ;
    REPEAT
        ctr := ctr+1 ;
        trash := ship_name[ctr] ;
    UNTIL trash[1] = c ;
    name_rank := ctr ;
END ;

PROCEDURE show_symbol( lmrn,x,y : Integer ; k : Char ) ;
BEGIN
    text_style( OUTLINED | THICKENED ) ;
    IF k = NOSHOT THEN
        text_color( GREEN )
    ELSE IF k = chr(HIT) THEN BEGIN
        text_color( RED ) ;
        text_style( NORMAL ) ;
    END
    ELSE
        text_color( BLACK ) ;
    draw_string( lmrn+24*x, 33*rez+12*rez*y,k ) ;
END ; ( show_symbol )

PROCEDURE setup_screen ;
VAR col,row,row2, ctr,ctr2,offset : Integer ;
    x,y,w,h : Integer ;
    ltr : Char ;
    init_str : String[30] ;
    trash : Str255 ;
PROCEDURE show_setup ;
VAR x,y : Integer ;
    c : Char ;

```

```

BEGIN ( show_setup )
  FOR x := 0 TO 9 DO BEGIN
    FOR y := 0 TO 9 DO BEGIN
      c := cmp_grid[ x,y, SHOTS ] ;
      show_symbol( 16, x, y, c ) ;

      IF hmn_grid[ x, y, SHIPS ] <> NOSHIP THEN BEGIN
        IF hmn_grid[ x, y, SHOTS ] = NOSHOT THEN BEGIN
          trash := ship_name[ name_rank(hmn_grid[ x, y, SHIPS ]) ] ;
          c := trash[1] ;
        END
        ELSE
          c := chr(HIT) ;
        END
        ELSE
          c := hmn_grid[ x, y, SHOTS ] ;
          show_symbol( 400, x, y, c ) ;
        END ;
      END ;
    text_color( BLACK ) ;
    text_style( NORMAL ) ;
  END ; ( show_setup )

BEGIN
  draw_bkgrnd ;
  Line_Color(1) ;
  Line_Style(1) ;
  Paint_Style(1) ;
  Paint_Color(0) ;
  Paint_Outline( TRUE ) ;

  Paint_Rect( 8,24*rez,624,120*rez+1 ) ;
  Frame_Rect( 7,23*rez+1,626,121*rez ) ;
  Frame_Rect( 8,24*rez,624,120*rez+1 ) ;
  FOR ctr2 := 1 TO 2*rez DO
    line( 9+ctr2*2,144*rez+ctr2, 635, 144*rez+ctr2 ) ;
  FOR ctr2 := 0 TO 3 DO
    line( 633+ctr2,24*rez+1+ctr2, 633+ctr2,146*rez ) ;

  FOR ctr2 := 0 TO 1 DO BEGIN
    offset := 384*ctr2 ;
    row := 36*rez ;
    col := 32+offset ;
    FOR ctr := 1 TO 9 DO BEGIN
      line( 8+offset,row, 246+offset, row ) ;
      line( col,24*rez+1, col, 144*rez ) ;
      row := row + 12*rez ;
      col := col + 24 ;
    END ;
  END ;
  row := 24*rez+1 ;
  row2 := 144*rez ;
  line( 247,row,247,row2 ) ;
  line( 248,row,248,row2 ) ;

  line( 392,row,392,row2 ) ;
  line( 393,row,393,row2 ) ;

  row := 23*rez+1 ;
  row2 := 120 * rez+1 ;
  Frame_Rect( 250,row,141,row2 ) ;
  Frame_Rect( 251,row+rez-1,139,row2+rez-1 ) ;
  init_str := Concat( chr(4), ' Ships Sunk: ',chr(3) ) ;
  draw_string( 256, 31*rez,init_str ) ;

  row := 33*rez+1 ;
  row2 := 101*rez ;
  line( 272,row,272,row2 ) ;
  line( 273,row,273,row2 ) ;
  line( 368,row,368,row2 ) ;
  line( 369,row,369,row2 ) ;

  row := 33*rez ;
  FOR ctr := 1 TO 4 DO BEGIN
    line( 252,row,390,row ) ;
    IF rez = 2 THEN
      line( 252,row+1,390,row+1 ) ;
      draw_string( 280, row+10*rez, ship_name[ctr] ) ;
      row := row+12*rez ;
    END ;
    line( 252,row,390,row ) ;
    IF rez = 2 THEN
      line( 252,row+1,390,row+1 ) ;
      draw_string( 280, row+10*rez, 'Aircraft' ) ;
      draw_string( 280, row+18*rez, 'Carrier' ) ;
      line( 252,row+20*rez,390,row+20*rez ) ;
      IF rez = 2 THEN
        line( 252,row+20*rez+1,390,row+20*rez+1 ) ;
      line( 252,row+21*rez+1,390,row+21*rez+1 ) ;
      IF rez = 2 THEN
        line( 252,row+22*rez,390,row+22*rez ) ;

text_style( THICKENED ) ;
draw_string( 264, row+34*rez+1, 'O p p o n e n t' ) ;

line( 252,123*rez,390,123*rez ) ;
IF rez = 2 THEN
  line( 252,247,390,247 ) ;
draw_string( 296, 135*rez, 'Y o u' ) ;

text_style( NORMAL ) ;
draw_string( 252, row+34*rez+1, chr(4) ) ;
draw_string( 380, 135*rez, chr(3) ) ;
show_setup ;
show_sunked ;
END ; ( setup_screen )

```

```

PROCEDURE human_input( VAR c,r : Integer; lmargn: Integer ) ;
VAR key : Integer ;
    offset : Integer ;
    mdmstr : Str5 ;
    x_inc,y_inc : Integer ;

PROCEDURE paint_cursor( mode : Integer ) ;
VAR x,y,w, h : Integer ;
BEGIN
    draw_mode( mode ) ;
    x := offset+c*24 ;
    y := r*12*rez + 24*rez + 2 ;
    w := 19 (8*3-5) ;
    IF rez = 1 THEN
        h := 9
    ELSE
        h := 21 ;
    paint_rect(x,y,w,h) ;

    IF x_inc + y_inc <> 0 THEN BEGIN
        c := c + x_inc ;
        r := r + y_inc ;
        IF c < 0 THEN c := 9
        ELSE IF c > 9 THEN c := 0
        ELSE IF r < 0 THEN r := 9
        ELSE IF r > 9 THEN r := 0 ;

        x := offset+c*24 ;
        y := r*12*rez + 24*rez + 2 ;
        paint_rect(x,y,w,h) ;
    END ;
END ;

BEGIN
    paint_style(1) ;
    Paint_Outline( FALSE ) ;
    offset := lmargn*384 + 11 ;
    x_inc := 0 ;
    y_inc := 0 ;
    paint_cursor( 3 ) ;
    REPEAT
        x_inc := 0 ;
        y_inc := 0 ;
        key := event_key ;
        IF bconstat(1) THEN
            IF chr(bconin(1)) = 'S' THEN
                modem_error( 0 ) ;

            IF NOT STOP THEN BEGIN
                IF (key = LEFT) OR (key = ONE) THEN
                    x_inc := -1
                ELSE IF (key = RIGHT) OR (key = THREE) THEN
                    x_inc := 1
                ELSE IF (key = DOWN) OR (key = TWO) THEN
                    y_inc := 1
                ELSE IF (key = UP) OR (key = FIVE) THEN
                    y_inc := -1
                ELSE IF key = UNDO THEN
                    IF play_mode = 'C' THEN
                        do_close
                    ELSE
                        modem_error( 1 )
                    ELSE key := key & $0f ;
                    IF x_inc + y_inc <> 0 THEN BEGIN
                        paint_cursor( 3 ) ;
                        IF ( NOT STOP) AND ( play_mode <> 'C') AND ( lmargn = 0 ) AND
                            ( key <> CR ) THEN BEGIN
                            mdmstr := Concat(chr( ord('0') | c ),chr( ord('0') | r ) ) ;
                            out_modem( mdmstr ) ;
                        END ;
                    END ;
                UNTIL ( key = CR ) OR stop ;
                IF NOT stop THEN
                    paint_cursor( 1 ) ;
                draw_mode(1) ;
            END ; ( human_input )

PROCEDURE human_setup ;
VAR ship, ship_length, ctr : Integer ;
    ok, across : Boolean ;
    col,row,x, y : Integer ;
    trash, trash2 : Str255 ;
    c : Char ;
    key : Long_Integer ;

FUNCTION room_available( x_inc, y_inc : Integer ) : Integer ;
VAR ctr, x, y : Integer ;
    done : Boolean ;
BEGIN
    x := col ;
    y := row ;
    ctr := 0 ;
    REPEAT
        done := (x>9) OR (y>9) ;
        IF NOT done THEN BEGIN
            done := ( hmn_grid[x,y,SHIPS] <> NOSHIP ) ;
            IF NOT done THEN BEGIN
                ctr := ctr + 1 ;
                x := x + x_inc ;
                y := y + y_inc ;
            END ;
        END ;
    UNTIL done ;
    room_available := ctr ;
END ; ( room_available )

BEGIN ( human_setup )
    draw_msg_box( 0 ) ;
    draw_msg_box( msg_idx[0] ) ;

```

```

col := 0 ;
row := 0 ;
trash2 := 'Position your ships:' ;
FOR ship := DESTROYER TO A-CARRIER DO BEGIN
  IF NOT stop THEN BEGIN
    CASE ship OF
      DESTROYER : ship_length := 2 ;
      SUBMARINE : ship_length := 3 ;
      Otherwise : ship_length := ship ;
    END ; { case }
    REPEAT
      trash := Concat( ship_name[ship], ' is ', chr(ship_length+48),
        ' units long.' ) ;
      message( trash2, 1 ) ;
      message( trash, 2 ) ;
      obj_setstate( msg_box, msg_idx[3], NORMAL, FALSE ) ;
      human_input( col, row, 1 ) ;
      IF NOT stop THEN BEGIN
        ok := TRUE ;
        across := ( room_available( 1, 0 ) >= ship_length ) ;
        IF across THEN BEGIN
          IF room_available( 0, 1 ) >= ship_length THEN BEGIN
            trash := Concat( '[', chr(3), ']across or ' ) ;
            trash := Concat( trash, '[', chr(2), ']down?' ) ;
            message( trash, 3 ) ;
            REPEAT
              key := event_key ;
            UNTIL ( key = RIGHT ) OR ( key = THREE )
              OR ( key = DOWN ) OR ( key = TWO ) ;
            across := ( key = RIGHT ) OR ( key = THREE ) ;
          END
        END
        ELSE IF room_available( 0, 1 ) < ship_length THEN
          ok := FALSE ;
        IF NOT ok THEN BEGIN
          show_symbol( 400, col, row, hmn_grid[ col, row, SHIPS ] ) ;
          text_style( NORMAL ) ;
          text_color( BLACK ) ;
          trash2 := ' Ship can''t fit there. ' ;
          obj_setstate( msg_box, msg_idx[3], SELECTED, FALSE ) ;
          message( trash2, 3 ) ;
        END
        ELSE
          message( ' ', 3 ) ;
          trash2 := 'Next ship:' ;
        END ;
      UNTIL ok OR stop ;
      IF NOT stop THEN BEGIN
        trash := ship_name[ ship ] ;
        c := trash[1] ;
        text_style( OUTLINED | THICKENED ) ;
        FOR ctr := 0 TO ship_length-1 DO
          IF ACROSS THEN BEGIN
            hmn_grid[ col+ctr, row, SHIPS ] := c ;
            show_symbol( 400, col+ctr, row, c ) ;
          END
          ELSE BEGIN
            hmn_grid[ col, row+ctr, SHIPS ] := c ;
            show_symbol( 400, col, row+ctr, c ) ;
          END ;
          text_style( NORMAL ) ;
        END ;
      END ;
    END ;
  END ; { human_setup }
END ;

PROCEDURE computer_setup ;
VAR ship, ship_length, ctr : Integer ;
trash : String ;
x, y : Integer ;
mdmstr : Str5 ;

PROCEDURE set_position ;
VAR ctr : Integer ;
ok, across : Boolean ;
col, row, x, y, x-inc, y-inc : Integer ;
c : Char ;
adjacent : Integer ;

PROCEDURE check_adjacent( cax, cay : Integer ) ;
BEGIN
  IF (cax>=0) AND (cay>=0) AND (cax<=9) AND (cay<=9) THEN
    IF cmp_grid[ cax, cay, SHIPS ] <> NOSHIP THEN
      adjacent := adjacent + 1 ;
    END ; { check_adjacent }
  END ;

BEGIN { set_position }
  REPEAT
    across := Odd(rand(100)) ;
    x-inc := 0 ;
    y-inc := 0 ;

    row := rand(10) ;
    col := rand(10) ;
    IF across THEN BEGIN
      ok := ( col + ship_length < 10 ) ;
      x-inc := 1 ;
    END
    ELSE BEGIN
      ok := ( row + ship_length < 10 ) ;
      y-inc := 1 ;
    END ;
    IF ok THEN BEGIN
      x := col ;
      y := row ;
      FOR ctr := 1 TO ship_length DO BEGIN
        IF cmp_grid[ x, y, SHIPS ] <> NOSHIP THEN
          ok := FALSE ;
          x := x + x-inc ;
          y := y + y-inc ;
        END ;
      END ;
    END ;
  UNTIL ok ;
END ;

```

```

END ;
END ;
IF ok THEN BEGIN
    adjacent := 0 ;
    x := col-x_inc ;
    y := row-y_inc ;
    check_adjacent( x, y ) ;

    x := col + x_inc*( ship_length-1 ) ;
    y := row + y_inc*( ship_length-1 ) ;
    check_adjacent( x, y ) ;

    x := col ;
    y := row ;
    FOR ctr := 1 TO ship_length DO BEGIN
        check_adjacent( x-y_inc, y-x_inc ) ;
        check_adjacent( x+y_inc, y+x_inc ) ;
        x := x + x_inc ;
        y := y + y_inc ;
    END ;
    ok := ( adjacent = 0 ) ;
    IF NOT ok THEN
        ok := ( rand(100) > adjacent + 92 ) ;
END ;
UNTIL ok ;
trash := ship_name[ ship ] ;
c := trash[1] ;
FOR ctr := 0 TO ship_length-1 DO
    IF ACROSS THEN
        cmp_grid[ col+ctr, row, SHIPS ] := c
    ELSE
        cmp_grid[ col, row+ctr, SHIPS ] := c ;
END ; ( set_position )

PROCEDURE send_hmn_grid ;
VAR x, y : Integer ;
BEGIN
    FOR x := 0 TO 9 DO
        FOR y := 0 TO 9 DO
            IF hmn_grid[ x,y, SHIPS ] <> NOSHIP THEN BEGIN
                mdmstr := Concat( chr( x | ord('0') ),
                                chr( y | ord('0') ),
                                hmn_grid[ x,y, SHIPS ] ) ;
                out_modem( mdmstr ) ;
            END ;
        END ;
    END ; ( send_hmn_grid )

PROCEDURE receive_cmp_grid ;
VAR x, y, ctr : Integer ;
BEGIN
    trash := 'Receiving opponent's fleet location.' ;
    message( trash, 2 ) ;
    FOR ctr := 1 TO 17 DO
        IF NOT stop THEN BEGIN
            in_modem( mdmstr ) ;
            IF NOT stop THEN BEGIN
                x := ord( mdmstr[1] ) & $f ;
                y := ord( mdmstr[2] ) & $f ;
                cmp_grid[ x,y, SHIPS ] := mdmstr[3] ;
            END ;
        END ;
    END ; ( receive_cmp_grid )

BEGIN ( computer_setup )
    IF play_mode = 'G' THEN BEGIN
        send_hmn_grid ;
        receive_cmp_grid ;
    END
    ELSE IF play_mode = 'H' THEN BEGIN
        receive_cmp_grid ;
        IF NOT stop THEN
            send_hmn_grid ;
    END
    ELSE FOR ship := DESTROYER TO A_CARRIER DO BEGIN
        IF ship < CRUISER THEN
            ship_length := ship+1
        ELSE
            ship_length := ship ;
        set_position ;
    END ;
END ; ( computer_setup )

PROCEDURE coin_flip ;
VAR coin_is_heads : Boolean ;
    call_is_heads : Boolean ;
    key : Char ;
    mdmstr : Str5 ;
    trash, trash2 : String ;

PROCEDURE h_or_t( its_h : Boolean ) ;
BEGIN
    IF its_h THEN
        trash := Concat( trash, 'Heads.' )
    ELSE
        trash := Concat( trash, 'Tails.' ) ;
END ; ( h_or_t )

BEGIN
    draw_msg_box( 0 ) ;
    trash := 'Let's flip a coin to see who' ;
    message( trash, 1 ) ;
    trash := 'starts.' ;
    IF play_mode <> 'H' THEN BEGIN
        trash := Concat( trash, 'Do you want Heads or Tails?' ) ;
        message( trash, 2 ) ;
        message( 'Press [H] or [T]...', 3 ) ;
        REPEAT
            key := chr( event_key & $f ) ;
        UNTIL key IN [ 'Y', 'H' ] ;
        IF play_mode = 'G' THEN BEGIN

```

```

    mdmstr := Concat( key ) ;
    out_modem( mdmstr ) ;
END ;
ELSE BEGIN
    trash := Concat( trash, 'Your opponent makes the call.' ) ;
    message( trash, 2 ) ;
    REPEAT
        in_modem( mdmstr ) ;
        IF NOT stop THEN
            key := mdmstr[1] ;
    UNTIL (key in ['T', 'H' ]) OR stop ;
END ;
IF NOT stop THEN BEGIN
    call_is_heads := ( key = 'H' ) ;
    draw_msg_box( 0 ) ;
    trash := ( 'You' ) ;
    IF play_mode = 'H' THEN
        trash := Concat( trash, 'r opponent' ) ;
    trash := Concat( trash, ' called' ) ;
    h_or_t( call_is_heads ) ;
    message( trash, 1 ) ;
    trash := ' it's ' ;
    IF play_mode <> 'G' THEN BEGIN
        coin_is_heads := Odd(rand(100)) ;
        IF play_mode = 'H' THEN
            IF coin_is_heads THEN
                out_modem( 'H' )
            ELSE
                out_modem( 'T' ) ;
        END
    ELSE BEGIN
        in_modem( mdmstr ) ;
        coin_is_heads := ( mdmstr = 'H' ) ;
    END ;
    h_or_t( coin_is_heads ) ;

    IF play_mode = 'H' THEN
        human_turn := ( coin_is_heads AND NOT call_is_heads ) OR
            ( NOT coin_is_heads AND call_is_heads )
    ELSE
        human_turn := ( coin_is_heads AND call_is_heads ) OR
            ( NOT coin_is_heads AND NOT call_is_heads ) ;

    IF human_turn THEN
        trash := Concat( trash, ' You go first.' )
    ELSE BEGIN
        IF play_mode = 'H' THEN
            trash := Concat( trash, ' Your opponent starts.' )
        ELSE
            trash := Concat( trash, ' I'll start first.' ) ;
    END ;
    IF (human_turn AND ( play_mode = 'H' )) OR
        ( NOT ( human_turn ) AND NOT( play_mode = 'H' ) ) THEN
        trash2 := 'But,'
    ELSE
        trash2 := 'And' ;
    Insert( trash2, trash, 1 ) ;
    message( trash, 2 ) ;
    pause( 12 ) ;
END ;
END ; ( coin_flip )

PROCEDURE do_human_turn ;
VAR ship, ship_length, ctr : Integer ;
    ok, across : Boolean ;
    k : Integer ;
    trash, trash2 : Str255 ;
    mdmstr : Str5 ;

BEGIN
    draw_msg_box( 0 ) ;
    draw_msg_box( msg_idx[0] ) ;
    trash2 := ' ' ;
    REPEAT
        message( trash2, 3 ) ;
        message( 'Fire away!', 1 ) ;
        obj_setstate( msg_box, msg_idx[3], NORMAL, FALSE ) ;

        human_input( hmn_col, hmn_row, 0 ) ;
    IF NOT stop THEN BEGIN
        ok := cmp_grid( hmn_col, hmn_row, SHOTS ) = NOSHOT ;
        IF ok THEN BEGIN
            IF play_mode <> 'C' THEN BEGIN
                mdmstr := 'F' ;
                out_modem( mdmstr ) ;
            END ;
            IF cmp_grid [ hmn_col, hmn_row, SHIPS ] <> NOSHIP THEN BEGIN
                cmp_grid[ hmn_col, hmn_row, SHOTS ] := chr(HIT) ;
                trash := 'It's a hit!!!' ;
                trash2 := ' ' ;
                show_symbol( 16, hmn_col, hmn_row, chr(HIT) ) ;
                text_color( BLACK ) ;
                message( trash2, 3 ) ;
                message( trash, 2 ) ;
                k := name_rank( cmp_grid[ hmn_col, hmn_row, SHIPS ] ) ;
                strength[ COMPUTER, k ] := strength[ COMPUTER, k ] - 1 ;
                IF strength[ COMPUTER, k ] = 0 THEN BEGIN
                    trash := Concat( 'You sank my ', ship_name[k], '...' ) ;
                    show_sunked ;
                    message( trash, 3 ) ;
                    strength[ COMPUTER, TOTAL ] := strength[ COMPUTER, TOTAL ] - 1 ;
                END ;
            END
        ELSE BEGIN
            trash2 := ' ' ;
            message( trash2, 3 ) ;
            message( 'Your shot misses...', 2 ) ;
            cmp_grid [ hmn_col, hmn_row, SHOTS ] := MISS ;
        END ;
    END ;
END

```

```

ELSE BEGIN
  trash2 := ' You've shot there already! ' ;
  obj_setstate( msg_box, msg_idx[3], SELECTED, FALSE ) ;
  IF cmp_grid[ hmn_col, hmn_row, SHIPS ] <> NOSHIP THEN
    show_symbol( 16, hmn_col, hmn_row, chr( HIT ) ) ;
    text_color( BLACK ) ;
  END ;
END ;
UNTIL ok OR stop ;
IF NOT stop THEN BEGIN
  pause( 12 ) ;
  human_turn := FALSE ;
END ;
END ; ( do_human_turn )

PROCEDURE do_computer_turn ;
VAR longest, shortest : Integer ;
x, y : Integer ;
done : Boolean ;
mdmstr : Str5 ;

FUNCTION adjacent : Integer ;
VAR ctr, col, row : Integer ;
BEGIN
  ctr := 0 ;
  FOR col := x-1 TO x+1 DO
    FOR row := y-1 TO y+1 DO
      IF ( col >= 0 ) AND ( col <= 9 ) THEN
        IF ( row >= 0 ) AND ( row <= 9 ) THEN
          IF ( hmn_grid[ col, row, SHOTS ] = chr( HIT ) ) OR
             ( ( hmn_grid[ col, row, SHIPS ] <> NOSHIP ) AND
               ( hmn_grid[ col, row, SHOTS ] <> NOSHOT ) ) THEN
            ctr := ctr + 1 ;
          adjacent := ctr ;
        END ;
      END ; ( adjacent )

FUNCTION untried( xx, yy, flag : Integer ) : Integer ;
VAR ctr, ctr2, col, row : Integer ;
done : Boolean ;

BEGIN
  ctr := 1 ;
  FOR ctr2 := 0 DOWNT0 flag DO BEGIN
    col := x ;
    row := y ;
    REPEAT
      row := row + yy ;
      col := col + xx ;
      done := ( col > 9 ) OR ( row > 9 ) OR ( col < 0 ) OR ( row < 0 ) ;
      IF NOT done THEN BEGIN
        done := ( hmn_grid[ col, row, SHOTS ] <> NOSHOT ) ;
        IF NOT done THEN
          ctr := ctr + 1 ;
      END ;
    UNTIL done ;
    xx := xx * (-1) ;
    yy := yy * (-1) ;
  END ;
  untried := ctr ;
END ; ( untried )

PROCEDURE random_srch ;
VAR index : Array [ 0..99 ] OF Integer ;
i, ctr : Integer ;
selective : Boolean ;

BEGIN ( random_srch )
  selective := TRUE ;
  ctr := -1 ;
  WHILE ctr = -1 DO BEGIN
    FOR i := 0 TO 99 DO BEGIN
      x := i MOD 10 ;
      y := i DIV 10 ;
      IF NOT selective OR ( Odd( x + y ) = pattern ) THEN
        IF ( hmn_grid[ x, y, SHOTS ] = NOSHOT ) THEN BEGIN
          IF selective THEN BEGIN
            done := ( adjacent = 0 ) ;
            IF NOT done THEN
              done := ( rand( 100 ) > 90 ) ;
          END
        ELSE
          done := TRUE ;
        IF done THEN BEGIN
          IF untried( 1, 0, -1 ) >= longest THEN BEGIN
            ctr := ctr + 1 ;
            index[ctr] := i ;
          END ;
          IF untried( 0, 1, -1 ) >= longest THEN BEGIN
            ctr := ctr + 1 ;
            index[ctr] := i ;
          END ;
        END ;
      END ;
    END ;
    selective := FALSE ;
  END ; ( while )
  i := rand( ctr + 1 ) ;
  i := index[i] ;
  x := i MOD 10 ;
  y := i DIV 10 ;
END ; ( random_srch )

PROCEDURE next_hit ;
CONST LOOK_LEFT = 0 ;
LOOK_RIGHT = 1 ;
LOOK_UP = 2 ;
LOOK_DOWN = 3 ;

VAR col, row, ctr, ctr2 : Integer ;
col2, row2, x_inc, y_inc : Integer ;

```

```

inc, start_xy : Integer ;
done, reverse : Boolean ;

PROCEDURE rand_pick( col, row : Integer ) ;
VAR ctr, ctr2, direction : Integer ;
    srch_pattern, temp : Array[ 0..3 ] OF Integer ;
BEGIN ( rand_pick )
    FOR ctr := LOOK_LEFT TO LOOK_DOWN DO
        temp[ctr] := ctr ;
    FOR ctr := LOOK_DOWN DOWNTO LOOK_LEFT DO BEGIN
        direction := rand( ctr+1 ) ;
        srch_pattern[ ctr ] := temp[ direction ] ;
        FOR ctr2 := direction TO LOOK_UP DO
            temp[ctr2] := temp[ctr2 + 1] ;
        END ;
        x := col ;
        y := row ;
        ctr := -1 ;
        REPEAT
            ctr := ctr+1 ;
            x_inc := 0 ;
            y_inc := 0 ;
            CASE srch_pattern[ ctr ] OF
                LOOK_LEFT : x_inc := -1 ;
                LOOK_RIGHT : x_inc := 1 ;
                LOOK_UP : y_inc := -1 ;
                LOOK_DOWN : y_inc := 1 ;
            END ; ( case )
            UNTIL ( untried( x_inc, y_inc, -1 ) >= shortest ) AND
                ( untried( x_inc, y_inc, 0 ) > 1 ) ;
            x := x + x_inc ;
            y := y + y_inc ;
        END ; ( rand_pick )

PROCEDURE get_next_hit( VAR c, r : Integer ) ;
BEGIN
    REPEAT
        REPEAT
            done := ( hmn_grid[ c, r, SHOTS ] = chr(HIT) ) ;
            IF NOT done THEN
                c := c + inc ;
            UNTIL ( c > 9 ) OR ( c < 0 ) OR done ;
            IF NOT done THEN BEGIN
                r := r + inc ;
                c := start_xy ;
            END ;
        UNTIL done ;
    END ; ( get_next_hit )

BEGIN ( next_hit )
    IF Odd( rand(100) ) THEN BEGIN
        inc := 1 ;
        start_xy := 0 ;
    END
    ELSE BEGIN
        inc := -1 ;
        start_xy := 9 ;
    END ;
    row := start_xy ;
    col := start_xy ;
    get_next_hit( col, row ) ;
    IF unresolved_hits = 1 THEN
        rand_pick( col, row )
    ELSE BEGIN
        col2 := col + inc ;
        row2 := row ;
        IF ( col2 > 9 ) OR ( col2 < 0 ) THEN BEGIN
            col2 := start_xy ;
            row2 := row2 + inc ;
        END ;
        get_next_hit( col2, row2 ) ;
        IF col2 = col THEN BEGIN
            REPEAT
                row2 := row2 + inc ;
                done := ( row2 > 9 ) OR ( row2 < 0 ) ;
                reverse := done ;
                IF NOT done THEN BEGIN
                    done := ( hmn_grid[ col2, row2, SHOTS ] <> chr(HIT) ) ;
                    reverse := ( hmn_grid[ col2, row2, SHOTS ] = MISS ) ;
                END ;
            UNTIL done ;
            IF reverse THEN
                REPEAT
                    row2 := row2 - inc ;
                    done := ( row2 < 0 ) OR ( row2 > 9 ) ;
                    IF NOT done THEN BEGIN
                        done := ( hmn_grid[ col2, row2, SHOTS ] <> chr(HIT) ) ;
                        reverse := ( hmn_grid[ col2, row2, SHOTS ] = MISS ) ;
                    END ;
                UNTIL done ;
            IF reverse THEN
                rand_pick( col, row )
            ELSE BEGIN
                x := col2 ;
                y := row2 ;
            END ;
        END
        ELSE BEGIN
            REPEAT
                col2 := col2 + inc ;
                done := ( col2 > 9 ) OR ( col2 < 0 ) ;
                reverse := done ;
                IF NOT done THEN BEGIN
                    done := ( hmn_grid[ col2, row2, SHOTS ] <> chr(HIT) ) ;
                    reverse := ( hmn_grid[ col2, row2, SHOTS ] = MISS ) ;
                END ;
            UNTIL done ;
            IF reverse THEN
                REPEAT
                    col2 := col2 - inc ;

```

```

done := ( col2 < 0 ) OR ( col2 > 9 ) ;
IF NOT done THEN BEGIN
  done := ( hmn_grid[ col2, row2, SHOTS ] <> chr(HIT) ) ;
  reverse := ( hmn_grid[ col2, row2, SHOTS ] = MISS ) ;
END ;
UNTIL done ;
IF reverse THEN
  rand_pick( col, row )
ELSE BEGIN
  x := col2 ;
  y := row2 ;
END ;
END ;
END ; ( next_hit )

PROCEDURE do_shot ;
VAR ship, ship_length, ctr, col, row, x_dir, y_dir : Integer ;
trash : Str255 ;

FUNCTION wrap_around( a,b : Integer ) : Integer ;
BEGIN
  IF Abs( a-b ) > 5 THEN
    wrap_around := -1
  ELSE
    wrap_around := 1 ;
  END ; ( wrap_around )

PROCEDURE paint_cursor( mode : Integer ) ;
VAR x,y,w, h : Integer ;
BEGIN
  IF NOT stop THEN BEGIN
    draw_mode( mode ) ;
    x := 395+cmp_col*24 ;
    y := cmp_row*12*rez + 24*rez + 2 ;
    w := 19 ;
    IF rez = 1 THEN
      h := 9
    ELSE
      h := 21 ;
    paint_rect(x,y,w,h) ;
  END ;
END ; ( paint_cursor )

BEGIN ( do_shot )
  paint_style(1) ;
  Paint_Outline( FALSE ) ;
  CASE rand(3) OF
    0: trash := 'Hmmm, let's try...' ;
    1: trash := 'Here's my shot.' ;
    2: trash := 'Okay then, how about this?' ;
  END ; ( case )
  message( trash,1 ) ;
  paint_cursor( 3 ) ;
  IF play_mode = 'C' THEN REPEAT
    pause( 2 ) ;
    paint_cursor( 3 ) ;
    x_dir := wrap_around( x, cmp_col ) ;
    y_dir := wrap_around( y, cmp_row ) ;
    IF (cmp_col <> x) AND (cmp_row <> y) THEN BEGIN
      IF Odd( rand(100) ) THEN BEGIN
        IF x > cmp_col THEN
          cmp_col := cmp_col + x_dir
        ELSE
          cmp_col := cmp_col - x_dir ;
      END
      ELSE BEGIN
        IF y > cmp_row THEN
          cmp_row := cmp_row + y_dir
        ELSE
          cmp_row := cmp_row - y_dir ;
      END ;
    END ;
    ELSE IF x > cmp_col THEN cmp_col := cmp_col + x_dir
    ELSE IF x < cmp_col THEN cmp_col := cmp_col - x_dir
    ELSE IF y > cmp_row THEN cmp_row := cmp_row + y_dir
    ELSE
      cmp_row := cmp_row - y_dir ;

    IF cmp_col > 9 THEN cmp_col := 0
    ELSE IF cmp_col < 0 THEN cmp_col := 9
    ELSE IF cmp_row > 9 THEN cmp_row := 0
    ELSE IF cmp_row < 0 THEN cmp_row := 9 ;
    paint_cursor( 3 ) ;
  UNTIL ((cmp_col = x) AND (cmp_row = y)) OR stop
  ELSE BEGIN
    x := -1 ;
    REPEAT
      in_modem( ndmstr ) ;
      paint_cursor( 3 ) ;
      IF NOT stop THEN BEGIN
        IF Length( ndmstr ) = 2 THEN BEGIN
          cmp_col := ord( ndmstr[1] ) - ord( '0' ) ;
          cmp_row := ord( ndmstr[2] ) - ord( '0' ) ;
          IF (cmp_col < 0 ) OR ( cmp_col > 9 ) OR
            (cmp_row < 0 ) OR ( cmp_row > 9 ) THEN
            modem_error( 2 )
          END
          ELSE IF ndmstr = 'F' THEN BEGIN
            x := cmp_col ;
            y := cmp_row ;
          END
          ELSE
            modem_error( 2 ) ;
          paint_cursor( 3 ) ;
        END ;
      UNTIL ( x <> -1 ) OR stop ;
    END ;
  IF NOT stop THEN BEGIN
    IF play_mode = 'C' THEN

```

```

    pause( 4 ) ;

paint_cursor( 1 ) ;
draw_model(1) ;
IF hmn_grid[ x, y, SHIPS ] <> NOSHIP THEN BEGIN
    text_color( RED ) ;
    draw_string( 400+24*cmp_col, 33*rez+12*rez*cmp_row, chr(5) ) ;
    text_color( BLACK ) ;
    unresolved_hits := unresolved_hits + 1 ;
    ship := name_rank( hmn_grid[ x, y, SHIPS ] ) ;
    strength[HUMAN, ship] := strength[HUMAN, ship]-1 ;
    hmn_grid[ x, y, SHOTS ] := chr(HIT) ;
    trash := 'It's a hit!!' ;
    message(trash,2) ;

    IF strength[HUMAN, ship]=0 THEN BEGIN
        show_sunked ;
        trash := Concat( 'Your ', ship_name[ship], ' has been sunk!' ) ;
        message(trash,3) ;
        strength[HUMAN, TOTAL] := strength[HUMAN, TOTAL]-1 ;

        trash := ship_name[ ship ] ;
        FOR col := 0 TO 9 DO
            FOR row := 0 TO 9 DO
                IF hmn_grid[ col, row, SHIPS ] = trash[1] THEN BEGIN
                    hmn_grid[ col, row, SHOTS ] := MISS ;
                    unresolved_hits := unresolved_hits - 1 ;
                END ;
            END ;
        END ;
    ELSE BEGIN
        hmn_grid[ x, y, SHOTS ] := MISS ;
        trash := 'It's a miss...' ;
        message(trash,2) ;
    END ;
    pause( 12 ) ;
END ;
END ; ( do_shot )

BEGIN ( do_computer_turn )
    draw_msg_box( 0 ) ;
    IF play_mode = 'C' THEN BEGIN
        longest := A_CARRIER ;
        WHILE strength[HUMAN, longest] = 0 DO
            longest := longest-1 ;
        shortest := DESTROYER ;
        WHILE strength[HUMAN, shortest] = 0 DO
            shortest := shortest+1 ;
        IF shortest < CRUISER THEN
            shortest := shortest+1 ;
        IF longest < CRUISER THEN
            longest := longest + 1 ;

        IF unresolved_hits <> 0 THEN
            next_hit
        ELSE
            random_srch ;
        END ;
        IF NOT stop THEN BEGIN
            do_shot ;
            IF NOT stop THEN
                human_turn := TRUE ;
            END ;
        END ; ( do_computer_turn )

PROCEDURE show_computer_setup ;
VAR k,x,y : Integer ;
BEGIN
    FOR x := 0 TO 9 DO
        FOR y := 0 TO 9 DO
            IF cmp_grid[ x, y, SHIPS ] <> NOSHIP THEN
                IF cmp_grid[ x, y, SHOTS ] <> chr(HIT) THEN
                    show_symbol( 16,x,y, cmp_grid[ x, y, SHIPS ] ) ;
                text_style( NORMAL ) ;
            END ; ( show_computer_setup )

PROCEDURE play_game ;
VAR key : Char ;
    msg1, msg2 : Str255 ;

BEGIN
    rez := getrez ;
    IF rez = 0 THEN BEGIN
        msg1 := '[3][ Battle Blips must run in! ]' ;
        msg1 := Concat( msg1, 'medium or high resolution. ] [Sorry!]' ) ;
        rez := do_alert( msg1, 1 ) ;
    END
    ELSE BEGIN
        do_open ;
        IF NOT stop THEN BEGIN
            REPEAT
                init_grids ;
                setup_screen ;
                IF cont_game THEN
                    wait_opponent
            ELSE BEGIN
                human_setup ;
                IF NOT stop THEN BEGIN
                    wait_opponent ;
                IF NOT stop THEN BEGIN
                    computer_setup ;
                    IF NOT stop THEN
                        coin_flip ;
                END ;
            END ;
        END ;
        IF NOT stop THEN
            cont_game := TRUE ;
        REPEAT
            IF NOT stop THEN

```

```

        IF human_turn THEN
            do_human_turn
        ELSE
            do_computer_turn ;
        UNTIL ( strength[ COMPUTER, TOTAL ] = 0 ) OR
              ( strength[ HUMAN, TOTAL ] = 0 ) OR
              stop ;

    IF NOT stop THEN BEGIN
        cont_game := FALSE ;
        draw_msg_box( 0 ) ;
        IF strength[COMPUTER,TOTAL] = 0 THEN BEGIN
            msg1 := 'Congratulations! You win!!' ;
            msg2 := ' ' ;
        END
        ELSE BEGIN
            msg1 := 'Sorry, you lose...' ;
            msg2 := 'Here's the rest of my fleet.' ;
            show_computer_setup ;
        END ;
        message( msg1, 1 ) ;
        message( msg2, 2 ) ;
        WHILE event_key < -1 DO ; {clear keypresses}
        IF play_mode = 'C' THEN BEGIN
            message('Care to play again? (Y/N)', 3 ) ;
            REPEAT
                key := chr( event_key & $5f ) ;
            UNTIL key IN ['Y','N', chr( CR ) ] ;
        END
        ELSE BEGIN
            message('Press any key to exit...', 3 ) ;
            WHILE event_key = -1 DO ; {wait for a keypress}
                key := 'N' ;
            END ;
        END ;
        UNTIL (key <> 'Y') OR stop ;
    END ;
    do_close ;
END ; { play_game }

PROCEDURE initialize ;
VAR junk : Integer ;
    trash : Str255 ;

BEGIN
    ship_name[1] := 'Destroyer' ;
    ship_name[2] := 'Submarine' ;
    ship_name[3] := 'Cruiser' ;
    ship_name[4] := 'Battleship' ;
    ship_name[5] := 'Aircraft Carrier' ;
    title := ' Battle Blips! ' ;

    msg_box := new_dialog( 0, 20,19,40,5 ) ;
    msg_idx[0] := add_ditem(msg_box, g_text, none, 1, 4, 38, 1, 0, 256 ) ;
    FOR junk := 1 TO 3 DO
        msg_idx[junk] := add_ditem(msg_box, g_boxtext, none, 1, junk, 38, 1, 0, 256 ) ;

    trash := Concat( chr(4), ' ', chr(3), ' ', chr(1), ' ', chr(2) ) ;
    trash := Concat( trash, ' [RETURN] [UNDO]' ) ;
    set_dtext( msg_box, msg_idx[0], trash, 5, TE_CENTER ) ;
    cont_game := FALSE ;
END ; { initialize }

(*-----*)
(* To create a desk accessory, modify the following BEGIN-END *)
(* block of code so that it will compile and modify the next *)
(* BEGIN-END block of code so that it won't compile. *)
(* Make sure the first program line: *)
(* ($A+,D-,S10 ) *)
(* is NOT commented out. *)
(*-----*)

(**)
BEGIN ( Battle_Blips ( accessory ) )
    ap_id := Init_Gem ;
    IF ap_id >= 0 THEN BEGIN
        dummy := Menu_Register ( ap_id, title ) ;
        initialize ;
        WHILE TRUE DO BEGIN
            dummy := get_event(E_Message, 0, 0, 0, 0,
                FALSE, 0, 0, 0, 0, FALSE, 0, 0, 0, 0,
                msg, dummy, dummy, dummy, dummy, dummy ) ;
            IF msg[0] = AC_OPEN THEN
                play_game ;
        END ; { while }
    END ;
END ;

(*-----*)
(* To create a standalone application, modify the following BEGIN-END *)
(* block of code so that it will compile and modify the previous *)
(* BEGIN-END block of code so that it won't compile. *)
(* Make sure the first program line: *)
(* ($A+,D-,S10 ) *)
(* IS commented out. *)
(*-----*)

(**)
BEGIN ( Battle_Blips ( program ) )
    ap_id := Init_Gem ;
    IF ap_id >= 0 THEN BEGIN
        initialize ;
        play_game ;

        exit_gem ;
    END ;
END. ( Battle_Blips )

```

Errors in Abacus

by Charles F. Johnson

One of the hardest things about programming the ST is just figuring out what everything means. You come across cryptic terms like "message pipes," "object trees," "memory form definition blocks"... yikes! What are these things? Who can you turn to for help? This is not my beautiful house! This is not my beautiful wife! (Apologies to the Talking Heads.)

In this article, I'm going to take a close look at a couple of the reference books available to the would-be ST programmer who doesn't want to cough up \$300 for the Atari Developer's Kit. I've used just about all of the popular ones myself (including the Developer's Kit), and unfortunately every one of them contains numerous mistakes. I'll provide a listing of some of the errors I've encountered in my own programming exploits, and hopefully spare one or two of you the woes I've had to go through. (Sob! Boo-hoo!... SLAP! Thanks—I needed that.)

By now, everyone knows that Abacus has cornered the market on reference books for the ST. At last count, they had no less than 12 titles, covering everything from beginner's Logo and ST Basic to 3D

graphics in assembly language. Most of the Abacus books do a very decent job of covering their subjects and are fairly well organized, although some lack indexes and other niceties. One has to wonder, though, if Abacus employs a proofreader at all... many of their books are inexcusably full of typos, misspellings, and just plain bad English. Of course, some of them are translated from German, which probably accounts for the contorted phrasing; but at times, reading an Abacus book is somewhat akin to reading Lewis Carroll's *Through the Looking Glass*.

For the programmer, two of these books are essential: *ST Internals* and the *GEM Programmer's Reference*. These two books contain most (certainly not all) of the information you'll need to program the ST; in fact, if you don't want to use GEM (i.e. no dialog boxes, drop-down menus, or any of that stuff), *ST Internals* will do quite nicely by itself. Actually, I have to admit I'm surprised that Abacus got permission to print the section on the Line A graphics primitives *at all*—the Line A document in the Atari Developer's Kit is marked "Confidential!" on each and every page!

Unfortunately, as valuable as *ST Internals* is, the typo monster ran amok in this book. There are so many mistakes that I'm not even going to try to list them all. Watch out for the function numbers they use in the examples for the GEMDOS calls; sometimes they're wrong, even though the descriptions use the right numbers. Another common error in *Internals* shows up in a lot of the GEMDOS examples in assembly language. GEMDOS is called from assembly language by pushing parameters on the stack, doing a TRAP #1 (the 68000's TRAP instruction causes a branch through a vector contained in low memory; there are 16 possible TRAP vectors), and then correcting the stack pointer. In many of the examples in *Internals*, they add the wrong value to the stack pointer to correct it. This can lead to big trouble, and it can be very hard to find this kind of error in a large program; so if you use the examples in *Internals* as guides for your own routines be sure to double-check your stack corrections.

Since this article was first written, Abacus has released two subsequent editions of *ST Internals*. The second edition fixed some but not all of the most glaring errors. The recently-released "Third Revision" is a substantial improvement over its predecessors, although many errors and typos still remain. This edition con-

tains quite a bit of new material, including information about GEMDOS memory management and the blitter chip, and a commented disassembly of the first part of the ST's ROM. Many more assembly language examples are given as well. If you're considering buying *ST Internals*, by all means try to make sure you get the third revision. If, however, you already own a copy of the first or second edition, you may find the error list provided below invaluable.

Some of the worst mistakes in the first and second editions of *ST Internals*:

Pg. 105—The second line of the example for the TERM function should read "TRAP #1." They left out the "#1."

Pg. 110—The description of the SETDRV function (# \$0E) is misleading. It reads, "The current drive can be determined with the function SETDRV." It should say, "The current drive can be *set*..." Also, the last sentence in the description is wrong. It should say, "After the call, D0 contains a bit map of the active drives."

Pg. 113—In the example for the SUPER call (# \$20) the fourth instruction is given as "ADD.L \$6,SP." This should be "ADD.L #6,SP."

Pg. 114—The wrong value is added to the stack in the example for SET DATE, function number \$2B. That "ADDQ.L #6,SP" should be "ADDQ.L #4,SP."

Pg. 115—The same mistake is made in the example for the SET TIME function.

Pg. 120—In the third paragraph of the description of the CREATE function, the second sentence contains a typo. It should read, "Attribute \$04 creates a 'hidden' file and attribute \$02 a 'hidden system' file."

Pg. 122—In the example for the OPEN function, the second line of code should read, "MOVE.L #filename,—(SP)."

Pg. 124—The wrong function number is used in the example for READ—function number \$3F. The example incorrectly uses \$3E.

Pg. 125—In the example for UNLINK (function \$41) there's a "#" sign missing from in front of the label "pathname." The first line of code in the example should read "MOVE.L #pathname,—(SP)."

Pg. 130—In the second example for the MALLOC function (# \$48), the fifth line of code is wrong. It says, "TST.W D0." It should read, "TST.L D0."

Pg. 133—Some important information is missing from the description of the EXEC call (# \$4B). The mode word can be 0, 3, 4, or 5, not simply 0 or 3 as Abacus states. Here are the functions of each mode:

0 = Load a program and immediately start.

3 = Load a program without executing. The address of the program's basepage (or an error message) is returned in D0.

4 = Execute program loaded with mode 3. Pass zeroes for the environment and filename, and pass the address of the basepage (returned from mode 3) as the command line.

5 = Create a basepage.

Also on page 133, the wrong value is added to the stack in the example. "ADD.L #14,SP" should be "ADD.L #16,SP."

Pg. 135—In the paragraph dealing with the 44-byte DTA buffer, bytes 26 through 29 are described as the "File size in bytes (low byte, high byte)." This is incorrect; the file size is in standard Motorola format, high byte to low byte. Also on page 135, the example given for the SFIRST call is very confusing. Here's how it should appear:

(SEE TABLE 2)

Pg. 160—The top line of this page should read:

3 logbase get logical screen base

Pg. 162—Another wrong value added to the stack! This time it's in the example for the XBIOS **setscreen** function (# 5). In both examples, the "ADD.L #10,SP" should be "ADD.L #12,SP".

Pg. 165—In the description of the XBIOS **floprd** function the possible sector numbers are listed as 0-9. They should be 1-9. (Do you think the person who wrote that ever actually used this function?)

Pg. 176—The description of the XBIOS **keytbl** function is rather garbled. To get the address of a table containing the three keyboard table pointers, pass -1 for all three parameters.

Pgs. 192,193—The descriptions of the **offgibit** (- 29) and **ongibit** (- 30) functions are reversed. Also, the descriptions say to pass the number of the bit you want to change—wrong; you pass a word value that will be ANDed or ORed with the register.

Pg. 203—In the assembly examples for the **wvbl** function (wait for vertical blank), the wrong function number is used. It should be 37, not 36.

Pg. 223—The description of the Line A TRANSFORM MOUSE opcode is missing some important data. The book says 34 words are necessary to define the new mouse form; actually, 37 words are necessary. Add these three lines to the beginning of their list of parameters:

INTIN(0) Ilot spot X position. (0-15).

INTIN(1) Ilot spot Y position. (0-15).

INTIN(2) Number of color planes.

Pg. 224—In the description of the Line A DRAW SPRITE opcode, word 3 (the format flag) is given incorrectly. The book says "0 = VDI format, 1 = XOR format"; it should be "0 or 1 = VDI format, -1 = XOR format." Actually, I think any positive number denotes VDI, and any negative number denotes XOR.

Pg. 249—Location \$44C is described as a word value (two bytes). This location contains the current screen resolution (0 = low, 1 = medium, 2 = high) and is a byte value, not a word.

Pg. 251—At the top of the page, the system variable **swv_vec** at location \$46E is described incorrectly. The book says, "A branch is made via this vector with the screen resolution is changed." That "with" should be "when." Also, the book doesn't say that this vector is only used if you change from a monochrome to a color monitor, or vice versa. The vector is *not* used when you switch between medium and low resolutions.

Pg. 251—The default value for location \$484 is incorrect. It should be 7 not 6.

Pg. 253—The default value for location \$4BA (the 200Hz timer) is given as \$71280. This isn't strictly wrong; but it is kind of ridiculous, since location \$4BA is incremented every 200th of a second. The Abacus "default value" is just what happened to be there at the instant they looked.

This error listing probably doesn't cover all of the mistakes in *ST Internals*; but it should give you an idea of the kinds of things to watch out for if you're using this book.

Another problem with the Abacus *ST Internals* book is that the names given for the GEMDOS function calls are all different from the names listed in the bindings for both Alcyon and Megamax C. This isn't really an error; after all, it doesn't matter what names you use for the functions. However, if you're programming in C it can be hard to find the proper descriptions since Abacus and your C manual will disagree on the function names. The chart in Figure 1 lists each GEMDOS function with both its "Abacus" and its "legal" name ("legal" meaning the name given to it in the C bindings).

The *GEM Programmer's Reference* contains fewer errors than *ST Internals*. The worst error (one that caused me tons of frustration during a recent programming project) is lurking in their examples for application initialization in assembly language (not in the C example). The initialization examples are found in three places in the Programmer's Reference; pages 50-53, 249-254, and 271-272. In all three

examples, they fail to save and use the proper value for the VDI handle. GEM initialization is achieved by several steps: **appl_init**, followed by **graf_handle**, followed by **v_opnvwk**. The **graf_handle** call returns a value that is passed to the **v_opnvwk** call; then **v_opnvwk** returns a value of its own, which should

8K x 8K RES AVAILABLE

SLIDES!

SEND US YOUR IMAGE ON DISK
GET BACK QUALITY 35MM
SLIDES - NO SCAN LINES-

SLIDES 2K RES -- \$6.75
(MINIMUM ORDER \$25.00)

DIGITAL COLOR SEPARATION
AVAILABLE

 **ImageSet**

415-626-8366 corp.
Computer Image Processing

555 19TH STREET
SAN FRANCISCO, CA 94107



CIRCLE #110 ON
READER SERVICE CARD.

ALICE

The Personal Pascal

An integrated programming environment with 700 HELP screens, an editor that makes errors impossible, and the best GEM interface anywhere. **Only \$79.95.**

"An excellent value." - Antic

"It is about as painless a method of learning Pascal as can be devised short of hypnosis. It works!" - Computer Shopper

"The product is all anyone could ask for. I would recommend this product to anyone who is considering learning PASCAL . . . or anyone who wishes to prototype small applications which deal closely with GEM." - ST Informer

Orders: 1-800-265-2732

Looking Glass Software
looking glass software

124 King St. N. Waterloo, ON. N2J 2X8 519/884-7473

CIRCLE #111 ON
READER SERVICE CARD.

GEMDOS Function Names.

<u>Abacus</u>	<u>C</u>	<u>No.</u>	<u>Abacus</u>	<u>C</u>	<u>No.</u>
TERM	Pterm0	\$00	KEEP PROCESS	Ptermres	\$31
CONIN	Cconin	\$01	DISK FREE SPACE	Dfree	\$36
CONOUT	Cconout	\$02	MKDIR	Dcreate	\$39
AUX INPUT	Cauxin	\$03	RMDIR	Ddelete	\$3A
AUX OUTPUT	Cauxout	\$04	CHDIR	Dsetpath	\$3B
PRINTER OUTPUT	Cprnout	\$05	CREATE	Fcreate	\$3C
RAWCONIO	Crawio	\$06	OPEN	Fopen	\$3D
DIR CONIN W/O ECHO	Crawcin	\$07	CLOSE	Fclose	\$3E
CONIN W/O ECHO	Cnecin	\$08	READ	Fread	\$3F
PRINT LINE	Cconws	\$09	WRITE	Fwrite	\$40
READLINE	Cconrs	\$0A	UNLINK	Fdelete	\$41
CONSTAT	Cconis	\$0B	LSEEK	Fseek	\$42
SETDRV	Dsetdrv	\$0E	CHANGE MODE	Fattrib	\$43
CONOUT STAT	Cconos	\$10	DUP	Fdup	\$45
PRTOUT STAT	Cprnos	\$11	FORCE	Fforce	\$46
AUXIN STAT	Cauxis	\$12	GETDIR	Dgetpath	\$47
AUXOUT STAT	Cauxos	\$13	MALLOC	Malloc	\$48
CURRENT DISK	Dgetdrv	\$19	MFREE	Mfree	\$49
SET DTA ADDRESS	Fsetdta	\$1A	SETBLOCK	Mshrink	\$4A
SUPER	Super	\$20	EXEC	Pexec	\$4B
GET DATE	Tgetdate	\$2A	TERM	Pterm	\$4C
SET DATE	Tsetdate	\$2B	SFIRST	Fsfirst	\$4E
GET TIME	Tgettime	\$2C	SNEXT	Fsnext	\$4F
SET TIME	Tsettime	\$2D	RENAME	Frename	\$56
GET DTA	Fgetdta	\$2F	GSDTOF	Fdatetime	\$57
GET VERSION NUMBER	Sversion	\$30			

FIGURE 1

be used as the "handle" number for all VDI function calls. In the Programmer's Reference, they use the value returned by **graf_handle** as the VDI handle. . . .this is *wrong, wrong, wrong!* After the **v_opnvwk** call, the *real* VDI handle is returned in the application's control array, in **control[6]**. Using the value from **graf_handle** can lead to all kinds of impossible-to-track-down bugs. In the case of my program, it ran fine without a single problem but caused *mucho* havoc with other programs after I exited. After running my program, some other programs would come up without a mouse cursor, and if I called accessories from the desktop they would not be erased from the screen when I closed them.

I suspect that a lot of the programs that have a reputation for causing trouble with other programs are using the wrong VDI handle value, and hopelessly confusing the poor ST! If you're programming in assembly language, always be sure to save the value that the VDI returns to you in **control[6]** after the **v_opnvwk** call, and pass it as

the handle when you call any other VDI functions. A note of interest—in the C examples in the Abacus books, the VDI handle is always obtained correctly. In the C statement:

v_opnvwk (work_in, &handle, work_out);

the ampersand symbol in front of the variable "handle" indicates that a value is returned in that variable. The C bindings for the **v_opnvwk** call take care of getting the proper VDI handle from your program's control array for you.

On page 199 of the Programmer's Reference, there is an error in the description of the VDI **vq_mouse** call. After the call, **intout[0]** will contain the status of the mouse buttons (not "button," as Abacus says). Bit 0 indicates the left button status, bit 1 the right; if the bit is on, that button is being pressed. Therefore, the possible button values returned from **vq_mouse** are:

- 0 = No buttons pressed
- 1 = Left button
- 2 = Right button
- 3 = Both buttons

When you click on a window to activate it ("top" it) or when you call an accessory from the 'Desk' drop-down menu, GEM lets the chosen application know of your desires by sending a "message" to it. The programmer sets aside a small buffer that serves as the "message pipe," the place where GEM will write these messages as they occur. (See, message pipes aren't all that mysterious, are they?) The first word (two bytes) in the message pipe is always the message identifier, and each message has a unique identifier. In the *GEM Programmer's Reference*, the message events are described on pages 308 through 311—and here we find some more errors.

Pg. 309—the **WM_FULLED** message is missing its description. This message indicates that the user has clicked on a window's "full box" (in the upper right-hand corner) to enlarge the window to full screen size. When a **WM_FULLED** message occurs, word 0 of the message pipe will contain 23, and word 3 will contain the window handle of the chosen window.

Pg. 311—several mistakes rear their ugly heads. The information for the WM_NEWTOP, AC_OPEN, and AC_CLOSE messages is all wrong. Here's how they should appear:

WM_NEWTOP:

The application is informed that a window was activated.

Word 0 = 29 (Abacus says 30...)
Word 3 = Window handle (They got this one right!)

AC_OPEN:

The user selected one of the six desk accessories.

Word 0 = 40 (Abacus says 30...again.)
Word 4 = The menu identification number, which can be read through the function MENU_REGISTER. (Abacus says this value is in word 3.)

AC_CLOSE:

This event occurs under the following conditions:

- The screen is cleared,
- The window data has changed,
- The running application was

interrupted.

Word 0 = 41 (Abacus says 31...)
Word 4 = Menu ID number (see also AC_OPEN)

Sadly, the wrong info for the AC_OPEN and AC_CLOSE messages is repeated throughout every source of documentation for the ST (including the Developer's Kit!). This little tidbit caused me a few days of grief while I was developing my Font Tricks accessory (from ST-LOG #10), since when an accessory isn't active it just sits and waits for the AC_OPEN message. If you follow the docs, you'll be watching for a message that never comes. Font Tricks was written in assembly language; once again, if you program in C and use the standard files for equates, you'll never run into this problem...

because the C bindings have these messages listed correctly! In fact, that's how I eventually found out the truth about the message numbers—by looking through the C equate files in the Developer's Kit.

On page 322 in the description of the AES **appl_write** call, the parameters for the control array are given incorrectly. The input parameters should be:

control(0) = 12
control(1) = 2
control(2) = 1
control(3) = 1
control(4) = 0

There are quite a lot of AES functions missing from the Programmer's Reference. For example, there is no mention at all of the **fsel_input** call...odd, because this is one of the most handy and easy-to-use functions in the AES Libraries. This is the call which puts the 'Item Selector' dialog box on the screen, allowing you to select files from disk for saving, loading, etc.

To close this tale of heartache and woe, here's a description of the syntax for the **fsel_input** call (just for the heck of it, in the Abacus style), for those of you without any docs on it:

FSEL_INPUT: Opcode = 90

FUNCTION

Display the GEM File Selector Dialog Box.

Input:

control(0) = 90
control(1) = 0
control(2) = 2
control(3) = 2
control(4) = 0

addrin(0) = dir_spec
addrin(1) = file_sel

Output:

intout(0) = ret_code
intout(1) = exit_but

dir_spec: Pointer to a string that describes the original directory specification. On exit, this points at the directory selected by the user.

file_sel: Pointer to a string that contains the filename selected by the user.

ret_code: If zero, an error occurred. If greater than zero, everything's fine.

exit_but: Indicates which exit button was selected. 0 = Cancel, 1 = OK.

Corrected SFIRST call

```
MOVE.L #dta,-(SP)    *Set up DTA buffer
MOVE.W #$1A,-(SP)    *Function number SETDTA
TRAP #1
ADDQ.L #6,SP
MOVE.W attrib,-(SP)  *Attribute value
MOVE.L #filnam,-(SP) *Name of file to search for
MOVE.W #$4E,-(SP)    *Function number
TRAP #1
ADDQ.L #8,SP
TST D0               * File found?
BNE notfound         *Apparently not
.
.
.
attrib:
        .dc.w 0        *Search for normal files only
filnam:
        .dc.b '*,*',0  *Search for 1st possible file
.
.
.
dta:
        .ds.b 44       *Space for the DTA buffer
```

Charles F. Johnson is a professional musician, and now semi-professional computer programmer/reviewer/author. He lives in Los Angeles with his wife Patty and Spike, the world's most intelligent (and gluttonous) cat. Charles is a sysop on the Analog and ST-LOG Atari SIG on Delphi; his user name is CFJ.

DATABASE DELPHI

by Andy Eddy

If you take a peek out your window, you'll see that summer is heating up. Likewise, things are heating up on Delphi. By now, the much requested separation of the Atari camps into two SIGs should have taken place. At the time of this writing (early April), the work was nearly completed to move ST files into that area and get the road paved for the smooth transition. The housekeeping includes such chores as selecting topic names for Forum and Database entries, and getting the shopping area "built" for those individuals and companies interested in selling their wares on Delphi. This is one of the less discussed aspects of the network, and we'll touch on it in a future Database Delphi installment.

Also up in the air at the time of this writing was the actual name of the SIG. The plan was to keep the title (as it will appear on the Groups menu) unique, so

users can type just a couple of letters to enter the desired SIG (such as typing AT to get to the ANALOG/Atari SIG previously). Unfortunately, the name couldn't begin with the obvious choice of "ST," or there would be a conflict with the Starship Commodore SIG; similarly, using "AT" for Atari or "AN" for ANALOG would conflict with the existing 8-bit SIG. There's nothing more annoying than getting "The term 'ST' defines more than one command," Delphi's reply to menu collisions like the ones just noted would cause.

By the time you sit in your sun-drenched lawn chair to read this—if you're not already prone, please do so now—the issue will have been resolved and you'll have found your way to our new home. Just the same, we'll note it in the next Database Delphi. No matter what the outcome, the staff overseeing the data flow will continue to be the

same cheery lot as before.

A Glance Askance

The forum has also been warming to a variety of subjects lately—some good, some potentially bad. Talk of WordPerfect dropping ST support (which was later vehemently denied by the company), mention of a new product called Turbo-ST (an interesting piece of software that claims to act like a text blitter, speeding moves of characters on-screen) and the potential of bringing up a matching desktop configuration when changing resolutions, were among the loftier subjects of discussion. The latter was bandied back and forth between some of the programming gurus online, namely Lloyd Pulley (MADMODIFIER, author of the public domain, multipurpose boot program called Megamatic), Darek Mihocka (DAREKM, author of the ST Xformer,

8-bit emulator) and our own Charles F. Johnson (CFJ). Apparently there is a solution to this dilemma, but you'll have to wait to see it for yourself, as Charles is planning on gracing *ST-Log* readers with his fix. For now, mum's the word.

As far as the other topics I mentioned, they are more related than they appear. If I may stand on the soapbox for a second, most everyone is sufficiently concerned by the stigma that piracy has placed on the ST market. The rumor spread that WordPerfect would drop the ST as a market due to the discovery of some BBSs carrying the word processor in their libraries; but of course WordPerfect isn't alone in their concern. Many other developers, big and small alike, are getting scared off.

If you look at the other side of the coin, smaller scale developers like Wayne Buckholdt of Softrek, the makers of Turbo-ST, show the exciting, barely tapped potential of the ST. Only if enough people buy the products—as opposed to stealing them—will our favorite computer continue to flourish. Furthermore, if piracy continues unabated, it lessens the ability for the manufacturers to support the brand lucratively.

The bottom line—and we've said it before—is to discourage piracy in all forms. It's illegal to possess a program that you haven't paid for, yet it's very difficult for anyone to do anything about it. It requires a certain amount of consumer responsibility. If you own pirate software, either destroy it or pay for it; if you know of someone dealing pirated products, tell them you don't appreciate what they are doing; if you know of a pirate BBS, contact the Software Publishers Association (Suite 1200, 1111 19th Street, N.W., Washington, D.C. 20036) and/or the manufacturers whose products appear on the board. Software for Atari computers has always been reasonably priced, so there's no reason to take it. You'll do us *all* a favor. End of sermon. Please be seated.

Ready, Set . . . SET!

In the brief space we have left, we can look over how to best wangle our way through the materials we have at hand on Delphi: SETTINGS.

At the ANALOG > prompt (or any SIG prompt for that matter), type SET. It works like this:

ANALOG > What do you want to do?
set Preferences Menu:

Name Change
Editor Preference
Topic Selection
Settings (Profile)
HELP
Exit

All the stuff here is reasonably self-explanatory, though we'll go in depth on the editors at a later date, as there is too much to cover in a short glimpse.

The power selection here is Settings, and if you enter SET again, this time from the PREFERENCES > prompt, you'll get the following:

PREFERENCES >
(Name,Edit,Top,Set,Help,Exit) set
Some of your temporary settings are being restored to their initialization values.

SETTINGS > ?

SETTINGS Menu:

BUSY-Mode	PROMPT-Mode
DEFAULT-Menu	SET-IIhigh-bit
DOWNLOAD-Line-terminators	SLASH-Term-settings
ECHO-Mode	TERMINAL-Type
EDITOR	TIMEOUT
FILE-TRANSFERS	UTILITIES
KERMIT-SETTINGS	WIDTH (Columns)
LENGTH (Lines/page)	XMODEM-SETTINGS
NETWORK-PARAMETERS	HELP
PASSWORD (Change)	EXIT

Again, the entries are descriptive enough in most cases to get you by. Simply put, you can tailor your characteristics to whatever you desire. Would you like to do file transfers in Xmodem? Go to the XMODEM-SETTING menu. Would you like to limit the time that you can stay idle and remain online? Simply go to the TIMEOUT area and set a value. That way, if you are called away in a hurry and don't log off, Delphi will automatically terminate the session in that amount of time. As a warning that the time-out figure is coming nigh, you'll get the message, "Please respond within 30 seconds to avoid being automatically logged off," just in case you were asleep at the wheel and still want to stick around.

Perhaps the most powerful feature from the Settings area is the DEFAULT-menu selection. Here you can enter what you want Delphi to do for you

when you first log on. For example, my Default menu is GRO AT WHO (short for GROUPS ATARI WHO). When the initial log-in procedure is completed by Delphi, it reads the GRO AT WHO as if it was typed in. Saving me the trouble, it runs me past the GROUPS AND CLUBS area, to the ANALOG/Atari SIG and then gives me a listing of who is online in the ANALOG Atari SIG at the time.

It's important to note that some of these selections are not for the layman telecommunicator. And keep in mind that if you frequent different SIGs, you'll need to set up some SETTINGS from area to area. Just the same, with a bit of playing, you can find out what most of the other functions do and can best put them to your benefit.

Book Him, Mike . . .

I've also just picked up a copy of Michael Banks' (Database Delphi columnist for *ANALOG Computing*) book, *Delphi: The Official Guide*, a Brady Book, which is published by Prentice Hall Press. I realize that informing you of this may ultimately put me out of a job, due to the fact that this book covers everything I might want to talk about here. A 500-page handbook, *Delphi: The Official Guide*, is a great reference tome that details all you'd ever want to know about Delphi. It's only \$19.95, and will easily save you that much in online costs. If you want to save some time in picking up a copy, you can order online by typing GO GUIDE.

In association with Delphi, *ANALOG* is able to offer a sign-up that can put the guide in your hands *and* get you a regular account on Delphi at the same time, provided you aren't already a member. This reduced price offer—available after May 1st—not only gives you a lifetime membership to Delphi (a \$19.95 value), but also a copy of *Delphi: The Official Guide* and a free hour of online time at non-prime rates. If you log into Delphi employing username JOINDELPHI and password ANALOG, you'll be brought up to this signup offer.

Finally, don't forget that we meet every Tuesday night in the Conference area, at 10 p.m. Eastern time. This will continue to be the date after the new ST SIG has started up. Turnout has been increasing and conversations cover all topics, so stop by and chat with us.

Till next month, C U online . . .

MORE SUPERIOR PRODUCTS FROM NAVARONE

QUALITY TOOLS FOR YOUR ST SYSTEM

ST VIDEO DIGITIZER



\$ 79^{.95}

Digitize from any standard composite video source (e.g. VCR, video camera, etc.). Save digitized pictures into NEO or DEGAS™ file formats. This is the fastest digitizer available for the ST. Capture single frames in less than a second. Excellent for student, hobbyist, or to put pictures in your desktop publishing projects. The picture above was taken with the ST Video Digitizer and printed directly on a laser printer.

ST SOUND DIGITIZER

\$ 99^{.95}

Digitize real-world sounds from microphone, record player, tape recorder, guitar, etc. Play back through your amplifier or MIDI keyboard. The ST Sound Digitizer can be used to create music, experiment with sounds, edit short commercials, or use for voice mail. Very easy to use software provides powerful editing and mixing features.

TIMEKEEPER

\$ 29^{.95}

This is our popular clock calendar plug-in cartridge. The Timekeeper comes complete with removable long life lithium battery ready to use. Just plug it into the cartridge slot and set up either an Auto folder or Accessory program to automatically set Time and Date each time you turn on your ST.

To Order: Call our toll free number or send M.O. plus shipping (call for rates). VISA, MC, C.O.D. welcome. California residents add 7% sales tax.

NAVARONE



1-800-624-6545 (Nationwide)
Or **(408) 378-8177** (California)

NAVARONE INDUSTRIES, INC. • 454 Kenneth Avenue • Campbell, CA 95008

Prices and availability are subject to change without prior notice. DEGAS is a registered trademark of Batteries Included, Inc.

CIRCLE #112 ON READER SERVICE CARD.

TUTORIAL

Step 1-Hard Facts

More Baudy Tales

Using a modem.
by Maurice Molyneaux

Back in the April '87 issue of *ST-Log* appeared my article "A Baudy Tale," which was a beginner's guide to telecommunications. In that article I discussed what a modem is, various aspects of how they work, terminology, etc. If you haven't read that article before, it might be a good idea to read it now (if you have the issue). In this installment of **Step 1** we're going to discuss not the subject of modem hardware, etc., but of *using* a modem and terminal software.

Before going on, I would like to thank Supra Corp. for providing me with their Supra 2400 baud modem for use in the writing of this article. Four different modems were used to research this article: the Hayes Smartmodem 1200, Atari

SX212, Avatex 1200hc, and the aforementioned Supra 2400.

What it is

If you're new to computers, you might not know exactly what a modem is. You are probably aware that they are used with computers and telephone lines. If you've read the papers, you've probably heard about those "hackers" who use modems to break into hospital, business and government computers, wreaking havoc and wrecking files. You may picture the guys who alter phone bills, and wreck the financial accounts of people they don't like by passing around their credit card numbers. You may even picture the movie *Wargames*, with thoughts of people causing or nearly causing a nuclear war or other disaster.

But, as is so often the case with the news and media, you only hear reports on the *misuses* of such technology. It's true, people have caused havoc with computer and modems, but they are the minority. Just as a handful of homicidal maniacs are not indicative of the rest of the population of this country, people who use modems for mischief are not indicative of the majority of us who use modems for the purposes they were designed for.

There are a lot of exaggerations about modems and the people who use them. These "tall tales" are based upon the misdeeds of a few unscrupulous people, true, but they do not tell the whole story. Breaking into other people's systems and messing with their files is not the reason modems were created. They were made to allow computers to "talk" to each other over conventional telephone lines, exchanging data over great distances quickly and easily.

"Modem" is an acronym for MODulator DEModulator, which is a fancy way of saying the sending modem MODulates its data into a carrier signal that can be sent over a phone line, and the modem on the other end DEModulates the signal, converting it back to its original form (binary numbers).

People use modems for all kinds of things, but the heart of it is always data transmission, whether that data be the latest stock market report, digitized pictures, or just sending weird stories to friends, it all boils down to sending that data (or, should I say, *copies* of it) from place to place.

Choosing a modem

Modem's for the ST are all (thus far) external, meaning they are a separate box

that connects to your computer rather than an internal card. Generally speaking, most any "generic" modem (in other words, one not designed for a specific system, like the XM301 for the Atari 8-bits) can be used on the ST. You needn't look for a modem to have "works with Atari ST" on its box. Any "generic" modem featuring an RS232 interface should work (be warned, however, there are always exceptions).

The first thing you should look for in a modem for your ST is a high transmission speed. The minimum speed modem you should consider buying is one capable of running at 1200 bps/ baud (bps = bits per second, often used interchangeably with "baud"—see "A Baudy Tale" in the April '87 *ANALOG Computing* for more on bps/ baud.) Forget any modem slower than that. Most ST files and programs are too large to be transferred at 300 bps... unless you don't mind your computer being in transfer mode for an hour or more!

2400 bps is becoming more and more common with ST owners, and with good reason. It is *fast!* Most people can comfortably read text being transmitted at 300 bps, and can skim text at 1200 bps. At 2400 bps you can only fast-scan it (unless you are a student of Evelyn Wood). At 2400 bps, I was able to scan 100 Forum messages in the Delphi ANALOG Atari SIG in just two minutes! Had I read them all I would have been there much longer. I merely made sure all the incoming text

went into my terminal program's capture buffer, and read the messages at my leisure after I hung up. Transferring files at 2400 bps is also much less painful.

As I stressed in "A Baudy Tale" it's a good idea to have a modem which features something of the Hayes modem command set (created for Hayes Microcomputer Products), or at least the Bell 212 standard (something of a subset of the Hayes command set). All four modems I used featured this capability. (Note: the Avatex modem I was using was the 1200hc, not the 1200. The hc model has a fuller Hayes command set than the 1200 model, and is therefore preferable—although both will work. *Do not* purchase a 1200i, as it's an *internal* modem and *not* for the ST!) The reason for wanting such command set compatibility is that the Hayes and Bell 212 command sets are something of a standard in the industry, and most modem software will work with them easily.

Now, let me quickly describe the modems I used (excuse any omissions, but I can't possibly discuss every modem usable with the ST). Unless otherwise noted, the modems all have approximately the same "footprint" as an SF354 or SF314 disk drive, and can therefore be "stacked" under or over them. All the modems feature status lights on their front panels, external power adapters, and female RS232 ports.

Hayes Smartmodem 1200

As a Hayes it features full Hayes compatibility, and it's well built, encased in a tough metal shell. You must plug a phone cord into the modem, but you cannot carry the line "thru" the modem to a phone. If you also need a phone on the same outlet, you'll need a splitter box. The modem can operate at bps rates from under 100 up to 1200. Settings can be altered with Hayes commands, and default settings changed by setting DIP switches located (inconveniently) behind the front panel. The power switch is located around the back of the unit.

Avatex 1200hc

Features pretty thorough Hayes command set compatibility. The 1200hc has a phone cable installed, and a plug for hooking your telephone "thru" the modem. The only problem with the permanently attached cable is that you couldn't quickly replace it if it got damaged. Max speed is 1200 bps. The modem's default settings can be set using DIP switches on the back of the unit. They can also be set using the Hayes commands. The unit

is encased in sturdy plastic, and features power, voice/data and 300/1200 bps buttons on its front panel... quite handy. This modem is covered by a two-year warranty. Very nice.

Atari SX212

With good Hayes/Bell 212 compatibility, the SX212 is a smart-looking unit, encased in "Atari gray" plastic and featuring diagonal status lights on the front panel. It can transfer data at rates of up to 1200 bps, and features no user alterable hardware switches. However, it can be programmed. The power switch is on the back of the unit, annoyingly placed between the power cord plug and the modem cable. In addition to the RS232 port, there is also an Atari SIO port used for hooking the SX212 to an Atari 8-bitter. There is no phone "thru" port.

Supra Modem 2400

Also with good Hayes compatibility, the Supra Modem 2400 features a Hayes-esque metal case, and is decidedly smaller than the other modems, which requires it to be at the top of a peripheral "stack." The front and back panels are closed with black plastic, that seems a tad bit soft. The power switch is on the front of the unit, and the back features a connector for phone line and a "thru" for a telephone. The modem features no DIP switches, but can be programmed, and, unlike the others, can remember your programmed settings even after you turn it off! This because it contains a static RAM chip that holds your own preferred settings (you must save them, of course). Top transmission speed is 2400 bps.

Intro to the Hayes command set

The power of the Hayes command set is that, if your modem supports it, you can program the modem itself. For example, most modems wait about 15 to 20 seconds after dialing a number to get a connection. If they don't get a connection, they hang up. Suppose the number you were dialing takes longer than this to respond. With many cheap modems there's nothing you can do, but with a Hayes compatible modem you can tell the modem how long to wait for a carrier detect (I'll explain this term later) before giving up. This is set using one of the modem's S registers, which are programmable "switches."

S register number 7 holds the value for the "Time to Wait for Ring-back/Carrier." Default is usually 20 or 30 units (units usually equal seconds). You might want

ST-U.S.E.

THE USED PROGRAM EXCHANGE FOR YOUR ST

Trade Your Old Programs for Exciting New Titles

Buy Quality ST Programs at a Fraction of the Original price.

Over 175 Titles currently in stock
New Arrivals Daily

NOW SELLING NEW SOFTWARE

Call or write today for a free price list and membership info.

CIRCLE #120 ON READER SERVICE CARD

ST-USE

P.O. Box 868

Great Barrington, MA 01230

(413) 528-4728 9 a.m. - 5 p.m. Eastern Time
MasterCharge and VISA Accepted

CIRCLE #113 ON
READER SERVICE CARD.

to make the modem wait 60 seconds before giving up. To do this you'll need to put the value 60 into the S7 register.

To let the modem know that you are sending a modem command, you need to use the "attention" command, which is "AT" (sans quotes). Some modems require all such commands to be entered in uppercase, else they will ignore them. If you have a modem, turn it on, run your telecommunications software, or even just use the Emulator desk accessory. When you are on the terminal (as in "communications terminal") screen, type AT and press RETURN. The modem should report back OK to this. If not, make sure the modem is on and connected. Also, if you typed lowercase characters, try upper case. Now, we need to use the attention command and tell the modem to put the value 60 into register S7. This is done with the command `ATS7=60 [RETURN]`. The modem will report back OK if it worked.

To see what is currently in a modem register, you type the attention command, the S register value, and a question mark. To see what the contents of register S7 are you would type `ATS7? [RETURN]`, and the modem will report back 60 if you followed the previous step.

Attention commands also precede dialing instructions. The command to dial the phone is ATD (attention—dial.) You would follow this with a phone number like `ATD 555-1212 [RETURN]`. This tells the modem to dial the specified number. Furthermore, you can tell the modem what sort of dialing to use: pulse (rotary dial) or touch tone. If you have a pulse line, you would add a P to the command, or a T if you have touch-tone dialing, as in `ATDP 555-1212` or `ATDT 555-1212`.

As much as the Hayes command set is a standard, there are still variations. For example, using the command `ATAA` on the Avatex 1200hc sets the modem to auto-answer state, where it will answer the phone when it rings. Using the same command on the other modems won't work this way, as they see it as an ATA command, causing the modem to go "off-hook." (ATA works on the 1200hc, by the way.) The usual way to set auto-answer on Hayes compatibles requires setting the S0 register to a non-zero number — in other words, telling it to pick up the phone after that number of rings. Zero (the usual default) means it won't answer at all, as in `ATS0±2`.

A complete list of S registers, modem commands, what they do and how to use them, should appear in your modem's manual.

Hook up and use

Hooking up a modem usable with the ST is usually as simple as purchasing a cable with appropriate 25-pin RS232 serial connectors, plugging one end into the modem and one into the ST's modem port. It also helps if you plug in the modem's power supply (the ST's modem port doesn't provide power for modems) and plug a phone cable from the modem into a phone jack.

Powering up is pretty simple. Unlike disk drives, which usually must be turned on prior to booting the computer, in most cases it doesn't matter when you turn on the modem, although some modems seem to do better if turned on after the computer proper.

Of course, the modem itself isn't of much use unless you have some software to run on it. Telecommunications software is usually known as "terminal" software, meaning it makes your computer/modem system act like a data terminal. The software handles the job of controlling the modem, and sending data to and receiving it from the modem. Some modems come packaged with such software, but as time goes on, less and less of this "bundling" is done, as the packaged software is rarely what the user needs.

One piece of terminal software came with your ST when you bought it. It's the VT-52 terminal emulator desk accessory (`DESK2.ACC` or `EMULATOR.ACC`), which, as the name suggests, makes your computer and modem act like a VT-52 data terminal. This software is very limited, and most users find it useless because it completely lacks any documentation. The VT-52 emulator can be used, but be forewarned that it is a terminal *only*. It features none of the niceties of a good terminal program. To dial with it you must issue dial commands directly to your modem, etc. It does not feature a capture buffer (where the text from your session is held in a buffer in RAM and/or saved to disk) nor can you upload or download files (upload means to send a file *up* to the system you are communicating with, download means to pull it *down* from the other system). You can type and read text, and that's about it. Commands for the emulator are Escape-key combinations (like pressing Escape then B).

You're much better off buying a real telecommunications program, like Flash, Interlink ST, or even ST Talk Professional (if it's even shipped yet). Such programs make life much easier. They store lists of

phone numbers that the modem can dial, feature capture buffers, support various types of data transfer (uploading and downloading data) and disk and printer support. The more powerful ones have capture buffers that act like mini word processors, letting you edit the text you've saved. They may also let you make "scripts" of commands which will allow your computer to automatically do certain things. For example, such a "script" might be used to make your modem dial a number, and it would then automatically "log on," supplying the system you are calling with information such as your name, password, etc. In some cases, these are so powerful that they can call a system, log on, leave and read messages, etc., and hang up, all without human assistance!

One of the nicest features of most terminal programs is their ability to use a "type ahead buffer." This buffer is usually a single line editor, which allows you to type your commands, use cursor and other editing keys, and only send that line to the other system when you press RETURN. This is handy because editing text on many systems is difficult or slow, if not nearly impossible! Furthermore, you can often edit the line you just sent and re-send it if you goofed typing it the first time!

Even better, some capture buffers allow you to mark certain sections of text as "blocks" which can be saved to disk individually (sans the rest of the buffer contents) or even uploaded to the system you are connected with! You can pretype entire messages in the capture buffer, log on, and upload it.

The odds of getting even

Although the manuals for your terminal program and modem are the best means you have for learning to use them, there are a number of nasty, brutish little details that such manuals often mention in passing that can be a major headache to new users if they fail to have certain options set properly. These "four horsemen" of the beginner's apocalypse are known as "Parity," "Duplex," "Line feeds," and "Block size." Terrible names and terrible pains if you don't know about them!

Parity, if any, sets a "parity bit," an error-checking function used to insure that the data is received exactly as it was sent. Errors occur because of the curse of "line noise," that is, background distortion on phone lines that looks like data to the receiving modem. Hence the need

for error checking. However, parity is usually set to NONE, because most good file transfer systems either set it themselves or use their own form of error checking. This usually has little effect on the usual text and typing done in "terminal mode" (when your computer is accessing another and you are reading text, etc.), because such text is usually sent "blind." This means that the receiving computer does not echo back what it received so that the transmitting system can insure it was received correctly. In error-checked modes, such as "echoing" is standard procedure. If you are having trouble with garbled data from a system you are calling, you should try changing the parity setting from your terminal program.

Duplex sets the manner in which data is transmitted, either both directions at once (full duplex) or only one way at a time (half duplex.) This is difficult to describe, but the end result of it is that on full duplex, what your computer sends to the other computer is also echoed to your own screen. In this manner, you can see what you're typing. However, if the computer you're communicating with is also echoing the text you send back, you're liable to see something like IIIIEELLOO when you typed HELLO. On half duplex what you type isn't echoed, and, depending on what the other computer is doing, either you will see what you type properly, or you won't see anything at all!

The ground rule with duplex is simple. If you are on half duplex and can't see what you type, try going to full. If you are on full and are seeing double characters for everything you send, switch to half. If everything looks OK, leave it alone, unless there is a person on the other end of the connection who can't see what you are typing, in which case you might want to experiment a bit with duplex settings on both ends until both of you can see what you need to. There are also some echo options which can be selected in some terminal programs, that may alleviate difficulties changing duplex alone won't solve.

Line feeds are little known by many users of computers, though they happen all the time. When you press RETURN or ENTER on a computer keyboard, usually what happens is the computer carries out a "carriage return/linefeed" combination. The carriage return moves the cursor back to the left margin of the screen. The line feed then drops the cursor down to the next line. Some terminals have a

separate Line Feed key, which is used in combination with or in lieu of a RETURN or ENTER key. (Some personal computers have line-feed keys too, check out a Kaypro running CP/M for an example.)

Your ST, modem and software are only "emulating" a data terminal, and do not feature a line feed key. Some systems treat the value for RETURN or ENTER as a carriage return *only*. You have to send a line feed character separately! Fortunately, most good terminal programs have a line feeds on/off toggle control somewhere. If you find that pressing RETURN returns your cursor to screen left, but doesn't go down a line, try toggling your terminal software's line feed option. Little is more frustrating than typing over what you've typed before, or what another user is typing to you at that same instant!

Block size refers to the size of a data "packet" sent in an error-checked transmission mode, like X modem. The size is usually measured in bytes. The most common size, used by the very common X modem transfer protocol, is 128 bytes (1/8th of a K.) It takes eight standard X modem blocks to make 1K (kilobyte) of data. A modified version of X modem, often called Y modem, uses 1024 byte blocks, meaning each block sent is 1K of data. Some computers do weird, nonstandard things. The dreaded C-64's modem programs often default to sending 256 byte blocks! It's easy to get confused at times. For example, in Flash you can set the X modem protocol to 128 or 1024 byte blocks. The latter is treated as Y modem, but if you forget to change this when you go to a system that doesn't support 1024 byte blocks, you won't be able to transfer any files. If you ever find that you cannot seem to send or receive data when using a file transfer protocol like X modem or Y modem (there's even a Zmodem!) try seeing whether you can change the block size. (Be warned that some terminal programs will not allow you to set this!)

On line survival

With a modem you can call all kinds of computer systems, from your friend's own personal computer to mainframes, superminis, and networks like GENie or Delphi. There are not too many standards between all these systems, so it's easy to get confused. Many BBS systems (bulletin boards) are dissimilar in all respects from one another, with no similarity in access, log-on, help commands, or log-off (hang up) procedures. Still, if you get

stuck where you don't want, don't panic, there are some things you can try. For example, the most common of all "abort" commands is the Control-C combination. In many cases, one of these alone is enough to blast you out of whatever you've stranded yourself in. In other cases, it takes multiple entries of this command. On Delphi for example, you can abort a file upload or download by rapidly pressing Control-C three times in a row. It's unlikely you'll accidentally hit such a control-key combination three times, so that way Delphi is sure you really want to quit.

Control-X and Control-Z are sometimes used, and may or may not help. Often, if you don't know where you are, what to do, or can't see a menu, typing a question mark (?) and pressing RETURN will provide a menu.

If you don't know how to log off of a system, some common commands are BYE, OFF, LOG or LOGOFF. "L" for "logoff" might work from a BBS menu. You should always try to log off properly, not just hang up or turn off your modem, because the other system might not realize what happened and think you are still on line. Some major networks can take a while to realize you're gone if you just hang up, and may still bill you for up to 20 minutes or more of on-line time when you weren't there, all because you failed to "sign off" properly!

In a real emergency, you *can* resort to turning off your modem, as that will break the connection. It's not really the best way to handle things, but when all else fails. . . .

Log off

This article, like others on so diverse and complex topics as telecommunications, tends to be a mish-mash of information. As I stated earlier, additional and worthwhile sources of information are in any documentation you have for your modem or terminal software. Also, the aforementioned original "A Baudy Tale" article may also be of help.

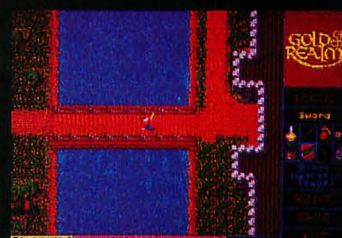
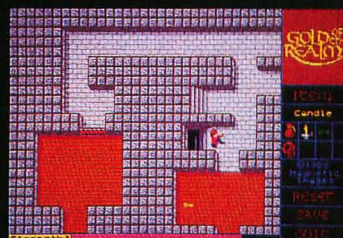
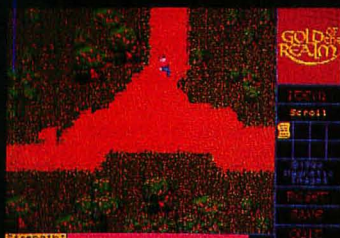
Allergic to all things Commodore, Maurice Molyneaux is an author and artist who—when not writing articles for **ST-Log**—continues to struggle with a recalcitrant eight-year-old science fiction novel, paints, illustrates and uses his ST for "every conceivable task." Despite a ridiculously French name, he claims to having been born in Vicenza, Italy, and denies vicious rumors that he eats escargot and calamari when computing. His Delphi username is MAURICEM.

For
Atari ST Series
Computers

GOLD OF the REALM



- Explore four different castles in search of hidden treasure!
- Over 300 different full color screens.
- Three degrees of difficulty.



P.O. Box 17422
Phoenix, AZ 85011
(602) 265-7849

Magnetic
Images™

Atari ST is a trademark
of Atari Corporation.

CIRCLE #114 ON READER SERVICE CARD.

IAN'S QUEST

by Ian Chadwick

It's been a long time since I wrote one of these columns. *ST-Log* is just out now and my column in it was written last fall, six or more months before it finally saw print. It's a trifle dated. . . I can't even remember the contents of any subsequent columns. Well, back to work, then.

Of late, I've had the opportunity to work with WordPerfect 4.1 and Microsoft Write for the ST. The former crashes a lot and won't always perform as ordered. The latter is a weak-kneed offering not a whit better than anything already on the market. Both are a *lot* more expensive than any of the competition as well.

Both programs are the products of large, wealthy and important software publishers—WordPerfect Corp. and Microsoft—whose strengths in the PC and Mac worlds are well known. Both companies must be highly embarrassed by these products, although for different reasons.

WP appeared in late fall, '87. Four revisions were shipped out by the end of January '88. And there are *still* serious bugs in it. The program merrily crashes, performs erratically, freezes up and dies with disturbing frequency. It does not inspire confidence.

What's the latest version? Well, at the time of this writing, it's January 29. But the versions *aren't* numbered so you can tell what you have—they all say 4.1. Rely on the file creation dates on the disks. A "clean" version of WP was slated for release in late March. Perhaps we've all received it now.

WP has a feature list as long as your arm. Very impressive, and if they all worked properly, it would be truly amazing (remember, they all *do* work on the PC). It has a super spelling checker and thesaurus, macro commands, on-screen column display, flexible printer control and a whole lot more—more than I dare list here. One can understand, at least, why the program costs so much—it offers value for the money.

WP 4.1 made a big hit in the PC world a couple of years back. But it was topped by 4.2 a year ago. 4.1 for the ST is a close work-alike to the PC version. Why didn't they bring out 4.2 for the ST, rather than 4.1? Why did they slavishly tie the ST version to the PC, rather than take advantage of the ST's various features such as graphics? Why did they release a flawed version that should have been kept in the beta stage for a few more months? Good questions.

This isn't typical of WordPerfect Corp.—they have a good reputation developed over four years of hard work. Obviously the ST marketplace doesn't have the potential of the MS-DOS world, so perhaps efforts to perfect the product aren't quite as active as we would hope—after all, they also sell a version for the Amiga, Apple II and a version 5.0 for the PC is about to be released. All these products take time, effort and money.

WordPerfect Corp. squashed rumours that they would pull out of the Atari market, due to rampant piracy. WP was found on at least three pirate BBSs, but they are still making the effort to stay with us. That shows a serious commitment on their part. If you want them to remain part of this market, then show them the same amount of respect and *don't* make or accept any pirate copies of their product. If they pull out, the chance of anyone else in their league coming over to the ST is slightly less than your being hit by Halley's comet.

I have faith that WP will finally come to its own—that the bugs will be fixed and the program will be finished. It was suggested that they might release 4.2 for the ST a year or so from now. Don't hold your breath—it will depend on sales of 4.1 to justify the expense and effort. Of course, the pirates could ruin it for all of us. . . .

Microsoft Write is another product altogether. It leaves me with no hope at all. The box art should have alerted me—it's hideous.

Write is similar to Word 1.0 for the Mac, a product that, when released four years ago, wasn't much of a hit. Don't confuse Write for the ST with the excellent program of that name bundled with Microsoft Windows. No hope. And don't think it bears any resemblance to the latest versions of Word on either the Mac or the PC. Will Microsoft upgrade the ST version? Add new features? Improve the display? Not within the Age of Mammals, anyway.

Write whimpers. It does little that other word processors don't do at half the price. It looks like it's WYSIWYG, but aside from screen fonts, it can't show text as it will appear when printed. Multiple columns, for example, don't display across the screen. Headers, footers and footnotes don't show on the page where they print. There isn't a spelling checker or thesaurus, and printer drivers are limited to the SMM804, Atari laser and Epson-compatibles. The list of features is depressingly short and unimpressive.

Like WP, Write copies from another system—the Mac in this instance. It does it so well that it even ignores the right mouse button (the Mac only has one mouse button. . .). Like WP, it is also not the latest version the company has to offer everyone else. But unlike WP, it leaves a distinctly unpleasant aftertaste. Why would anyone bother bringing out such a lame duck? And then charging top dollar for it! Surely Jack and the boys don't think we're *that* gullible!?

I heard that Microsoft washed its hands of the Write project early on and gave it to Atari to finish development (more than a year in the doing thereof). A local source even told me that only a single, co-op student programmer was assigned to the project. Why doesn't that surprise me? Even if it isn't true, the program *looks* like it was a back-burner project. I'm surprised that Microsoft even allowed its name to be associated with the final product. It looks like no one was ever consulted on the concept of word processors and what the consumer might want in one.

On the back of the Write package it states: "Team Microsoft Write with an Atari Mega computer and the Atari SLM804 laser printer for an excellent desktop publishing system." Write can't import or print graphics, doesn't do kerning, can't produce rules, won't display multiple columns—the simplest things required by desktop publishing. To call this statement misleading doesn't do it justice. I say it's an out-

right lie. Write is *absolutely* unsuited for desktop publishing.

WP and Write are major software disappointments, although as I said earlier, for different reasons. I had high hopes that these two companies—major dynamos in the industry and major competitors for the same market (word processing) would race to offer the best programs they could. Neither one achieved that. I'm willing to wait for the finished WP, but as far as I'm concerned, Write isn't worth the shelf space it consumes.

Way back when, I wrote about the Atari PC1. It's been available in Canada since the end of December but I understand not in the USA. The PC2 started shipping up here in early March. I've been led to believe that the PC series is unloved and unwanted by the U.S. Atari folk. No one seems to care to see them released in the USA. Maybe because they're tired of flogging the mediocre?

The PC1 is a good idea in a handsome package, but poorly implemented as a consumer product—the design lacks foresight and comprehension of the marketplace demands. It has built-in monochrome, CGA and EGA graphics and a Microsoft compatible mouse. It comes with a good amber monitor but bundling discourages making a purchase of a color or high-res monitor later. The monitor port is standard TTL, without composite or RGB output. So far so good. It goes pretty much downhill from here.

The PC1 has a 720K 3.5" internal floppy drive and can accept another two externally. The nice thing is that they're standard ST314 drives. The nasty thing is that Atari doesn't make a 5.25" drive for the machine and there's still a lot of software out there that's not available on 3.5" disks. It also makes for difficult data transfer if your office machine uses 1.2MB floppies!

The basic memory is 512K RAM, expandable to 640K maximum. With no expansion cards, forget RAMdisks or additional EMS or LIMS memory cards. This is a severe restriction. Many PC programs make use of expanded memory—Lotus 1-2-3 for example.

The PC1's biggest failing is the lack of standard expansion slots. There is only a single custom bus—for which the schematics have not been released! The myriad of PC/XT cards available for almost every other PC aren't of any use with the PC1. Forget internal clock calendars, modems, new graphics drivers, additional parallel or serial

ports, joysticks, fax cards, J Laser cards, RAM cards or even a hard drive. If you get anything, it must connect through the existing serial, parallel or floppy port. See what I mean about short-sightedness?

The PC2 is a better machine but still a long way from perfect. It offers pretty much everything the PC1 has, plus a bit more compatibility with the real-world PC market. The biggest differ-

Both machines come with a two-button ST mouse (although the best PC mice, like Logitech's, are the three-button type) but no utilities to create menus, drivers, and so on for non-mouse programs (with which every PC mouse comes, of course).

ence is that it can house a 20mb hard drive and a single 5.25" floppy or two floppies. 20mb, however, is a small drive—I filled 40mb on my AT clone in a few months! Also, you apparently can't mix 3.5" and 5.25" drive types to take advantage of both (I use one of each on my system).

The PC2 has four standard expansion slots, which is adequate for most users. It doesn't have the external floppy port, but has the parallel, serial and mouse ports like the PC1.

Both machines are extremely IBM/PC compatible. They are switchable between 4.77 and 8 mhz speeds, but I don't know how many, if any, wait-states that includes, so the realized speed may be less. 4.77 mhz really drags on the 8088, and I can't imagine why anyone with even a semblance of brain power would work at that slow a speed. Many competing XT clones offer 10mhz "turbo" speeds.

The keyboard, alas, is lightweight and the keys appear to be cut from the ST mold—a trifle larger than standard PC size, so clumsy and awkward—as the ST keyboard is. However, it is detachable. You can replace it with a Keytronics or a similar keyboard.

Both machines come with MS-DOS 3.21, GWBASIC, GEM desktop, GEM Write and GEM Paint. I don't know why Atari failed to provide the latest version of DOS 3.3. I also wonder why the PC1 rates all of this software when the ST limped along with an unfinished version of NeoChrome, third-rate 1ST Word, a slow and cranky BASIC and GEM without GDOS. Think maybe we got the nasty end of the stick?

All in all, I can't recommend the PC1; the design is too restrictive and there are a lot of expandable competitors at the same price with a few more options

from which to choose. It's not even suited as a home machine, because it can't be upgraded or enhanced. However, if you need an expensive paperweight....

The PC2 is a reasonable entry-level system for the home user. The limitations on floppy drives (number and type) and hard drive size certainly lower it below any professional or business user category. But as a backup machine, when fully loaded, it should be adequate.

Other notes

In my next column, I'll look at an entirely new ST product—ST X*Press designed by Alan Page, co-author of Flash. Imagine being able to get all the news 24 hours a day—international, local, business, your favorite sports stories, market updates, weather, fashion, and so on. Not a newspaper—they'd be on your ST. Instead of rummaging through page after page looking for stories of interest, you could collect stories with specific keywords and save them to disk to read later. That's just part of what ST X*Press does when combined with your TV cable. It does a whole lot more too, but I'll talk about it next time. A very exciting new offering for this market.

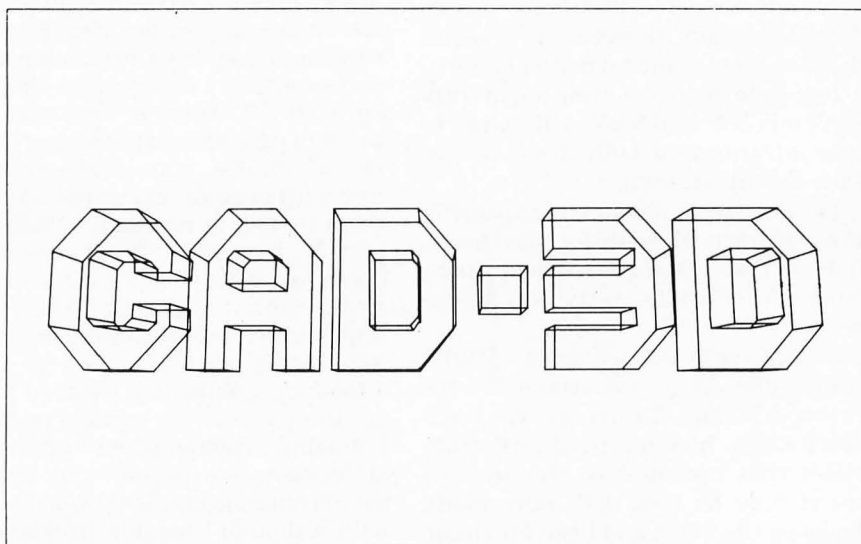
Desktop publishing: I haven't seen anything except the earliest release of Publishing Partner (their "professional" version is expected—one hopes a significant improvement over the original), although I see new packages are out there now (Fleet Street publisher for one). I hope to be able to get the newest releases and compare them with Xerox's Ventura Publisher for the PC in the near future.

I've played a lot of Wargame Construction Set, from SSI, recently. A thinking person's game—challenging but fun. I'll review it soon, along with several new products including Juggler from Michtron, GFA Artist, Gunship and Defender of the Crown.

My thanks to readers who sent me PD programs (especially the GFA listings) after my disk hard crashed. I have since heard of others losing their data the same way. I appreciate everyone's support.

Cheers!

Ian Chadwick is a Toronto-based technical writer specializing in desktop publishing. He is currently designing several wargames, including one for the ST. He is looking for reference books on Napoleonic warfare, campaigns, armies, etc. Contact him at: 47 Oakcrest Ave, Toronto, Ont., Canada, M4C 1B4.



THE CYBER STUDIO

***CAD-3D 2.0** by Tom Hudson
 ***CYBERMATE** by Mark Kimball
 2-disk set (requires one megabyte RAM)
 \$89.95

STEREOTEK 3D GLASSES

by LC Technologies
 \$149.95
 Add-on glasses only: \$99.95

3D DEVELOPER'S DISK

For CAD-3D 2.0
 by Tom Hudson
 \$29.95

All products marketed by:
The Catalog
 524 Second Street
 San Francisco, CA 94107
 Orders: (800) 443-0100
 Customer Service: (415) 957-0886

Review by Andy Eddy
 and Charles F. Johnson

RE VIEW

**You can do
 partial spins
 of whatever
 percentage of
 degrees you
 choose, to
 bring about
 more
 intricate
 shapes.**

Since their introduction, computers have been touted as omnipotent tools, with the power to increase productivity while liberating human beings from the tedium of repetitious tasks. The latest 68000-based personal computers—among them, the Atari ST—promise to bring the number-crunching strength of the \$40,000 workstation to the common man. All that's left is for programmers to create applications that will utilize that muscle for practical, real-world purposes.

One of the more powerful facets of computer usage these days is *Computer-Aided Design* (CAD). CAD software offers a way to design and visualize objects *before* money is spent—and perhaps wasted—on creating the

final product. Furthermore, the capable ST makes an inexpensive station for CAD duties.

Tom Hudson's original CAD-3D has been enhanced and streamlined for efficiency to provide more than just simple three-view drawings. At the same time, many "bells and whistles" have been added to **Stereo CAD-3D 2.0**, making it even easier to design objects, letting the user build complex groups of CAD-3D objects through the use of standard camera movements (zooms and rotations, for example) and object manipulations. It also bears the distinction of being the first ST product that *requires* one megabyte of RAM to operate.

By far the most exciting addition is

animation through frame-by-frame recording to disk (we'll discuss the details shortly). To assist in that goal, The Catalog provides an animation editing language called **Cybermate** with CAD-3D 2.0. The combined package is named **Cyber Studio**.

The Cybermate language was created by Mark Kimball, who was primarily involved in the **StereoTek** project at Tektronix. The StereoTek glasses are a new approach to 3D; instead of traditional red and green lenses, these glasses contain liquid crystal shutters that alternately open and close in tight synchronization with the ST monitor's screen refresh rate. The program alternates between two different screens at the same rate, so fast that only a slight flickering is noticeable. A View-Master effect is achieved because each eye sees a separate image from a slightly different perspective. CAD-3D 2.0 and Cybermate both support the StereoTek glasses, so entire 3D animations can be created.

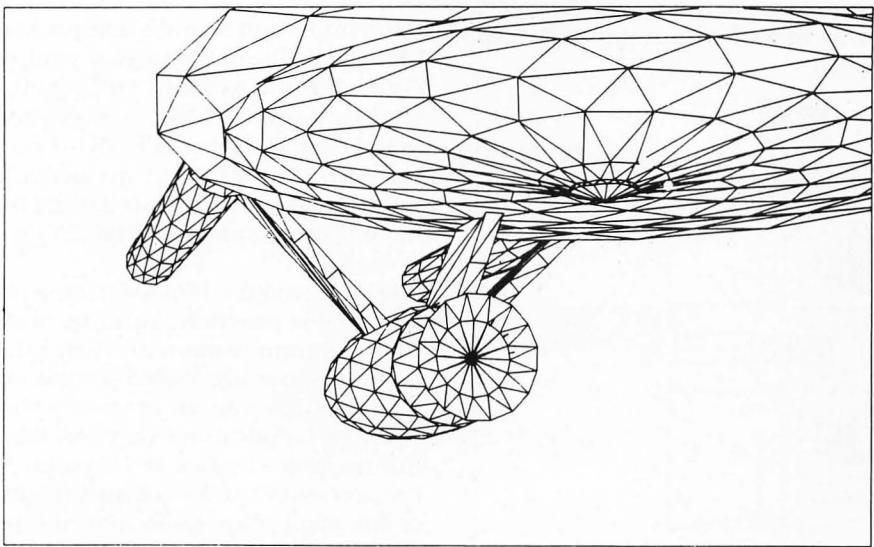
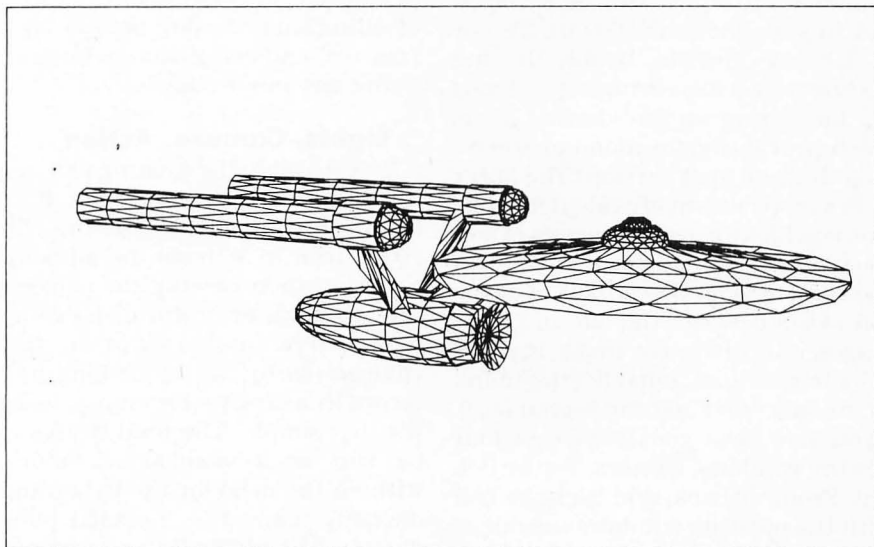
Put Away Your Hammer And Nails . . .

Comparing the new program with its predecessor, CAD 2.0 (as we'll refer to it from here on) primarily has a refined user interface. The new package is icon-driven; anything you need can be achieved by sliding the mouse pointer to a specific icon and clicking. This differs from CAD 1.0 (still available by itself), which works entirely from drop-down menus.

Beyond that, the scope of the package is immense. For starters, one of the icons contains a bank of "primitives," basic shapes like spheres, cubes and wedges that can be used as building blocks to create more intricate shapes. And you can easily create your own set of primitives with two included utilities.

The "Spin" routine is used for lathing symmetrical objects—such as a wine glass or light bulb—from an outline. Once this "template" is completed, clicking on the "Do spin" line from the drop-down menu magically carves out the item and places it in your viewing windows. You can also do partial spins of whatever percentage or number of degrees you choose, to bring about more intricate shapes.

The other object-sculpturing tool is the "Extrude" routine, which works like a hot knife to cut non-symmetrical pieces. Placing the points of the object's perimeter on the template and clicking



on "Do extrude" will similarly result in a 3D object.

These two easy-to-use routines give great control over the items you work with. Though you can use an overlaid grid for precision plotting, it's difficult to make rounded edges. You must click the mouse exactly on each point of the arc; freehand drawing of circles and arcs is close to impossible. Some companies are distributing ST-compatible graphics tablets that should help with object creation.

A promising and less expensive alternative is a separate program (currently in the works and possibly out by the time you read this) that will allow you to create CAD-3D objects from DEGAS picture files. Many of the necessary tools for making cleaner curves and, more importantly, editing oddball shapes are inherently designed into DEGAS, whereas attempting those tasks with either of the Spin or Extrude routines will likely result in a struggle. If this product fulfills its promise, it could present a library of animation characters—such as digitized acquisitions—and enable quality computer "movies."

Stepping beyond the basics, complex creations can be constructed by taking multiple objects and combining them mathematically. Four interactions—Adding ("gluing" together), Subtracting (cutting the overlap of one object from another), And (making a separate item from the overlap area of two objects), or Stamping (to emboss a "decal" of one object's impression, like letters on an airplane wing)—can be called by the "Join" icon, giving you the ability to make intricate models from simpler,

building block shapes.

You're not limited to just making objects, either. Once you've constructed an object, CAD 2.0 frees you to alter parameters that control any part of object viewing. The resultant images can be viewed in four modes—Wireframe, Solid, Hidden Line Removed (a combination of Solid and Wireframe), or Solid Outline (a Solid shape with highlighted edges)—and even particulars like lighting (strength and locations) and coloring (both object and background) can be varied to suit your needs. Additionally, you can use a DEGAS picture as a backdrop for further ambience and artistic effect.

To view the scene as a full-screen depiction, a click on the "Superview" icon will display whatever is in the camera view, which can be saved to disk in your choice of DEGAS, NEO or C.O.L.R. file formats. Double-clicking the Superview icon brings up a dialog box for setting up the viewing mode (listed previously), in mono or stereo, or in draft or final version. The latter gives a more accurately calculated picture, less likely to contain errors of perspective. For stereo viewing, you can also control what percentage of deviation exists between the left and right images, and where the image appears "in" the monitor, "outside" the monitor, or "at center" (at the screen face).

You also have your choice of four viewing windows: Camera, Top or Bottom, Front or Back, and Right or Left (with the opposite windows capable of being toggled with each other). Your viewpoint can be varied by moving the camera angle or zoom, and the object(s) can be moved within any active win-

dow (excluding the Camera window, which is only for "sightseeing") by dragging it with the mouse. Incidentally, any window can be set to fill half the screen for better control over the adjustment and refinement of the object.

Additional tools are available to manipulate objects within the CAD 2.0 "universe"—the cube-shaped world that your objects inhabit. There are icons for moving, rotating and sizing; including group or single object rotation; rotation around the universe's center point, a group's center point or user-specified point; and camera rotation and banking.

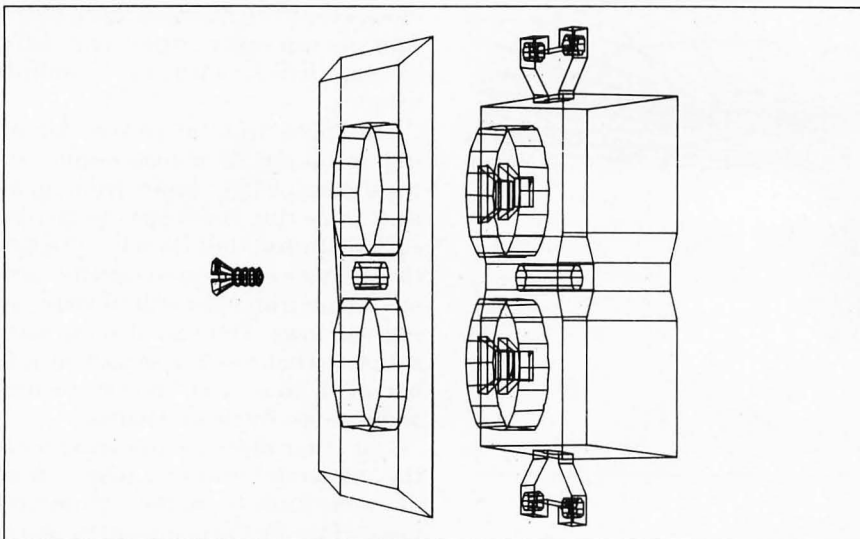
For any CAD program to work in a professional capacity, it needs to be able to identify scale and establish its accuracy. Precision scaling of objects is amply provided for in CAD 2.0 by labelling one of your objects as the "master" and using it as a reference in sizing any other objects.

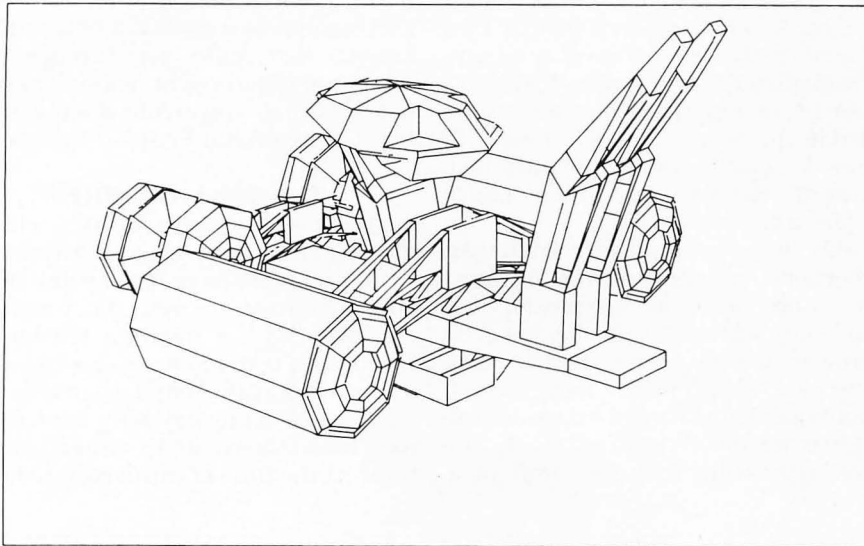
Lights, Camera, Action . . .

Now we get to the exciting part: making your designs come to life. It's accomplished by clicking on the "film-strip" icon to activate the animation recorder, then moving the objects in the windows or shifting the camera perspective, and recording those changes one-by-one (by clicking on the record icon) to save the "frame" to disk. It's that simple. The final results can be run as a stand-alone "movie," without the need for the CyberStudio disk, by using the included public-domain ANIMATE2.PRGM file (available in the Databases of Analog's ST SIG on Delphi).

It's important to note that products like the Future Design Disk and the Fonts, Primitives, Hints and Tips Disk are being released to give users additional components and tools for working with CAD-3D. With expansion in mind, Hudson made CAD 2.0 the first module of a potentially larger "Cyber" system.

GEM contains built-in commands that enable programs running on the ST to communicate with each other, using a structure called a "message pipe." In this way, an accessory and a running application may exchange information—in fact, it's possible for the accessory to take complete control of the application (assuming the programs are written to work together, of course). This is the principle behind the pipeline; it allows programmers to write accessories that can greatly ex-





tend CAD-3D's capacities. To further this end, The Catalog has released the 3D Developer's Disk, which contains several practical programming examples of how to access the pipeline, and complete documentation on all the message protocols supported by CAD 2.0.

The manuals for the 3D Developer's Disk and Cybermate are provided in the form of text files on disk that you must print out yourself. . . a not-too-minor inconvenience, since these files produce over 70 pages of hardcopy *each*. The files are formatted for a standard 8 1/2" by 11" page, but you have to make sure your printhead is set exactly at the perforation line before you start printing, or the headers and footers will be incorrectly aligned. A large portion of the text in the 3D Developer's Disk documentation was published previously in STart magazine, Summer 1987.

The CYBSMASH and PD3DCTL accessories (see below) are examples of what can be achieved by using the message pipeline. Just about every function of CAD-3D 2.0 can be remotely controlled; hopefully, programmers will begin using this ability to add features to CAD-3D, like real-time machine language object rotation routines, or better compression routines for animation files.

The first example—and Hudson's acknowledged testbed—of this potential is the PD3DCTL desk accessory. It's a strange name for a powerful tool, but here's how the name breaks down: PD stands for Public Domain; 3D refers to the CAD program; CTL is an acronym for Control.

To summarize, PD3DCTL (which is

The remote camera could be placed anywhere in the CAD 2.0 "universe," flying around and even through objects.

also available in the ST SIG on Delphi) is a freeware taste of the forthcoming Cyber Control language; the aim being automation of the animation sequence. Visualizing the tedium of the above situation (recording a frame, moving the objects or camera a tiny bit, recording the frame, etc.), you can see how beneficial it would be to have the computer handle those tasks. After all, as we mentioned at the top of the article, this is the strongest feature of computers.

With PD3DCTL/Cyber Control, you can write a BASIC-like script file using a simple programming language to take control of CAD 2.0's desktop and

the animation sequence. It can take hours to build an animation by hand; running things from your command script can save a lot of time and free you to do other things. It's a powerful tool with loops, variables and functions, resulting in an "Automated Animation Construction Set."

To show Cyber Control's strength, they've also added some features not programmed into CAD-3D, such as remote cameras; at present, the only camera in CAD 2.0 is always facing the center of the CAD universe, a hindrance to certain effects in an animation. The remote camera could be placed anywhere in the CAD 2.0 "universe," flying around and even through objects.

The Cybermate Language.

Included in every CAD 2.0 package is Cybermate, an "object-oriented" programming language very similar to Forth (in fact, supposedly based on MT Forth), which gives you the ability to manipulate the "delta" animation files created by CAD 2.0. "Delta Compression" is the data-compaction scheme used by CAD 2.0 to keep the size of animation files within manageable limits by saving only the frame-to-frame changes, instead of saving an entire 32K screen for each animation frame.

Cybermate allows you to control and alter many aspects of these delta files—cutting frames in and out of sequences (similar to "splicing" film or video tape), stepping through an animation sequence, changing its speed or screen location, setting loop points, chaining to other delta files (to get around memory limitations and allow much larger animation sequences), pixel-by-pixel dissolves from one "scene" to the next, and more. Cybermate also allows you to include files generated by The Catalog's 'G.I.S.T. sound editor program, to spice up your mini-movies with appropriate sound effects.

The Forth Connection.

Cybermate's familial relationship with Forth is both a strength and a weakness. Although Forth has many advocates, a case could be made that it is not the easiest or most intuitive language for people to learn. As in Forth, all Cybermate math operations are performed using Reverse Polish Notation; to add two and three the Cybermate statement would be "2 3 +," not the way most people are accustomed to thinking about math. Cybermate programs are organized into one or more 16-line "screens," another Forth con-

vention. To edit "Cybercode," you must load each screen individually, edit the text it contains, and save it to disk before editing another screen; you can have only one screen's worth of code in memory at any one time. This procedure is rather clumsy, especially for casual programmers accustomed to the full-screen editors used by most implementations of BASIC. We strongly suggest the use of a RAM disk or a hard disk while editing Cybercode—otherwise the constant disk accesses will have you in the rubber room in short order.

On the other hand, Forth is generally recognized as a good language for motion control applications, which makes it particularly well-suited for animation editing on the ST. The "screen" editing concept (although inconvenient at times) has the advantage of freeing up lots of memory for animation objects, which are usually quite large. Forth is a "user-extensible" language; this means that you can develop routines (called "words") for general purposes and actually make them part of the language—like adding your own commands. Also, since Forth/Cybermate is a compiled language, it's possible to produce a .PRG (or "runtime") version of an animation that can be run as a stand-alone file, without the need for Cybermate or any special display program.

The Cybermate language system has four primary modes of operation: Interpreter, Text Editor, Preview Editor, and Display. Interpreter mode acts much like BASIC's "direct" mode. You can type in any legal Cybermate command and it will be acted upon immediately. Text edit mode lets you create and edit screens of Cybercode, which will comprise your animation program. The Text Editor has many word processor-like features, including cut-and-paste and Insert/Replace text entry modes. The Preview Editor displays your animation frame-by-frame, showing you the current time values and color settings and allowing you to alter them. Preview Edit mode is useful for determining exact timing of animation sequences, without having to edit a screen and re-compile it. Display mode is entered when you "run" a Cybermate program.

Cybermate is referred to as "object-oriented" because of the way it handles animation data files; when a file is loaded, Cybermate creates an internal structure called an "object" using the data in that file. You give each object a unique name, and refer to the object by that name for all programming/editing operations. Objects can be of three

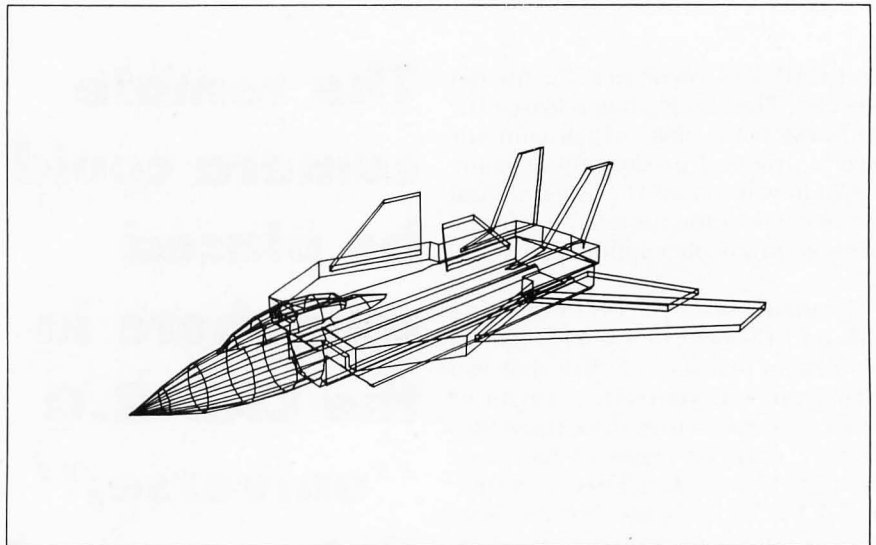
basic types: sound effect files produced by G.I.S.T., uncompressed picture files created by the DEGAS paint program, or sequence files created by CAD 2.0's record function. Once an object is defined in this way, it can be "cloned" to make a copy for working purposes. These are the main building blocks of a Cybermate animation.

Like most computer languages, Cybermate has variables, looping structures, and data areas. A typical Cybermate program might display the first frame of an animation sequence, then enter a "TICK/TOCK" loop, which would update the screen image according to Cybermate's internal clock. The clock regulating the execution of a

bly the "forth programmer's Bible"). The manual does contain a well-written tutorial that walks you through the main functions of Cybermate, using 3D objects which you create when you go through the main CAD 2.0 tutorial.

The Cyber-verdict.

Cybermate works well as a whole, although a few annoying quirks are evident. You must be in low resolution to run Cybermate...yet, the program starts up with a medium resolution 80-column text screen. Since the program obviously handles resolution changes itself anyway, why not let the user load Cybermate in either resolution? At the time of this writing, there



TICK/TOCK loop can be adjusted to any value, to achieve faster or slower animation speeds. A TICK/TOCK loop is similar to a FOR/NEXT loop in BASIC; all code between the TICK and TOCK statements is executed at regular intervals, until the last frame in a sequence is detected or the loop is exited by other means.

Cybermate also supports some traditional Forth looping structures, like DO/LOOP, and BEGIN/UNTIL/REPEAT, and decisional structures like IF/THEN. A full range of logical and bitwise operators is available for integer math functions.

If you have no experience with Forth, be prepared to spend a good deal of time puzzling over the Cybercode examples contained in the CAD-3D package. At the end of the Cybermate manual is a short appendix titled "Generic Forth" which will help a little bit, but the manual is *not* intended to teach Forth. Instead, it recommends several books devoted to this purpose, including *Starting Forth* by Leo Brodie (argua-

ly no support for monochrome, even though CAD 2.0 can generate monochrome DELTA files. (Note: According to Catalog Marketing Director Gary Yost, monochrome monitor support is planned for Cybermate, and a version that works in monochrome may even be released by the time you read this.) Also, the interpreter mode's keyboard input has an odd, somewhat disconcerting feature; the backspace key does not delete characters, it just moves the cursor backwards, leaving the text intact.

It's important to note that Cybermate is probably not for everyone. Indeed, the manual states that "Cybermate is NOT the place to start in the Cyber Studio system." A comprehensive understanding of CAD 2.0 is essential before you can begin to tackle Cybermate; and even then you've got some work ahead of you. A background in some sort of programming will definitely help; a background in Forth would be even better. However, if you make the effort to climb Cybermate's rather steep

STPlus•STPlus•STPlus•STPlus

P.O. 1197, Berkeley, Ca. 94701

We all want the ST to grow, so let's BUY MORE SOFTWARE and discourage

BUSINESS

DBMan 4.0	175.00
Datamanager	56.00
Superbase	104.95
Trimbase	69.95
Phasar	63.95
Zoomracks 2	84.95
Base 2	42.95
The Informer	69.95
Wordperfect	189.95
1st Word Plus	69.95
Word Up!	74.95
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
Datatrive	35.00
STOneWrite	48.95
VIP GEM	104.95
DacEasy Payroll	48.00
DacEasy Acctg	52.00
WordWriter ST	56.00
SwiftCalc	56.00
EZ Calc by Royal	48.95
Analyze Spread	25.95
Final Word	99.95
PublishingPartner	140.00
T-works Publish It	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	35.00
Max Pack	35.00
Stuff	27.95
Flash 1.5	21.00
SBT accting ea.	275.00
Omni Res	27.95
Turbo ST(~blitter)	35.00
Dollars & Sense	69.95

GRAPHICS

Degas Elite	55.95
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
Athena 2	69.95

GAMES

Gunship	35.00
Shadowgate	35.00
Uninvited	35.00
Mouse Quest	14.00
Slaygon	27.95
Barbarian	27.95
Obliterator	27.95
Guantlet	35.00
Dark Castle	27.95
F-15 Strike Eagle	27.95
Star Trek-Rebel U.	27.95

MUSIC

<i>Passport</i>	
Master Tracks	280.00
MasterTracks Jr.	104.95
Midisoft Studio	69.95
<i>Hybrid Arts</i>	
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
D-50 Editor	call
<i>Voice Masters</i>	
Yamaha TX81Z	69.95
Roland AJ 1 & 2	69.95
Yamaha 21,27,100	55.95
Oasis Editor	175.95
Hybridswitch	21.95
ADAP Smptecue	175.95
upgrade old box	70.00
MIDI-MAZE	27.95
ADAP 2 direct to	2795.0
60mghd sampler	
Midplexer	249.95
<i>Dr. T's</i>	
KCSequencer	199.95
KCS 1.6 w/PVG	289.95
MIDI rec studio	27.95
NEW Copyist	139.95

How would you like to be an ST dealer? If you are interested, I am looking for a few limited partners to work with in areas which lack ST support. This is not a solicitation by Atari nor to circumvent Atari's network but an invitation to work with an established dealer to set up new dealerships. I am especially interested in college and business areas. Call (415)841-9183 to discuss

PROGRAMMING

GFA Basic	48.95
GFA Book	35.00
GFA Compiler	42.00
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	69.95p

GAMES

Tanglewood	27.95
Test Drive	35.00
Chessmastr2000	32.95
StarGliderbw&cl	32.95
Hunt for Red Oct	35.00
Police Quest	35.00
Allants	24.95
Allen 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
<i>Cheetah Midi Inst.</i>	
MK5 Keyboard	249.95
MK5 II	399.95
MK5 V	549.95
MK7 VA	794.95
MIDI Drum	339.95
Power Play Drum	369.95
Drum Interfacer	119.95
Synth Module	534.95
SMPTE to MIDI	349.95
DX Heaven editr	104.95
Korg, Kawai, etc	call
CZ Patch editor	79.95
CZ patches	39.95
DX patches	39.95
HARDWARE	
20 meg hard disk	558.95
30 meg	749.95
60 meg	1249.95
Atari CD-ROM	499.95

National (800) 433-6222 California (800) 874-4789 (415)849-8717 Prices subject to change without notice. We ship ANYWHERE! \$4.00 min S&H. No 1040's or Megs mail order. Hand delivery only, List plus \$100.

CIRCLE #115 ON READER SERVICE CARD.

learning curve, you'll find yourself in possession of a very impressive set of animation editing tools.

The Wrap-up.

This package is typically *Hudsonesque*—our term for the type of carefully-written, easily-worked software for which Tom is well-known. If the communications pipeline is imaginatively exploited, we may see Atari ST computers gaining more respect.

Microbyte Floppy Disk Drive

Paradox Enterprises, Inc.

Version A, 360K Nth \$235

Version B, 1.2MB Nth \$245

by Matthew J.W. Ratcliff

The Microbyte 5-1/4-inch floppy disk drive for the Atari ST is an extremely useful device to own—if you need one. If you have an IBM at work, or a PC at home, you'll probably want, if not need, this drive. (I tested the 360K version for this review.)

We know ST cannot run IBM programs (at least, not without an emulator), so what use is this drive? Let me give you a few examples. If you use Generic CADD at work, you might like to bring your designs home and use First CADD on those same files. (These programs don't use the same format, but are written by the same software house. The ST version of First CADD comes with a utility for performing the necessary conversions). A similar argument holds true for Lotus and VIP Professional, which are file compatible.

Moreover, if you write a lot, it's always convenient to be able to easily move text files between the two machines. I occasionally develop code for work at home, and vice versa (on my lunch breaks or course), so I've really had a chance to put the *Microbyte* through its paces. Currently, 5-1/4-inch floppies cost about one-fourth as much as the 3-1/2-inch diskettes, so the *Microbyte* is handy for archiving a lot of files economically.

The drive comes in a sturdy metal case (color matched to the ST system), and is a half-height mechanism, just like the current crop of DS/DD drives found in IBM PC XT compatibles. This unit employs the same power supply as Atari's 8-bit 1050 disk drive: a simple 9-volt AC output transformer (easily replaced).

Of course, it's as simple to hook up as any other ST drive. The interface cable is hard-wired into the drive, with a male connector for the computer or the other drive at the opposite end.

There's no output connector on this unit. This means that the *Microbyte* is drive B if you have a two-drive system. If you wish to use it as drive A, you cannot have a second floppy on the ST.

The connector is a bit oversized. Although I had no problem daisy chaining it to my old SF354 Atari drive, when I upgraded to the newer SF314, I found it very difficult to get the *Microbyte* connector to stay in place. Careful arrangement of the cables solved the problem.

Once the drive is hooked up to the ST, the busy light comes on and stays on while the computer is booting. When the system is up, and all is well with the *Microbyte*, the busy light pulsates dimly, like a heart beat, to let you know everything's A-okay with the unit.

Two programs come with the drive. The Paradox program must run from an auto folder at boot time, which sets a different *step rate* for the *Microbyte* drive. The head access time is slower for a 5-1/4-inch unit than than the 3-1/2-inch floppies normally used. If the step rate is not set properly, you'll get constant read and write errors.

The second program is a format utility, used to perform a double-sided, double-density 360K format of the 5-1/4-inch drive. (The 360K formatting we normally do is for a *single-sided* 3-1/2-inch unit; it just won't work for a 5-1/4-inch drive.) The biggest problem with this formatting utility is that disks prepared with it *cannot* be read on an IBM PC or compatible. This is frustrating because you must format all your 5-1/4-inch disks on the PC, if porting files is your primary goal. (A disk formatted on the PC can be read and written on *either* computer.)

The formatting program provided by Paradox writes all zeros to the empty sectors, as does the GEM format utility. However, the IBM expects to see the following bytes (this information courtesy of Joe Pierce, Delphi username JOEPIERCE), in the first four positions of track zero, sector zero:

PS7 FT6 \$EB, \$2C, \$90, \$49

If those bytes are not there, the IBM assumes the disk is blank, or damaged. When Joe gave me this data, I took an ST formatted 5-1/4-inch disk to work and used a disk editor to write those bytes out there. After the modification, the IBM no longer thought the disk was blank, but could read and write to the disk with no problems.

Paradox definitely needs to update its formatter with this information. I

would prefer an accessory version of the format utility, so a 5-1/4-inch disk can be formatted from a desktop menu, like the 3-1/2-inch disks.

Because of the unique format of a 5-1/4-inch disk, none of the hard drive backup utilities will work with the *Microbyte*, which, of course, is one of the uses I had planned for it. I have been archiving some valuable folders from the hard drive "by file" to the 5-1/4-inch disks with no problems, however. A hard drive backup utility, specifically written for the *Microbyte* drive, would certainly be useful.

One particular annoyance you will encounter with this drive is that the power switch is at the *rear*, making it difficult to access if you have a computer hutch or similar setup. You will also find it *necessary* to shut it off from time to time with certain commercial software. Some programs (World Championship Karate, for instance) detect the presence of a second drive. If they find one, they automatically expect a data disk to be there.

After several months of daily use, I've found the *Microbyte* to be a quiet, fast and reliable little workhorse. It comes in handy for prototyping files and making inexpensive backups of my files, and most important, keeping the *clutter* off my hard drive.

The hard disk drive HD+ ASTRA SYSTEMS

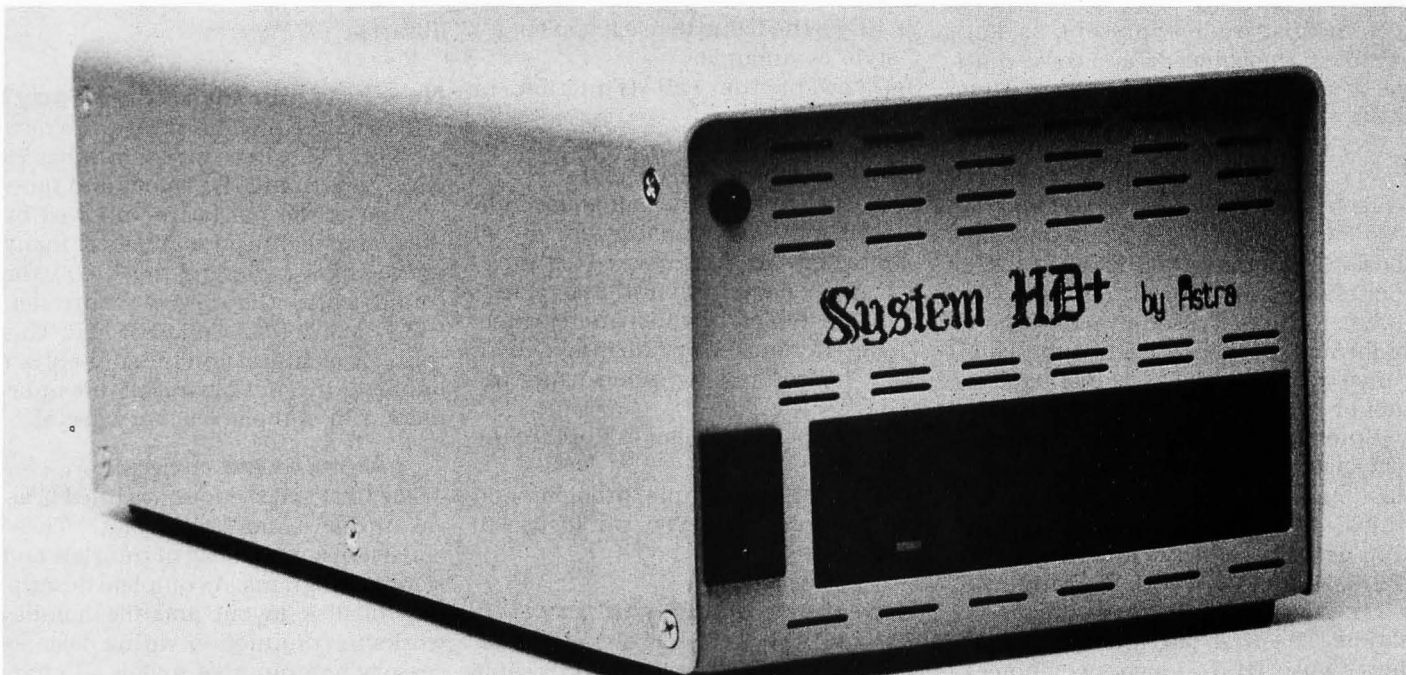
by David Plotkin

The **HD+** from Astra Systems is a combination 20-megabyte hard-disk drive and double-sided floppy, all packaged in a box no larger than other hard drives. It's ideal for 1040 owners, or those looking to upgrade one-drive systems.

The **HD+** comes preformatted into two 10-megabyte partitions, and can be used within minutes of unpacking. The ST's operating system cannot recognize disk drives larger than 16 megabytes, therefore you're unable to use the whole 20-megabyte drive as a single logical drive. What you do, instead, is split the single physical drive into two logical drives (such as "C" and "D").

The hard drive includes an extra-long cable for connecting the built-in floppy drive, allowing you to place the **HD+** farther away from your ST than would normally be possible. There's a "floppy in" connection, but no "floppy out" connection, so the floppy drive in the **HD+** *must* be the last one in the chain (drive B in a two-drive system).

Software includes a formatting pro-



gram (to reformat the disk and/or change the partition size), a backup program, a head parking program, and the hard drive install program. The backup program uses a special fast format and puts close to a megabyte of data on a double-sided disk. It can copy files which are larger than a floppy and do partial backups. The head park program allows you to lock the head in place before moving the drive. Always do this! The hard drive install program is placed in an AUTO folder on the boot-up disk and enables the ST to recognize the hard drive. Currently, all programs you want executed on boot-up (hard drive install, clock, GDOS, etc.) must be in the AUTO folder on the boot-up floppy.

Desk accessories may be either on the floppy or on the hard drive itself. Astra doesn't supply a hard drive auto-boot program, although I don't see why the Supra version wouldn't work. I haven't tested this, however.

The **HD+** is *solid* and built like a tank. The hard drive mechanism is a top-of-

the-line Rodime, and the floppy is of high quality as well—Panasonic or Chinon, depending on the vintage. The front panel features an on/off switch, hard drive light and floppy light. The hard drive unit must be turned on to use the floppy, even if you boot up without the hard drive install program (the hard drive won't be available if you do this, however). The unit is relatively quiet for a hard drive, the low hum of the fan being the most notable

sound.

Once you're tried a hard drive, you'll never go back. The increase in speed is astounding. For example, Word-Writer ST takes close to 30 seconds to load from floppy, but only six seconds from the hard drive. The speed depends on how full the disk is and how much your files are fragmented, but is always significant, nonetheless. You'll have to be careful about buying copy-protected software however, since most of it cannot be installed on a hard drive. I solved this problem by refusing to buy such software. Fortunately, the best packages are not protected.

The **HD+** is relatively expensive, selling for between \$850 and \$950. This is more expensive than the Atari hard drive and a double-sided floppy (SF-314), but there are compensating factors. The first is that the unit is much more compact than two separate pieces of equipment. The second is that SF-314s are scarce, so you may not be able to find one. Finally, the Astra units are top quality.

I'm not saying Atari's equipment is second rate, but my own experience has been disappointing, especially with the disk drives, and I don't like to take chances with my valuable programs and data.

With high speed mechanical equipment (hard drives spin at 3600 rpm), the two old adages, "You get what you pay for" and "Let the buyer beware," are especially true. The Astra **HD+** is fast, well built and extremely reliable. In my book, that makes it a bargain.

TACKLE BOX from SRM Enterprises
P.O. Box 40
USAFA, CO. 80840
(303) 472-6624
Atari 520/1040/Mega ST

by David Plotkin

Tackle Box (TB) "A Utility for Tackling Personal Pascal" is a package of software and information which should be of use to anyone hoping to program their ST. It includes information on the VDI, AES, the operating system, a variety of technical subjects, and most of the hardware chips which make up the ST computers. It comes complete with software libraries (including a math library) which will make programming easier, and it's also probably the most clearly written and least error-ridden source of information about the ST.

Personal Pascal is a product of OSS. The language arrived on the scene approximately a year and a half ago, and is the programming language of choice for many people, especially those who are not technically inclined. The popularity of the language stemmed at least in part because of the better-than-average explanations of how to use the implements of GEM (windows, mouse, menus, dialog boxes, etc.) in your programs. Quite a few GEM commands are directly supported from Personal Pascal. For example, it is, to my knowledge, the only language which allows you to build dialog boxes from within the program itself.

Many GEM and operating system

commands are not present as keywords in the language, and these must be accessed by using generic system calls, such as the VDI_CALL and AES_CALL commands. You must set up your program to use such generic calls by including a variety of custom TYPES in your declarations and also declaring a multitude of EXTERNAL Functions and Procedures to access them. You must also have a source of information on how to use the calls, since the Personal Pascal manual does not provide any information on them.

Enter Tackle Box. This monumental package (it is over 900 pages of information, measuring almost three inches thick) provides just about everything you need to access the functions of the BIOS, XBIOS, GEMDOS, VDI, and AES.

First, we'll talk about the documentation. The first pages detail exactly how to use TB. You have two choices. First, you can use the generic Pascal calls. Certain files from the TB software libraries need to be included when you compile your program, but the libraries provide all the custom TYPES. This option is a little messy since the generic calls are long and full of numbers and arrays which must be initialized. The second choice is to use calls identical to those used in C. These are shorter and more self-explanatory. They are also much cleaner, since you generally don't need to deal with the arrays like INTIN and PTSIN; these are handled automatically by the C calls. Further, you can benefit from the large amount of other literature which has been written about using C in various periodicals and books. To use the C-style calls, you need to include some of the TB libraries as well as link some ".O" files provided on the disk. The source code for these .O files is also provided, so that you can modify them to remove Functions and Procedures you aren't using (this shortens the final object code size).

The documentation continues with sections on GEMDOS, BIOS, XBIOS, VDI and AES. Each section is conveniently tabbed and includes a section table-of-contents to make information easier to find. Sections start out with a well-written introduction explaining what the referenced part of the system software is for and how it operates. Finally, each command is documented on its own page. This includes:

1) Number of the call (for use with generic Pascal commands).

2) KEY—the name to use if you opt for C-style commands.

3) Type of the call (Function or Procedure).

4) If a function, the type of the returned value (integer, etc).

5) Purpose—what the call is used for.

6) Method—An explanation of all parameters which are passed to the call with their names or location in arrays and all parameters which are returned.

7) Restrictions—any warnings about what to look out for when using the call.

8) Implementation—an example of the call.

9) Example—a sample problem and the program code you would use to solve it.

How much would you pay . . .

The documentation is liberally illustrated with samples where words can't handle the descriptions—polymarker types, patterns, etc. Even a helpful picture of the various raster blit modes is included. Further, where a call duplicates a function or procedure already built into Personal Pascal, the manual notes this and recommends you use the built-in version.

But wait, there's more . . .

Then there's the math library. If you've read many Personal Pascal listings, you know that certain useful things are missing. Many trigonometric and transcendental math functions are not provided, nor is there any easy way to convert numbers to strings (for use with DRAWSTRING) and back. The math library section of TB provides all this and more. Sines, cosines, hyperbolic functions, logarithms, powers and decimal degree to DMS conversion are all provided. It has base conversions (binary, octal, hex and decimal) which are a godsend to the programmer. It also has a battery of sound commands. These make using sound fairly simple, since you can directly specify the sound register (there are three available), note, octave and volume, and turn the noise on and off. The math library also includes its own functions to get and set the date and time which are much easier to use than performing bit-twiddling on the numbers you get from GEM. Finally, there are the functions which return the addresses of arrays, screens, and MFDBs. There is even a function to convert a Pascal string to a C-type string, which is necessary to use the C-type calls in the

libraries.

Now how much would you pay?

If you are "into" hardware, the next section of TB will be welcome. It has 78 pages describing the layout and functioning of the hardware, followed by the actual technical manuals of many of the chips, including the 68000, the sound chip, the floppy controller, GLUE, MMU, DMA and SHIFTER. This is pretty technical and is definitely *not* light reading, but having all this information in one place is very useful.

At no extra charge . . .

The final two sections are titled "Pascal tutorial" and "Tackle kit." These consist of a mixed bag of tutorials and sample programs. A complete description of disk layout and file handles works in conjunction with a desk accessory program that makes all kinds of disk operations available from any GEM program. Now you can format disks, copy disks, move files and other disk operations. This is a nice bonus, since you would normally have to pay for a program like this.

Other tutorials include color animation, loading picture files, screen swapping, a section on MIDI, a joystick sampling routine and even a whole section on GDOS. This last is the best description I have seen on the intricacies of this subject. The sample programs include your own copy of GDOS (courtesy of Atari), a GDOS font editor with source files, numerous samples of GDOS fonts, and hints on how to use the font editor to create fonts for GDOS programs like Easy-Draw (from Migraph).

To top it all off, the manual even includes a glossary, reference guide and index.

I have only a few minor complaints about Tackle Box. The manual is peppered with minor typographical errors. These primarily fall into the category of variable declarations which don't match the program examples, and misspellings, which can easily be spotted by even the novice Pascal Programmer. I suppose in a work of this size, a few typos are to be expected! The sample programs are provided on two disks in ARCed form, and you must deARC them, which is something of a pain. And some parts of the manual are difficult to understand because they assume the user knows too much. This is especially true of the highly technical calls, such as reading the intelligent

keyboard.

NOW How much would you pay?

Tackle Box is a remarkably useful work which belongs on the shelf of any programmer, even if they don't use Pascal. It is worth its price just for the clear, concise documentation, with the software libraries being an added bonus for the Pascal user. Captain Rusty Mullins, U.S.A.F., president of SRM enterprises, has really produced a superior product. Once you have seen it, you will agree with me—it's the package Atari should have provided with the ST.

Circuit Maker

Illiad Software
Review by Frank Cohen

It is 4 a.m. Your university level, digital electronic-design class will be holding its final exam in the morning and your term project is due today. You might wonder why your eye's are bloodshot and your fingertips have burn marks. It's probably because you've been up all night working on a "breadboard" to make an integrated circuit do something simple.

Breadboards were created years ago as a temporary way to design and test simple electrical circuits. A breadboard looks like a bunch of Lego building blocks and is covered with places to snap and couple wires: Integrated Circuits (ICs), Light Emitting Diodes (LED), and switches. An electrical engineer usually starts with the design of an electrical circuit on paper. To test the design, the individual components of the circuit are assembled and placed onto the breadboard. A power supply is connected to the breadboard and the circuit is tested. If the circuit fails to work properly, the breadboard components can be easily shuffled around until the circuit begins to work.

Circuit Maker makes the breadboard a thing of beauty. Instead of using physical components, Circuit Maker displays the breadboard and all the possible components on your Atari ST's screen. Only a few clicks of the mouse are needed to make a connection between an IC and an LED. Selecting the RUN command from a drop-down menu simulates the circuit while it is running and the LED will become illuminated. *Voila!* Instant electronics, just add water!

Circuit Maker is a "simulation" pro-

gram. It falls into the category of Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) software systems. CAD/CAM software simulates the physical world as lines and drawings on your ST's screen. Normally, commands are given to a CAD/CAM program using the mouse. The program's ability to accurately manipulate and display the simulation is what determines a good program from a bad program. Circuit Maker simulates most of the commonly used electronic components in a very friendly and intuitive manner.

Circuit Maker is packaged in a simple, unadorned user manual. A registration card is included so that you may receive the planned upgrades directly from the manufacturer. Illiad Software offers one free upgrade to registered users. After that, backups and upgrades cost \$10 each.

The manual covers enough information for you to run the program and understand each of the program's functions, and also includes several experiments in a tutorial section that require a minimal amount of electronics experience. However, for those of you with little or no electronics experience, the manual is not a book on digital electronic's design. For that, you will have to find an introductory book or more likely take a course on the subject.

For experienced electronic designers or students, Circuit Maker is a powerful way to create, test and record digital electronic designs. While a circuit is running, software oscilloscopes may be attached anywhere in the circuit to view logic levels of any component or wire. The dynamic ability to test circuits is very impressive.

The user of Circuit Maker has a large library of electronic components to use. Common NAND and NOR gates, flip-flops, gates, seven segment decoders and displays, LED's, digital timers and oscillators, and switches are available for circuit design. Wires and connect points are used to connect any of the available circuits. Once connected, the RUN command makes the circuit live and any of the output devices (LEDs and Seven Segment Displays) show the performance of the components as the circuit performs its designed task.

Circuit Maker uses GDOS, so what appears on the screen may also be printed out. GDOS is the controversial GEM operating system enhancement that allows fonts and graphics to be shown on

the screen and printed on a laser printer, plotter or normal dot matrix printer.

Illiad Software purchased a limited license to use GDOS from Atari for a \$500 fee. Michael Newson, Illiad Software's Marketing Director, indicated that Circuit Maker uses version 1.8 of GDOS and comes with a printer driver that will print only on Epson graphics printers. Newson said several other printer drivers might become available in the near future.

Circuit Maker was written in the Modula 2 programming language. At times, it can be cumbersome to operate as simple mouse clicks and other graphic functions have long lag times. Circuit Maker is fairly memory efficient, however a one-megabyte system is needed to hold a realistic number of components in a circuit.

Illiad Software also markets a complete CAD package called Athena. Expected early next year, Athena 2 Professional will also have CAM functions to drive tooling and manufacturing machines capable of receiving computer commands. In its utopian state, the designs developed in Circuit Maker will be sent to an "auto routing" program which will determine the exact component layout of your design. Athena 2 will be used to produce the final PC Board design and program a milling machine to produce the finished, printed circuit board.

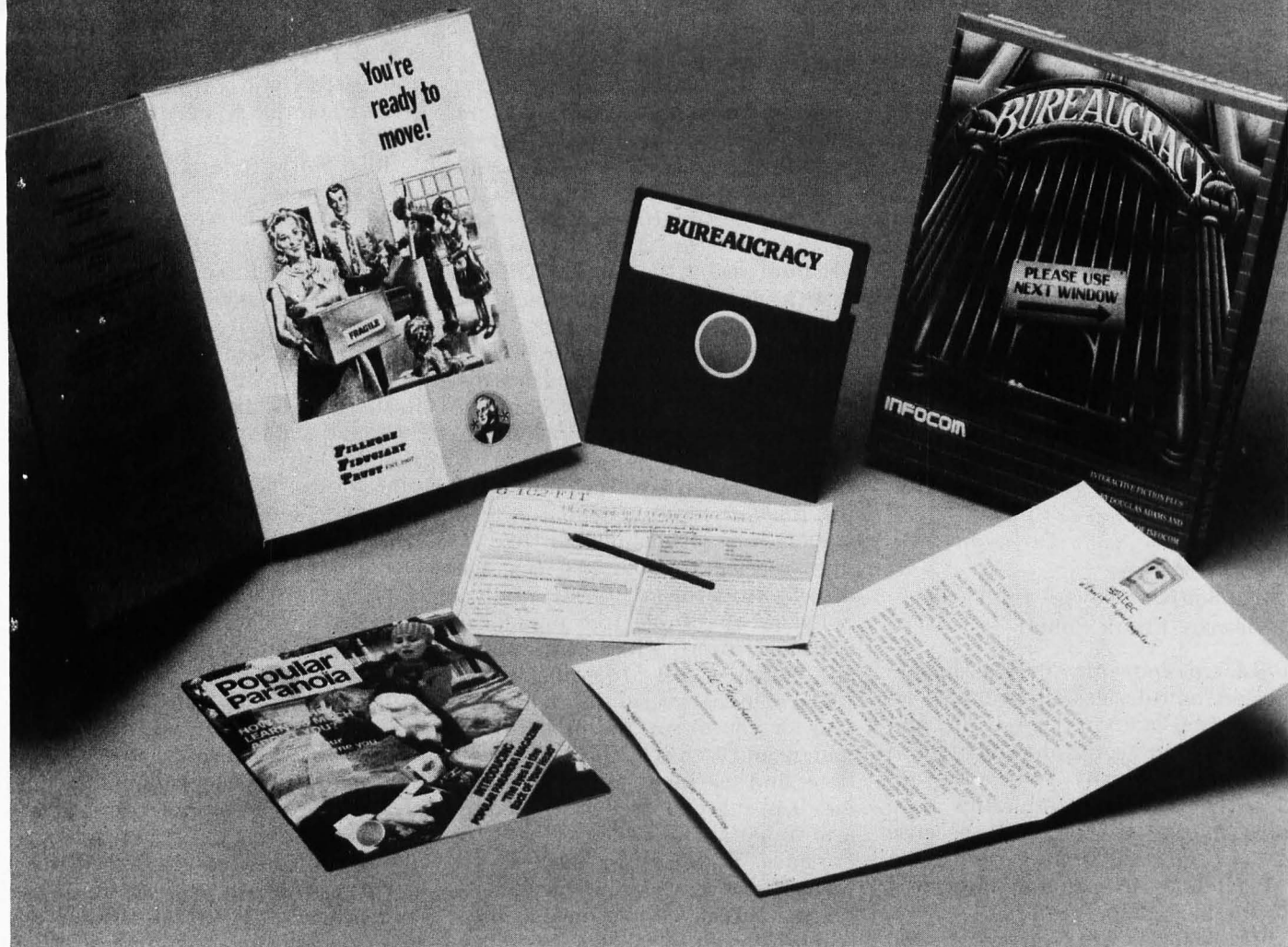
Circuit Maker was designed by Ozzie Boeshans, an electronics designer at Signetics, one of the largest integrated component manufacturers.

Bureaucracy

by Douglas Adams, et al
INFOCOM
125 Cambridge Park Drive
Cambridge, MA 02140
(617)0576-3190
ST Disk \$39.95
by Andy Eddy

Everyone, at one point in their life or another, faces the "red tape shuffle," the hideous condition that comes from dealing with big companies and their infernally inefficient ways of doing things. Proof of this malady shows up regularly; blood-boiling examples include having a phone conversation put on hold for a matter of weeks or getting the runaround while trying to straighten out a \$1 million computer error—obviously not in your favor—on your bank account.

Douglas Adams—best known for



bringing his twisted, yet realistic, sense of humor to light on the radio, on television, and later via computer disk in the *Hitchhiker's Guide To The Galaxy*—and Infocom have taken true-to-life horrors like these and turned them into a warped adventure that is called, fittingly enough, **Bureaucracy**.

The scene is set (based on an actual experience that Adams went through): You have just moved into a new house, as well as acquired a new position in a new company. As if dealing with all that isn't enough, your local bank has misplaced your change-of-address form. So what, you say? Well, the bank has sent your newly renewed, *valid* credit card to your old address and you're supposed to be on your way to a working vacation in Paris. As you quickly find out, nothing seems to go your way. . . . NOTHING! Needless to say, you're up against a wall—a *bare* wall, as the moving company has yet to deliver your furniture from your old residence. See what I mean?

Bureaucracy is much like your standard text adventure; you have to find items and their uses to help you

in your quest, and certain goals have point values. The difference here is that your progress isn't predominantly monitored by the number of moves you make or your score—after all, how could you score in a life experience like this?

Realistically, you'll know how you're faring by way of an on-screen blood-pressure reading. Everything that goes wrong causes your blood pressure to rise: aggravating bank tellers sending you to closed windows, filling out form after confusing form, phoning your girlfriend/boyfriend to find that they're no longer interested in you, even by using words that the game's parser doesn't understand (though the game sarcastically tells you that you aren't licensed or authorized to use that word).

It gets you right in the mood from the start. The first thing you're told when you boot up is that you aren't licensed to operate the software, but through their kindness you can fill out an on-screen form to straighten out the predicament. You're shown a form, and one by one you're prompted to fill in the entries—albeit in a random order that sets you off

balance—and certain slots bring about a snide comment from the game on your answer. Then it proceeds to *mangle* the information anyway; getting your house number wrong, calling you Ms. instead of Mr., and on and on.

Bringing the chase into more realistic ambience, Bureaucracy takes the initial form you filled out and tailors the rest of the contest so it closely parallels your life, creating a type of Computer Mad-Libs, if you will. Your entries become the focal point of the story—the bank is located a couple of doors from your house address, your present and past girl/boyfriend's names are listed in your phone book and they later leave disheartening messages on your answering machine, someone will invariably walk by carrying something shaded in your least favorite color.

Best of all is the whimsically sardonic feel to each and every situation you're up against. Meandering through your neighborhood is like a trip to the looney bin, as you meet the strangest group of people and animals that you could possibly imagine. Door-front intercoms spewing forth the

weirdest banter, a deaf matron with an elephant gun, a one-winged macaw whose cage is lined with something you'd like to get your hands on. And that's just for starters.

You'll laugh and chuckle with everything the game throws your way, but frankly, it gets under your skin after a while because you can relate to what is going on in real terms. This goes beyond the typically wonderful Infocom style, because Bureaucracy uses your name, speaks to you directly and tells you, point-blank, what a mess your life is becoming! Why shouldn't you feel edgy and apprehensive laughing about your life going down the tubes—even if it is a game?

Bureaucracy is a wild romp through corporate affairs and its interaction on the hapless, helpless "little guy." Send a copy of it to a bank manager near you and then go hide your money in your mattress. Besides having a lesson for big companies in there somewhere, it can be entertaining to us peons as well.

SupraModem 2400

Supra Corporation
1133 Commercial Way
Albany, OR 97321
(503) 967-9075
by Andy Eddy
All Atari Computers \$219.95

It wasn't that long ago that we reviewed Supra and QMI's 1200 baud modems. In that article, we noted that modems have really dropped in price as their speed has increased; now that statement seems even more applicable. It appears that 300 baud is now passe, 1200 baud is more or less the new standard and 2400 is on its way to giving 1200 a battle for the top spot.

Helping out in that respect is the inexpensive SupraModem 2400. For Atari owners, there are actually two models. The unit for the ST comes with Supra's own Omega terminal software, and the modem can actually be used on many computers, as the ST utilizes standard IBM cables for hook-up. The 8-bit telecomm crowd gets the same modem, but the package includes the fine, public domain terminal program, Express with a version of the documentation Supra has cooked up).

To connect the modem up to the custom ports on the 8-bit line, Supra also adds in a 13-pin Serial I/O, direct-connect interface. This allows you to laisy-chain it to your other compatible peripherals and avoid the added ex-

pense of another interface unit, such as ICD's P:R: Connection or Atari's 850. For this review, we used the ST version, but the modem itself is exactly the same in both configurations, so no performance differences should be noticed.

The first realization you get after taking the modem out of the box is its diminutive appearance. It measures 1" by 4⁵/₈" by 6¹/₂"—not much bigger than two cigarette packs placed side-by-side—and looks to be about half the size

**Thankfully
missing from
Supra's
docs are
misspellings
and awkward
phrases that
generally
accompany
this Godzilla
school of
technical
writing.**

of most standard modems. Size has nothing to do with ability though, as the SupraModem is still equipped with many of the indicators and functions of other modems: 8 LEDs that update the status of High Speed operation, Auto Answer enabling, Carrier Detect, Off-Hook condition, Data Reception and Transmission, and Terminal and Modem Readiness; an internal speaker; full-featured dialing capability for Touch-Tone and Pulse; user-definable, yet non-volatile, RAM memory for storing of various data such as an often-used phone number, type of dialing, speak-

er volume and more; and most importantly, Hayes-compatibility.

The last feature is one of the most powerful aspects of telecomputing. From that you can set up the performance of the modem for criteria such as how many rings before modem picks up the line, the amount of time it waits for a dial tone before it starts dialing a number, the status of the speaker and its volume, even activation of certain internal diagnostic tests.

These abilities provide you with major league telecomputing muscle, but that also brings with it the potential for confusion and difficulty in learning. Luckily, Supra has coupled the hardware with a decent manual that details how to handle the configuration of the modem. The other benefit of this is the lack of DIP switches anywhere on the modem; it's all taken care of through these internal registers.

This alone puts the SupraModem 2400 head and shoulders above some of the other 2400 modems I've seen recently. The other problem I've had with competing modems, whose manuals usually come from the other side of the world and are then translated into English, is the nearly-unreadable quality of documentation—especially given the frequently intense subject matter. Thankfully missing from Supra's docs are misspellings and awkward phrases that generally accompany this Godzilla school of technical writing.

Actually hooking up the SupraModem is simplicity as well. As tiny as it is, all you have exposed are two modular phone jacks (one for the incoming phone line and the other to connect to a phone), an RS-232 connector passing data between the computer and the modem and a small socket for the power transformer cable. The front panel only contains the eight previously-noted LEDs and a click-on/click-off power switch.

You have to give the Supra folks a lot of credit for as compact and well designed a product as they've come up with—which is their own design, as opposed to their self-labeled Avatex 1200. Modem hook-up is simple, operation takes very little getting used to and the price is an incredible value, a trendsetter, to say the least.

The only hitch I've found is the modem's dropping a connection when the computer is shut off or reset. There are some times that the program you are running may crash, and losing the link-up can be somewhat annoying. Sometimes this brings with it an ER-



ROR message on the next time you try to dial or use any AT (to get the modems ATtention) command. A call to Supra brought a lack of awareness of the problems (though this has been confirmed by one other SupraModem 2400 user, in their defense it could be a problem limited to some of the first units off the production line), but they promised to look into it and get back to me. This follows along with the standard of support they've provided over the years.

All told, the SupraModem is a fantastic deal. For only \$219.95—though I've heard it's been sold in some stores for around \$150—you can't find a more enticing bargain. Once again, Supra challenges the marketplace and gives Atari users the next stepping-stone.

Buzzword

by Paul Granchelli, Carl & Warren Strobel, & friends
The Buzzword Game Company
P.O. Box 440747
Aurora, Colo 80044
512K Disk \$39.95

It was almost a year ago that I took a look at the 8-bit version of this program. I found it to be a refreshingly new and original word game, one which could be enjoyed by all members of the family. However, the only

problem was that not every Atari family could get involved. This has been rectified with the arrival of the ST version of this game.

Buzzword is a word game, plain and simple. There are no fancy graphics, no plunges into outer space. After selecting a category (which is chosen from one of the 200 pre-printed cards supplied with the game), players take turns guessing words, given the first letter of the word, the number of letters in the word, and a pool of letters still unused. Points are awarded for correct guesses, deducted for errors. Three levels, from Bright to Gifted, allow all players to compete evenly. But if this were the extent of the complexity of this game, it would quickly be shelved. Fortunately, Buzzword has enough variety to keep your disk drive humming for months to come.

Each category contains up to 50 different answers. Since only nine are used at any one time, chosen randomly with partial dependence on the level of play each time a category is chosen, you can play many times without learning the answers. Each category has been assigned a replayability factor as a guide to the number of times it can safely be repeated. And the categories and answers are numerous and diverse enough to interest and challenge everyone. From parlor games

to candy bars to Indian tribes, the cards test knowledge in hundreds of areas. A special category, Buzzword connection, challenges you to supply words which naturally follow a given word, such as "red." Answers in this case might be "red carpet" (easy) and "red shift" (hard).

Four basic options combine to create 32 different games. First, singles or doubles can compete. Then, hints may be limited to only the first letter or only the length of the answer, or, for the truly brave, no clues whatsoever. The number of letters in the available letter pool can be displayed or hidden, and a time limit to guess a word can be imposed. But even with all these options, game play is simple. After booting up, you simply select a card, a play level, and the desired options, and then you begin typing in words. Your only problem will be misspelled words and typos. Thus it is desirable to place your best typist and speller at the keyboard.

The game has preserved the screen display used in the 8-bit version with little enhancement, other than the fact that the resolution is somewhat greater. The top right corner of the screen contains the current card, displaying the category and clues, if any. Below that is the letter pool and to the upper left is the scoring area. The bottom left of the display holds a typewriter on which you type your answers.

Documentation consists of a game manual, ST loading instructions, and two decks of game cards. The manual's 24 pages are fully indexed and completely describe game play, while handy charts summarize the 32 play options and the 200 categories. The decks of playing cards, however, seemed unnecessary and tended to slow play and make it less challenging. Indeed, in testing both the 8-bit and ST versions we have never used the cards extensively, and they seem inspired by the cards used in Trivial Pursuit. However, for those who like the cards, they are well printed and sturdy and provide helpful clues, and, when used as suggested in the manual, create even more variations on the game.

Overall, Buzzword is an original and fun game. It can be played alone or with an unlimited number of people with equal enjoyment, and every member of the family (ages ten to adult) can compete. Whether considered as an entertaining diversion, or as a vocabulary building tool, all owners of Buzzword will find themselves to be winners.

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**



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 lose 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.



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**



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




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 ship. & hdlg. (US & Canada). Ohio residents add 5 1/2% sales tax. Foreign orders add \$8.00



WordPerfect®
for the Atari ST

Dictionary

THESAURUS

WordPerfect® in Every Way

If you're looking for software that takes full advantage of your Atari's capabilities while providing an extensive range of features, look no further.

WordPerfect offers the power you need with features like Columns, Indexing, Merge, Macros, Speller, and Thesaurus. You'll find them useful for everything from simple memos to complex reports. All features are easily accessed using the Atari mouse and pull-down menus, or WordPerfect's standard keystrokes.

WordPerfect's GEM-based design taps the Atari's resources with ready access to desktop accessories, full color adjustment for color monitors, a definable mouse pointer and cursor, and the ability to move and size up to four windows. And WordPerfect is written in assembly language to take full advantage of the Atari's speed.

WordPerfect Corporation offers Atari users the stability of a proven product, produced by a reliable leader in software manufacturing. With full documentation, toll-free customer support, and free software upgrades, your investment will be profitable for years to come.

Expand your options with WordPerfect—the most powerful word processor you can buy for the Atari ST. For a demonstration, contact your local dealer.

WordPerfect
CORPORATION

1555 N. Technology Way • Orem, UT 84057
Tel: (801) 225-5000 • Telex: 820618 • FAX: (801) 227-4288

WordPerfect is a registered trademark of WordPerfect Corporation. All other products and brand names are registered trademarks or trademarks of their respective companies.

CIRCLE #117 ON READER SERVICE CARD.