ST-LOG

THE ATARI ST MONTHLY MAGAZINE

U.S.A. $3.50
CANADA $4.75

MAY 1987

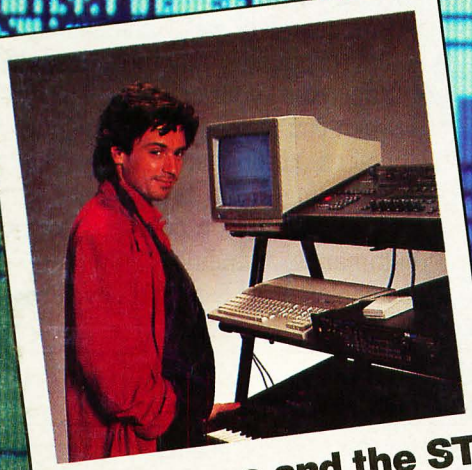
ISSUE 14

MUSIC ISSUE:

Music Steps and Triads

CZ-101 Patch Librarian

PLUS:
ST Clock,
The Making of ST-Talk,
and much more!



Music pros and the ST

0 74470 19938 0 05

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

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

BRING YOUR PROGRAMMING UP TO SPEED.

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

STREAMLINE DEVELOPMENT WITH POWER UTILITIES.

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

NEW VERSION 2.0 FEATURES

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

Sieve benchmark*:
Compile and link time in seconds.

Mark Williams C	46.0
Megamax	78.5

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

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

DEPEND ON A NAME WITH A HISTORY OF PERFORMANCE.

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

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

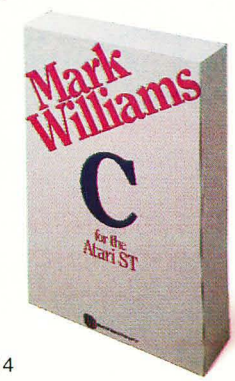
"The all-around best choice for serious software development on the ST."–Douglas Weir of *ANALOG COMPUTING*

GET WHAT YOUR ATARI ST HAS BEEN WAITING FOR.

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

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

*In Illinois call: 312-472-6659



Mark Williams Company

1430 West Wrightwood, Chicago, Illinois 60614

THE ATARI ST MONTHLY MAGAZINE

ST-LOG

ISSUE 14
MAY 1987

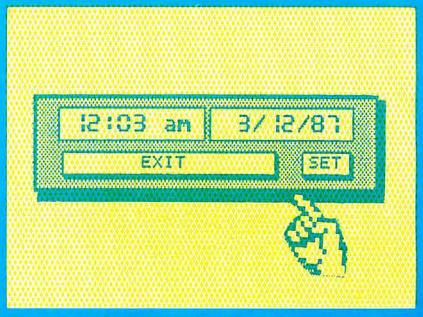
FEATURES

REVIEWS

COLUMNS

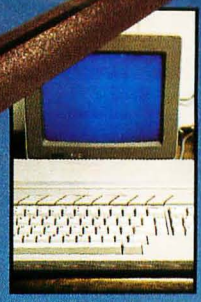
11

17



38

ST-Log (ISSN 0890-9601) is published monthly for $28.00 ($36.00 in Canada, $39.00 foreign) per year by ANALOG 400/800 Corp., 565 Main Street, Cherry Valley, MA 01611. Second class postage paid at Worcester, MA and additional mailing offices. POSTMASTER: Send address changes to **ST-Log**, P.O. Box 625, Holmes, PA 19043.



Make the connection!

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

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

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

Special rates

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

How to connect

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

THE #1 MAGAZINE FOR ATARI® COMPUTER OWNERS

ANALOG COMPUTING

P.O. BOX 23, WORCESTER, MA 01603

MOVING?

DON'T MISS A SINGLE ISSUE

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

DO YOU HAVE A QUESTION ABOUT YOUR SUBSCRIPTION?

Name _____

Street Address _____

City _____ State ____ Zip ____

Check the appropriate boxes below:

☐ New subscription. Please allow 4 to 8 weeks for your first copy to be mailed.

☐ Renewal subscription. Please include a current address label to insure prompt and proper extension.

☐ 1 year — $28.00. This rate limited to the U.S. and its possessions.

☐ Payment enclosed. ☐ Bill me.

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

ATTACH LABEL HERE

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

ST-Log Staff

Editors/Publishers: Michael J. DesChenes, Lee H. Pappas. *Managing Editor:* Diane L. Gaw. *East Coast Editor:* Arthur Leyenberger. *Midwest Editor:* Matthew J.W. Ratcliff. *Contributing Editors:* Charles F. Johnson, Arnie Katz, Bill Kunkel, Maurice Molyneaux, Steve Panak, D.F. Scott, Joyce Worley. *Cover computer graphics:* André Molyneux. *Technical Division:* Charles Bachand, Clayton Walnum, Douglas Weir. *Production Editor:* Kathy Wiesner. *Production:* Connie Moore, Lisa Openshaw, Edythe Stoddard. *Advertising Manager:* Michael J. DesChenes. *Accounting/Circulation:* Robin Levitsky. *Production/Distribution:* Lorell Press, Inc. *Contributors:* John Brochu, Malcolm Cecil, Andy Eddy, Deborah Elder, Frank Foster, Don Grusin, Peter Naleszkiewicz, Regena, Kirk Stover.

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

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

Where to write

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

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

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

Advertising Sales

Permissions

Subscriptions

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

Authors

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

Editorial

In keeping with our musical theme this month, we thought it would be appropriate to ask someone who's been deeply involved with the music industry to step into this month's editorial spot. Don Grusin—musician, producer, composer—fits the bill perfectly. Don's latest solo album is titled Ten K - LA *and is available on JVC compact disks. He was the producer of Ernie Watt's grammy award winning* Sanctuary *album from Warner Brothers, and, outside the record world, was co-producer with his brother, well-known jazz musician Dave Grusin, on the film* Lucas, *directed by David Seltzer. Don tells us he currently lives near Venice, California, and runs every day on the beach.*

Traveling last summer with a group of my favorite musicians, I chanced to bring up the inevitable topic. . ."So, what do you think about the computer explosion?" and, "Do you guys use this technology for your music?" Tim Landers, the fine bassist, was quick to direct me to the world of Atari. At first I was pretty skeptical, having played around with some early versions of video games. I was soon to be pleasantly surprised.

For a year or so I used the Yamaha QX1 for my own composing and sequencing, and, although I had asked them to update some features of that device, I knew I'd eventually have to come to grips with the 20th century and the rise of computer technology—if only to keep pace with my four-year-old daughter, who seems genetically and immediately suited to using the mouse. . .her **DEGAS** creations rival anybody's, I would guess. And so, with able guidance from Jeff Fair, head of R&D at Hybrid Arts, and prompted by my associates, I took on the task of climbing the steepest learning curve I've encountered since graduate school.

Voilá! Within a few days I had mastered the simple stuff, and not long after I was cranking out musical ideas and songs. The combination of MIDI and **Midi-Track ST** from the guys at Hybrid Arts has spawned a technology which practically evolves on its own and, to my delight, actually makes room for more relaxed creative combustion than I would have ever imagined! Although I'm still fairly green around computers and programming, I have found the ST to make composing, sequencing, arranging and recording more simple, easier to initiate and organize (using library files), and it has augmented my experimental work to levels I wouldn't have conceived of independently.

Of particular use to me is the SMPTE feature associated with the **Midi-Track ST**. At the studio we very coolly write and read SMPTE code using this program, and find it to be as simple as pie.

Recently I've been writing music for video and film, and it's no secret that interfacing a SMPTE-coded video cassette with the ST, then through to the MIDI synthesizers and drum machines, makes the scoring task available to folks who may not even have the slightest lick of music training. Though I would not encourage anyone to delete rigorous music study from their academic plans, it is true that, using the software now available, one with an adventurous imagination can tackle projects thought to be only the playground of seasoned professionals.

We've already witnessed this phenomenon in the record business. While we all will have our subjective judgements regarding the quality of the music and sound produced by neophytes, the movement and the direction of musical art is as volatile and progressive as, perhaps, at any time in history. The combination of the "toys," the imagination and radical curiosity of the users, and the interplay between the music and computer industry has created a new "music-sphere." It's really a case where anybody (well, almost anybody) can play, and everyone can enjoy. And, in the process, we're all contributing to newer and multi-dimensioned learning curves which, given our natural curiosity, we'll challenge and conquer—and then do it all over again!

One last note. While my enthusiasm may gush a bit, I need to point out that limitations do exist and are obvious to users both in and out of the music field. I'd like to mention just two: (1) Breakdown. . .this is when you just about have the song figured out, but you haven't thought to copy the data on another floppy disk, because you don't want to take the time from that all-consuming creative five minutes in a row, and blewey!—downtime. (2) The problem of choicemaking. In the analog world you get maybe one or two chances to correct mistakes in playing, in the feel of the track, in the length and other obvious variables one would fix. In Atari-land, I have spent valuable couch-potato and fly-fishing time *making up my mind* about these things, because it's possible to make everything. . .*perfect*.

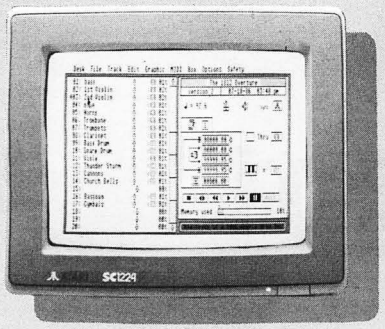
Don Grusin

Tapeless Technology™

Inspired MIDI Software from

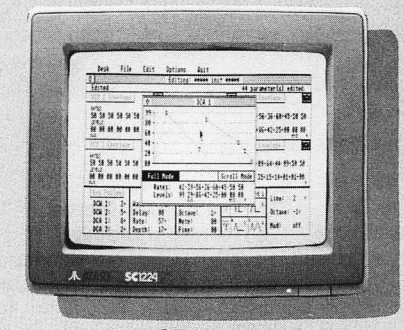
Hybrid Arts, Inc.™

Nobody knows MIDI on the ATARI like Hybrid Arts. We have been the undisputed experts since MIDI started, and now we've got all the bases covered.



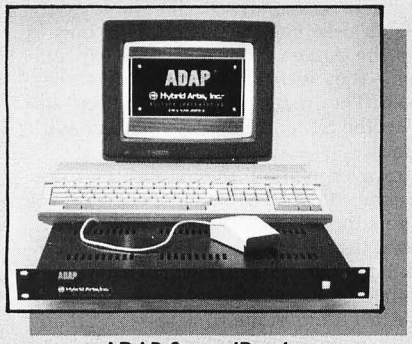
MidiTrack ST

MIDI Recorders: With several different professional quality systems to choose from, Hybrid Arts has a MIDI Recorder for every musician and any budget from hobbist to broadcast professional. Starting with **EZ-Track**, the 20 track polyphonic recorder with editing features like quantizing and instant transpose. **MidiTrack ST** has a full range of advanced editing features including an auto patch injection system and FSK, Roland and TTL sync to tape for studio use. And **MidiTrack Professional** comes complete with our **SMPTEMate** box. SMPTE specs include SMPTE read/write, drop frame, non drop frame, 24, 25, or 30 fps, jam sync, both reading and writing of user bit data. Resolution is at the bit level.



CZ-Android

Patch Librarians: A complete line of products are available for sound patch libraries. **Genpatch** is designed to work with all MIDI keyboards regardless of manufacturer. **DX-Android**, the highly acclaimed editor/librarian program for the Yamaha DX-7 and TX series. This was the first product of its kind to offer both graphic and numeric editing along with the famous android functions that create new musically useful sounds instantaneously. **CZ-Android** is the CASIO verison that works with all of the CZ keyboard series including the new CZ-1 and CZ-230S.



ADAP SoundRack

Sampling and SMPTE hardware: The revolutionary **ADAP SoundRack** (Analog to Digital Audio Processor) takes full advantage of the speed and memory of the ATARI 104OST. The "ADAP" is a hardware/software peripheral for the ST that allows music and broadcast professionals to sample 20 seconds at 44.1 khz with 16 bit resolution. Full MIDI compatability is implemented. System is upgradeable to direct to hard disk stereo digital mastering and SMPTE interlock.

The MidiMate software series for the ATARI 8 bit computers: For the musician on a limited budget but still needing professional quality, Hybrid Arts has a complete line of studio-tested software. **MidiTrack III** is a 16 track MIDI recorder, synchronizer and remote controller for the ATARI130XE. Standard features include 10,500 note capacity, looping, copy and paste editing, programmable tempo, track combining and uncombining. **MMS** is a sophisticated monophonic single step entry sequencer. The operator can enter musical notes from either the computer keyboard or the MIDI synth keyboard. 99 tracks are available to work with. **GenPatch** is a generic patch librarian system for all MIDI keyboards. **DX-Editor** is a numeric editor and librarian for the YAMAHA DX-7 and TX. **Oasis** is our low cost Visual Editing System for the Ensoniq Mirage keyboard sampler. Features include full MASOS implementation, 8 levels of magnification, redrawing of wave forms with high resolution graphics. **MidiCom** is a MIDI telecommunications program designed for easy use.

MidiTrack III

Call or write for a free catalog and newsletter which includes a complete listing of products and pricing.

Hybrid Arts, Inc.™

11920 W. Olympic Blvd., LA, Ca. 90064
Computer BBS: (213) 826-4288 Main Office: (213) 826-3777
Telex: 4931231 (HYBRDUI)

Reader comment

Toward sophisticated ST understanding.

A number of letters in your **Reader comment** section have favored a less technical flavor to your magazine. D.F. Scott, in his March 1987 **Status report**, argued for those who want at least as much, if not more, technical material in each issue.

I strongly support Mr. Scott's view. I am not a computer programmer or technician, and never will be. But I enjoy doing things for myself, and my nature demands that I understand the tools I use. I have dealt with almost every kind of household repair, tune my own car, and so on.

My purchase of an ST was a commitment to more advanced technology and, more especially, to more sophisticated understanding. At first, I found your magazine insufficiently technical. The only reason I bought it was for the Clayton Walnum **C-manship** series, which I continue to find exceptionally helpful. Similarly, I am delighted with the inauguration of Douglas Weir's parallel series on assembly language.

If this kind of article represents a trend for your magazine, then I guess it's time for me to take out a subscription.

Sincerely yours,
Fred Corbett
Etobicoke, Ontario, Canada

We think you—and any like-minded ST users—will favor our reply to the next letter. *—Ed.*

Critique corner.

I read your review of **Skyfox** in the March 1987 issue of **ST-Log**, and was appalled. Steve Panak drooled over the game as if it were **Starglider** or **High Roller**; he said that **Skyfox** was out of this world.

Now, when that game was available, I ran right out and bought it, without even opening it. . .$36 down the drain! **Skyfox** is terrible! The sounds are faithful reproductions of the Apple II's; the graphics were choppy and low quality; and the game has got to be one of the most poorly designed I have ever seen! It's an awful game, the playability is terrible; it's about as fun to play as Synapse's **Dimension X** was a couple of years ago!

I'm sorry to disagree with Mr. Panak, as I usually agree with his reviews, but this time I think he is 180 degrees off. I think you should be careful of printing reviews such as that one, lest you get a reputation for kissing up to software companies to get advertising, something which your principal competitor does all the time.

When I got **Skyfox** and saw what a dud it was, I immediately wrote to Trip Hawkins, the president of Electronic Arts. His response was quick—and disappointing. He was sorry that I didn't like **Skyfox**, and was upset that I wasn't as supportive of his company's problems as, say, a Commodore 64 owner would be (in a similar letter to a friend, he said he thought **Skyfox** "was a great game!")

In response to my question about the possibility of **Archon** or **M.U.L.E.** being made for the ST, he said, "Programmers have to eat, too. It seems that developers feel they can eat better by making a new title for the 64 than by porting an old game to the ST." He also complained that not enough ST users decided to buy **Financial Cookbook** (around 7000 sold on the ST, to around 19000 on the Amiga), and inferred that we had all pirated the program.

Now, I realize that Electronic Arts is having difficulty supporting the ST. Piracy on the ST is a very real problem, just as it was on the old Atari 8-bits. And—when they put all their money on the Commodore Amiga's sweeping the marketplace, only to have the Amiga fall way below expectations—they were understandably embarrassed. They're taking a "wait and see" status on the ST, and I can sympathize with them.

However, while I understand their predicament, I don't see how they can expect me to support them by buying their software if they won't support my machine by making good software. Electronic Arts was not made popular by dropping support of the Atari 8-bits some time ago, and many of the same people who were incensed about their attitude back then are today's ST owners. The only way Electronic Arts can make a good show of their support of the ST is to make software that utilizes the power of the Atari ST, that makes it shine. So get with it, EA!

Before I go, I would just like to thank you for remaining supportive of the Atari world, both 8-bit and ST. I have been with your magazines since **ANALOG Computing**'s issue 7 (I was interested in Tom Hudson's **Graphic Violence!** program in issue 8), and have always been pleased with you. I hope you start support of GFA BASIC soon, as I am really getting into that language, and would like to see some programs written in GFA BASIC.

Sincerely,
Peter Payne
San Diego, CA

*First, we appreciate your opinion on **Skyfox**, and remind our readers the re-*

MOVING SALE SPECIAL #1

ATARI 520 SYSTEM PACKAGE

INCLUDING RGB OR MONOCHROME MONITOR, MOUSE, DISK DRIVE, BASIC, TOS ON ROM, AND RF MODULATOR
FULL MAUNFACTURER'S WARRANTY APPLIES
CALL FOR LATEST PRICE

COMPUTABILITY MOVING SALE

MOVING SALE SPECIAL #2

ATARI 1040ST SYSTEM PACKAGE

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

COMPUTABILITY MOVING SALE

TIMEWORKS

Wordwriter ST . . . . 48.95 Data Manager ST . . 48.95
Swiftcalc ST . . . . . 48.95 Sylvia Porter ST 48.95

FOR ATARI 520 ST 1040 ST COMPUTERS

MOVING SALE SPECIAL #3

ATARI 20 MEG SH 204 HARD DRIVE **Call**

MOVING SALE SPECIAL #4

ATARI SF 314 DISK DRIVE **Call**
DOUBLE SIDED / 1 MEGABYTE STORAGE

WE ARE MOVING TO LARGER AND EXPANDED OFFICE AND WAREHOUSE FACILITIES TO SERVICE YOU BETTER

MOVING SALE SPECIAL #5

ST MODEM PACKAGE **$149**

*AVATEX 1200HC MODEM *ST MODEM CABLE *ST TALK SOFTWARE

MOVING SALE SPECIAL #6

OKIMATE 20 COLOR PRINTER **$199**
& ST PLUG N' PRINT **FREE SHIPPING** IN CONTINENTAL USA

AT LOWER PRICES

MOVING SALE SPECIAL #7

EPSON PRINTERS

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

STAR MICRONICS

NX-10	Call	NX-15	329
NL-10	Call	ND-15	449
NR-10	179	NR-15	539
Powertype			189
NB 24/15 (24 Wire)			Call

WE ARE MOVING TO LARGER AND EXPANDED OFFICE AND WAREHOUSE FACILITIES TO SERVICE YOU BETTER

MOVING SALE SPECIAL #8

PANASONIC PRINTERS		OKIDATA PRINTERS	
Panasonic 1080i	199	Okidata 120	215
Panasonic 1091i	259	Okidata 182	249
Panasonic 1092i	349	Okidata 192+	369
Panasonic 3131	279	Okidata 193+	559
Panasonic 3151	409	Okidata 292E	449
Panasonic 1592	419	Okidata 293E	599
Panasonic 1595	539		

COMPUTABILITY MOVING SALE

NEW LOWER PRICES

ATARI 520 ST AND 1040 ST COMPUTERS

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

ST GRAPHICS
Degas Elite	48.95
Easy Draw	48.95
Cad 3-D	36.95
Graphic Artist	124.95
Paintworks	44.95
1st Cadd	31.95
Aegis Animator	48.95
Neochrome	34.95

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

ST WORD PROCESSORS
Regent Word II	57.95
Wordwriter ST	48.95
Thunder	24.95
Paperclip Elite	Call

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

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

ST ACCESSORIES
Flip n' File II-Micro	19.95
Dustcovers	Call
3.5 Disk Drive Clean Kit	Call
Mouse Pad	8.95
Mouse House	6.95
RCA 6-Way Noise/Surge Prot.	34.95
Anti-Glare Screen	19.95

ST UTILITIES
Music Studio	39.95
Time Link	31.95
Middiplay	32.95
Micro Cookbook	32.95
CZ Droid	69.95
EZ Track	39.95
Write 90	18.95
Fast	31.95
Labelmaster	24.95
Video Wizard	39.95
Desk Cart	62.95

DESKTOP PUBLISHING
Drawrite	99.95
Publishing Partner	99.95

ST DATABASES
DB Man	96.95
Regent Base	57.95
Data Manager ST	48.95
Zoomracks II	96.95

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

ST LANGUAGES
Personal Pascal	49.95
Mark Williams C	114.95
Metacomco Pascal	69.95
Macroassembler	49.95
Lattice C	99.95
Cambridge Lisp	139.95
Modula II	49.95
Metacommco Make	49.95
Modula II Developer's	Call
Fast Basic	89.95
LDW Basic	44.95

HIPPOPOTAMUS
Call for items and prices

ST BUSINESS
VIP Professional	Call
Swiftcalc ST	48.95
Isgur Portfolio	124.95
Financial Cookbook	32.95
DAC Payroll	32.95
DAC Easy Accounting	44.95
Dollars and Sense	64.95
Home Accountant	31.95
Sylvia Porter	48.95
Swiftax '86	48.95

CASIO KEYBOARDS
CZ 101	Call
CZ 230S	Call

ST Educational
Decimal Dungeon	24.95
Fraction Action	24.95
Kinderama	24.95
Read & Rhyme	24.95
Animal Kingdom	24.95
Speller Bee	31.95
Kid Talk	31.95
Math Talk	31.95
First Shapes	31.95
Winnie the Pooh	16.95

ADVANCED ORDERS

Read-A-Rama	31.95
Aesop Fables	31.95
Magical Myths	31.95
All About America	31.95
Adv. of Sinbad	31.95
Arabian Nights	31.95
Wiztype	25.95
Buzzword	29.95
1st Letters and Words	31.95
Math Talk Fractions	31.95

ST TELECOMM
ST Talk Ver 2.0	18.95
BBIST	31.95
Flash	27.95
I.S Talk	31.95

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

ST ADVENTURES
Hacker	29.95
Hacker II	32.95
Sundog	24.95
Black Cauldron	25.95
Apshai Trilogy	25.95
Universe II	44.95
The Pawn	29.95
Starglider	29.95
Dungeonmaster	24.95
Alternate Reality	31.95
Tass Times	32.95
Mercenary	24.95
Autoduel	32.95
Ogre	25.95
Defender of Crown	32.95
S.D.I.	32.95
Balance of Power	32.95
Ultima III or IV	38.95
Portal	32.95
Kings Quest 1,2 or 3	32.95
Space Quest	32.95
Tracker	29.95
Colonial Conquest	25.95
Roadwar 2000	25.95
Wizard's Crown	25.95
Rings of Zilfin	25.95
Sinbad	32.95
Phantasie 1,2 or 3	25.95

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

EST. 1982

ComputAbility™

P.O. Box 17882, Milwaukee, WI 53217

ORDER LINES OPEN • Mon.-Fri. 11 a.m. — 7 p.m. CST • Sat. 12 p.m. — 5 p.m. CST

To Order Call Toll Free

800-558-0003

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

414-357-8181

TELEX NUMBER 9102406440

ANSWERBACK = COMPUT MILW UQ

No surcharge for MasterCard or Visa

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

views are just that: opinion. We print—and always will—reviews which reflect the writer's honest ideas about a product, without regard to whether or not the manufacturer is an advertiser. And we receive our share of criticism from companies because of this consumer-oriented viewpoint.

Second, EA's purchase of Batteries Included brings up interesting speculation about their position on ST products. Check ***A different kind of computer game*** *on page 75.*

Finally, you'll be happy to hear that, soon, we'll be grouping ***C-manship*** *and* ***Assembly Line*** *with a third column, to make up a "programming" section in* ***ST-Log****. That third column will be a series on—you guessed it—GFA BASIC. —Ed.*

Major Motion hints.

With reference to your excellent review on **Major Motion** by MichTron (in **ST-Log** 8, **ANALOG Computing** 48), may I have the honour and privilege of sharing with all fellow gamers some hints that I have learned?

(1) "Clean Kills" — This trick enables you to get rid of Draconian vehicles immune to your machine-gun fire (except the Evil Twin—later), by gently nudging a corner of their rear until they are pushed to the side of the highway. (Or, you can clean kill by slowing down *in front* of a corner of the vehicle.)

(2) "Evil Twin" — When the Evil Twin appears, slow down at once and stop beside the car, while it's still parked at the roadside. Then, using sideways movement of the mouse, push the Evil Twin right over the side of the road until it is wrecked. Repeat the process when *two* Evil Twins appear. As the Evil Twin always keeps to the front of your car, you can adjust your speed until you're nudging directly on the Evil Twin's back; then move slowly to the side of the road as far as possible without wrecking yourself! (With this technique, *anything* parked on the roadside can be destroyed.)

(3) "Evading the Jet" — When you destroy two or more civilian vehicles, the jet will appear, right? And blow you up, right? Now you can kill all the civilians you want! Wait until the Truck appears, then signal for it to take you in. Do *not* enter the Truck yet when the ramp appears. *Now* destroy civilian cars, *then scramble* to get into the Truck! Alternatively, when you have the Turbo-Burst capability, destroy civilian cars. Now drive at high speed. When the jet appears and fires the missile, activate Turbo-Burst! Another alternative... swerve!

That's all, folks. Happy gaming and good luck.

Yours cordially,
Raymond Foo
Singapore

On Apple compatibility and preview demos.

I have two things I would like to make comments on: First, I have been hearing rumors lately that Apple is loosening its iron grip on the computer market. As evidence, I've even heard that some company has actually put a Mac-compatible computer out on the market. If these rumors are true, how did this other company get around the strict copyrights held by Apple? Does this mean that companies such as Data Pacific will soon be allowed to produce a full-fledged "Mac-cartridge"? Let's hope so!

The second subject I would like to address is old news to be sure, but I just had to put in my two cents: I would like to propose two possible solutions to aid in the prevention of software piracy. First, I would like to ask all those software developers out there to make the instruction manuals for their products available separately from the programs, at a lesser cost. This would allow people to get an idea of what they are going to be buying *before* they shell out the fifty to a hundred bucks for the program.

I would also ask that software developers provide demo disks for, say, ten to twelve dollars. It would help give people an idea of what the program can do, without actually buying a working version of the software. The end result would be something vaguely similar to the **Neo-Chrome** program that was shipped with the STs.

How, you may ask, would this help stop piracy? Many people out there who own Ataris (including myself) rely solely on mail-order catalogs as a source of software. Sad but true, many of us have no choice; Atari is still not recognized widely enough that one can just go into any old bookstore, computer store, or whatever, and expect to find Atari-compatible software. (I will add, however, that the situation is beginning to improve.)

Sincerely,
R.C. Philbrick

The case of the missing key.

Maurice Molyneaux has pointed out to us that, in the "Common Key Functions" chart published in last issue's ***Step 1*** *(page 44), the RETURN key is missing. Those of you using the chart should insert that key between the SPACE BAR and ESCape key blocks, in the left column. Sorry about that, Chief. —Ed.*

The Ultimate MIDI Dream Machine

Never mind the budget —think big!

by Malcolm Cecil

It's hard to believe—MIDI was born only four years ago! So much has happened in that short time. The original idea behind MIDI was that two music synthesizers of different manufacture ought to be able to talk to each other.

The resulting flood of MIDI equipment is enough to boggle the biggest and bravest brain: synthesizers, samplers, drum machines, tape recorders, signal processors, audio mixing boards, digital delays, digital reverberation devices, harmonizers, through boxes, blenders, switcher boxes, synchronizers, sequencers and even MIDI lighting systems. Add a wide range of software that turns home computers into recorders, sequencers, editors, score writers, voice patch librarians, samplers and digital recorders, and we're faced with an array of music-generating equipment at prices ranging from a couple of hundred dollars to over half a million. More and more musicians are setting up their own MIDI recording studios in their homes.

So what is all this MIDI, anyway?

MIDI (an acronym for Musical Instrument Digital Interface) is a real-time music communication network, similar in many ways to any local area network. It has an open-ended architecture allowing more equipment to be added to the net. When a large number of units is networked, transmission delays and contention between controllers become the limiting factors which prevent endless expansion.

The nice thing about a MIDI network is that it can be reconfigured to suit the task being performed.

The simplest setup is a single controller feeding a network of synthesizers—usually "daisy chained" from the controller's MIDI-out port to the first synth's MIDI-in port, then from the latter's through port to the next synth's MIDI-in port, and so on down the chain to all the synths.

Each MIDI device's through port outputs an opto-isolated, and therefore slightly delayed, copy of the signal at its MIDI-in port. With just a few units, delay is negligible. But, as more units are added, the through delay builds up, till it can no longer be tolerated. To eliminate this, a "star" configuration is used—one centralized device feeds all the others simultaneously.

Just suppose for a moment there are no budgetary limitations. . .What would be the *Ultimate MIDI Dream Machine*?

Noise makers.

MIDI's sixteen independent channels allow up to sixteen different musical parts to play at once, so our UMDM (as it will be fondly known) will certainly need a minimum of sixteen synths—and we could double up with two or more synths per channel, or use more than one MIDI network, so somewhere around thirty to forty synths would be nice. Of course, several of those would be "sampling" synths—a Fairlight CMI, an Emulator and a Kurzweil or two would do just fine!

It's always nice to have some of the good old analog, pre-MIDI synthesizers (like the Moog, Arp and Oberheim) around for their unique sounds, but we will need a MIDI to V.C. (Voltage Control) converter for each analog synth we include in our system.

Add a couple of drum machines and that should just about do it for the music-generating devices.

Controllers.

The piano keyboard is by far the most popular form of MIDI controller, although guitar interfaces are available and drum interfaces are quite popular. We would certainly want at least one of each of these. We might even include a xylophone-style mallet MIDI controller. Of course, it goes without saying that we'd want all the usual joysticks, wheels, pedals and a breath controller. And a Fairlight Voice Tracker—which generates MIDI note, velocity and pitch-bend information from a microphone input—is an absolute must.

Another form of controller is the sequencer. MIDI sequencers come in two flavors: hardware and software. With so many synthesizers, it becomes a problem to play them all at once. One solution is to record each musical part on a separate track of an audio recorder, and build up the performance in layers. This can also be done with MIDI sequencers, many of which emulate the multitrack audio tape recorder.

Another method is to have several players record simultaneously. This usually produces a more exciting performance, that isn't as "stiff" as one created by a single performer "overdubbing" one track upon another.

Audio signal processing and mixing.

MIDI reverberation and digital delay are just the tip of the iceberg when it comes to audio signal processing. Yamaha's SPX-90 is a wondrous device, giving a huge va-

riety of effects—and it's MIDI controllable. Allen and Heath, a British company, has introduced a superb MIDI-controlled, 32-input, automated audio mixer which must be included in the UMDM. The Dream Machine must also be able to be synchronized with traditional professional audio, video and film equipment.

Since synthesized sounds are not heard until they come out of a speaker, an accurate monitor system in an acoustically balanced and noise insulated room are required to house the UMDM.

The icing on the cake.

The Dream Machine would not be complete without the top of the line synthesizer, the Synclavier. This amazing instrument can digitally record sixteen tracks of audio in real time, as well as thirty-two channels of MIDI—all in sync with film or video. It is also a full-blown digital synthesizer with amazing capability. Of course, it only costs half a million bucks, but the ultimate never did come cheap!

Ah, what a dream...

...But is it just a dream? In fact, one such Ultimate MIDI Dream Machine does exist. It lives on the Paramount film lot in Hollywood, and was recently inaugurated there as Record Plant Stage L. It's being used extensively for movie scoring.

Stage L was opened in February. It used to be the "Foley" sound stage (the name given to sound effects generation in the film industry). It is now set up exclusively for creating electronic music and sound effects, synchronized to motion pictures and television.

One of the largest MIDI installations in the world, its long list of equipment includes a Synclavier, a Fairlight CMI, a Kurzweil, several Prophets, Yamahas and Rolands, and four computers. Both MIDI (via a novel fiberoptic link) and audio can be hooked in to the automated Solid State Logic mixing console in Stage M's control room. Stage M is the huge music sound stage next door, which can house the Los Angeles Symphony Orchestra and the Boston Pops at the same time. The Stage M recording console is so large that it needs a team of four engineers to run it when a large orchestra is recording live.

The four computers run customized software that not only keeps track of the musical score beat by beat and frame by frame, but also keeps track of administrative and publishing information—like who composed and arranged each segment, and its time and title—usually a big headache to sort out at the end of a major movie project. The software even generates a printed report used by ASCAP (the American Society of Composers, Authors and Publishers) to calculate royalties and license fees.

Of even more importance, it generates SMPTE (Society of Motion Picture and Television Engineers) time code and MIDI timing clocks, acting as a centralized synchronizer for the entire system—including film projectors, video and audio recorders, and MIDI equipment.

A total of five interlinked MIDI workstations are available, as well as the ability to work in real time with anything from a vocalist to a full live orchestra.

Incidentally, even when a traditional orchestra is used, the software keeps track of the tempo and gives cues to the conductor for the timing "hits"—those moments when the music must exactly coincide with an action on-screen.

Even though Stage L has only recently opened, it has already made a major impact on the way movies are scored. Synthesized sound has become the most popular form of music-making in Hollywood today. ///

About the author: Malcolm Cecil studied music at the Guildhall School of Music and Physics at London Polytechnic. He has always pursued a dual career in both music and technology.

From 1957 to 1968, he was one of England's top acoustic bassists, well known as both a jazz and orchestral performer. He was principal bassist with the B.B.C. Radio Orchestra from 1964 to 1967. Mr. Cecil was a radar instructor in the Royal Air Force, and later worked as a consultant to numerous film, television, radio and recording studios.

He came to the U.S. in 1968 and became a recording engineer, producer and synthesizer pioneer, winning a Grammy for the best engineered album of 1973 and for the album of the year. Another Grammy came in 1974 for his production, engineering and synthesizer work with Stevie Wonder on Talking Book, Innervisions *and* Fulfillingness' First Finale.

As if this weren't enough, Mr. Cecil built the world's first (and still the largest) polyphonic synthesizer—T.O.N.T.O. (The Original New Timbral Orchestra, currently being MIDI-ized), and composed, performed and produced three albums of original music still considered classics.

He began programming computers in 1978 and is currently president of Electronic Music Publishing House Inc., a company dedicated to music software development for the ST. He has over twenty gold albums for engineering, production and work with synthesizers, for artists like Joan Baez, Stephen Stills, Minnie Ripperton, Bobby Womack and the Isley Brothers. He's also been involved in electronic film scoring for movies and TV, including the original Star Trek *movie.*

Casio's *CZ-101.*

CZ-101

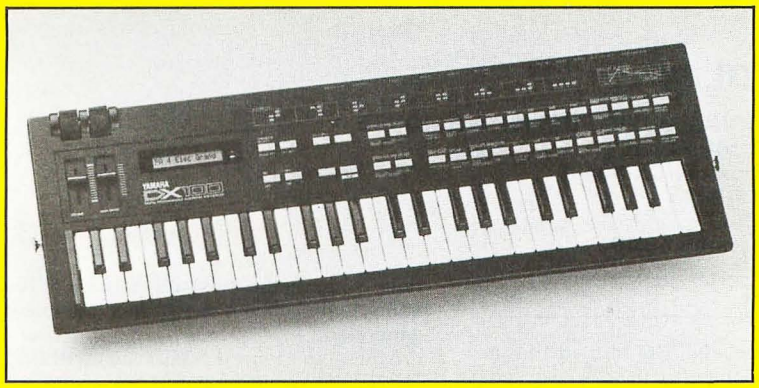
CASIO, INC.
15 Gardner Road
Fairfield, NJ 07006

$499.00

DX-100

YAMAHA INTERNATIONAL CORP.
Digital Instrument Division
P.O. Box 6600
Buena Park, CA 90622
$445.00

by Andy Eddy



Yamaha's *DX-100*

As with all electronic innovations, MIDI has, over its short lifetime, come within reach of the "average" consumer. Aiding this is the Atari ST, which has MIDI ports built in, relieving the user from having to purchase a separate hardware interface.

But, till recently, digital keyboards were quite expensive—that is, until the Casio **CZ-101** digital synthesizer hit the marketplace. To counter, Yamaha, whose **DX-7** has become something of a music industry standard, released their **DX-100**. Many of their features are similar, many are different. One thing is for sure: these are the cheapest MIDI-equipped keyboards you can buy.

Casio's *CZ-101.*

From first glance, you'd think that the **CZ-101** is just another "toy" keyboard from Casio. In fact, it's a scaled-down version of their **CZ-1000**. Its small size does come in handy in some aspects, for all users.

Seeing that, with MIDI, you can remotely run a keyboard as a "slave" from another keyboard—or sequenced from a computer—the forty-nine half-size keys that the **101** is equipped with don't have to become a burden; the internal electronics are all that's required. This explains why some keyboardists are pulling the guts out and rack-mounting them for additional space savings.

Miniaturization is taken to the limit in the **CZ-101**. Its light weight—about 7 pounds—and ability to use battery power make it very portable. The only hindrance is that it doesn't have an internal speaker, but it comes with a line cord for hooking into an external amplifier, as well as a headphone jack for personal listening. AC and car battery adapters are available as an alternative to running the unit on batteries; it lasts about 5 hours in that mode.

As for sounds, the **CZ-101** keyboard comes equipped with thirty-two ready-to-play "patches" (sounds); sixteen are fixed "presets," and sixteen are "internal" (RAM-based) and can be changed by the user. Additionally, there's a slot in the rear of the unit that accepts an optional RAM cartridge for an *added* sixteen sounds. A bank of keys right above the keyboard makes it easy to choose a given patch quickly.

The accuracy of some of the preset sounds, like the electric organ and accordion, show the range of instruments at the user's command. The **101** isn't limited to "standard" sounds, or even tonality. The use of vibrato, white noise mixing and wave distortion in varying degrees gives you the ability to make just about any sound imaginable.

Amply-sized booklets come with the synthesizer (you get two sets: one in English and Spanish, the other in Japanese), to aid in using the various controls, entering prefabricated sounds (a separate manual is provided which has page after page of patches), and, most helpfully, understanding the parameters involved in changing or creating sounds. This is a complex concept, breaking down a noise into components that can be recreated by the internal electronics, and it is well explained.

Yamaha's *DX-100.*

The **DX-100** is very similar to the **CZ-101** in many aspects: it weighs about 6 pounds, has half-sized keys, can be run on AC or battery power (with a claimed duration of 10 hours), has line and headphone outputs (no internal speaker here, either), and is a scaled-down version of a full-featured synthesizer, Yamaha's **DX-27**. From there, the differences are more substantial.

The **100** comes with *192* ready-to-use patches in a preset configuration. Any of these patches, in addition to those that you

may have created, can be put into an "internal" RAM section for instant recall and alteration. This gives you many quality sounds to work with right off the bat; no need to mess around creating one from scratch.

While the **100** doesn't have a cartridge slot like the **CZ-101**, it does have a jack for saving sounds to a cassette "data recorder" (the kind that were used for computer data storage before disk drives were less expensive). This method of saving patches, while a tad more difficult to work with, is much more accessible and affordable to the low-budget player. But it's nice to know that, when you do have to build a sound from the ground up, the **100** is ready and able.

To avoid a severely cluttered input section, Yamaha has given many of the buttons dual duty—much like a scientific calculator's function key. It's this doubling process that allows them to provide so many sounds, as well as sound creation functions. As with the **CZ-101**, the **DX-100** is very simple to program, and the manuals that come with it are equally helpful in reaching that goal. While it may seem, from looking at them, that the booklets will take all your time to digest, they in fact contain versions of the text in three different languages (English, French and German).

At the same time, Yamaha appears to have given the **DX-100** a bit more strength for the user, with regard to professional performance applications. The placement of sockets for foot pedal and breath controller options—the latter alters the synthesizer's output when it is blown into, in the same sense that a horn player can affect his or her instrument's sound with breath control—in addition to a pitch control wheel for note bending (which also appears on the **CZ-101**), and a modulation wheel for selective vibrato and tremolo effect, show a flexibility for enhancing the sound quality. All of this helps keep the **DX-100** from being a rigid-sounding instrument.

Summary.

These keyboards are well within reach of all musicians, rookies and professionals alike. With *careful* searching, these professional-sounding synthesizers can be found in some discount electronics stores for around $200. This is a price that not only entices beginning MIDI users, but attracts as buyers many professional keyboard players, who want them for their smallness combined with their powerful creative capabilities. At that cost, it would take no large outlay of cash to set up a well-equipped MIDI studio to meet your individual needs.

To complement the synthesizers' public appeal, there are quite a few programs available—both on the store shelves and in the public domain—that are devoted specifically to the CZ and DX series of keyboards. These include some powerful sound creation utilities that, given some patience, will let you produce some interesting and musically usable patches. //

Andy Eddy works as a cable TV technician in Connecticut, but has been interested in computers since high school. While his family's Atari 800 is four years old, he's been avidly playing arcade games since ***Space Invaders*** *and is a former record holder on* ***Battlezone****.*

MIDIPLAY

ELECTRONIC MUSIC PUBLISHING HOUSE
2210 Wilshire Blvd.
Santa Monica, CA 90403
(213) 455-2025
$49.95
Demo version $5.00

by Andy Eddy

Keyboard Magazine has termed him a "synthesizer pioneer." He has worked alongside such diversely talented musicians as Herbie Mann, Stan Getz, Stevie Wonder and the Jackson Five. In addition, his engineering talents helped create one of the most intricate synthesizers in the form of TONTO (The Original New Timbral Orchestra). The gentleman whose resume contains all this is Malcolm Cecil.

Now he's turned his attention to the Atari world, focusing primarily on the MIDI capabilities of the ST computers. His Electronic Music Publishing House has released its first product in **MIDIPLAY**, a MIDI sequencer that, when coupled with a MIDI synthesizer, permits you to enter and play back musical selections.

A unique feature of this program, which will benefit those who don't own a keyboard, is the ability to play back music *without* a synthesizer, using the ST's three voices to output the notes via the monitor's internal speaker. Granted, this method is a bit limiting with regard to sound quality, but **MIDIPLAY** allows you to alter the enveloping (attack, decay, sustain and release characteristics) and vibrato of the notes, to make them appear *less* "computer generated."

More experienced MIDI users will find **MIDIPLAY** somewhat restricting for recording, however. For instance, there's no internal metronome to keep track of tempo, so some external means for maintaining a steady, continuous beat is required if serious composing is to take place.

It's doubtful you'd want to compose anything other than basic pieces, due to the fact that **MIDIPLAY** doesn't provide multiple tracks for recording. The strength of MIDI recording is that you can lay down a track, go back to the beginning and add another track to the first without affecting it—in essence, layering the music part by part. This flexibility isn't provided in **MIDIPLAY**, which somewhat hampers composers interested in multi-part creations.

There is also no provision for cut-and-paste editing, though sections can be segmented with markers and those pieces separately saved. Simply put, **MIDIPLAY** is, as it states in the manual, a single-track recorder. Just like a simple tape recorder, it will overwrite the previous data in memory if you re-record on top of it.

On the other hand, **MIDIPLAY** *does* accept data from any of the sixteen MIDI channels available, and retains the other MIDI criteria that a compatible instrument may send—pitch bends, program changes, etc.—and routes them to the instrument upon playback. It's important to note that this data will not be played back if you're using the ST's internal voicing, or using a less feature-laden MIDI keyboard than the one you recorded on, in the same sense that you can't watch a show in color on a black-and-white television set.

The emphasis of **MIDIPLAY** is on its playback capabilities, as the name and manual imply. You can listen to one song by itself, or string together a medley of tunes into memory—totaling about three hours of music for a 1-megabyte ST—which EMPH claims is great for parties. To assist **MIDIPLAY** users in that quest, they're constantly producing music disks that can be purchased separately. Each disk revolves around a specific theme (**Hits of '86**, **Christmas Songs**, **Classics**, etc.), contains around an hour of music—about twelve numbers—and lists for $19.95.

The ability to instruct is something that you won't likely find in other products of this kind. A keyboard is graphically laid out on-screen, with each note in the mu-

sic shown as a highlighted keystrike on the display. This is the case whether you're recording or playing back. Slowing down the tempo to analyze and replicate what's happening on-screen is a good way to get a perception of chording and fingering techniques, something that can take instructors many years to teach you.

The strongest point is that **MIDIPLAY** is quite simple to use; beginners can get the hang of most of the user interface commands right off the screen. Function keys are utilized for most major functions, like *disk accesses* and *start/stop* of playback. These are charted across the top of your display, which eliminates the usual flipping back and forth through the manual.

Other processes (like envelope, tempo and key changes, for example) require a little reading to find the correct keystrokes, but these are few and can be memorized in no time.

The main question in all this: is **MIDIPLAY** for you? If you're a professional musician, the answer is probably no. There are a great many programs out, or coming out, that take full command of what the MIDI interface offers—multi-track, multi-channel recording and playback, word-processor-like editing, internal time-clocking, etc.

On the other hand, perhaps you're a fledgling keyboardist who wants to understand more of MIDI's complexity. Maybe you'd like to learn how to play from an instructor who won't smack your hands with a ruler when you slip. In that case, you could do worse than to employ **MIDIPLAY**, combine it with an ST and a MIDI keyboard, and listen carefully. There's hardly a chance that you'll find another teacher who'll double as an entertainer for your next party. ///

*Also, check the comparative review—covering the **Keyboard Controlled Sequencer**, **MidiSoft Studio** and **SMPTE-Track**—on page 47.*

S.D.I.

by Master Designer Software
MINDSCAPE
5746 Corsa Ave., Ste. 215
Westlake Village, CA 93161
Low resolution $49.95

by D.F. Scott

"Interactive fiction" in computer games has thus far been championed by Infocom. They present purposely incomplete scripts, literally, in such a way that the objective is for you to determine how the script is intended to be completed. CinemaWare, with **S.D.I.**, hopes to be the visually-resplendent version of Infocom, by taking the script concept out of its literal sense and putting it into the familiar world of arcade action and splashy graphics.

Here's the **S.D.I.** story line: the KGB has had enough of peace; so, just to show the U.S. that they're really a bunch of mean, nasty characters, they decide to launch a "limited missile strike." First, though, in order to get through to the targeted U.S. cities, they must disable the **S.D.I.** (Strategic Defense Initiative) satellites.

So you, as Capt. McCormick (if you're female, you'll just have to play a guy), receive a Red Alpha Priority One message from Washington—as opposed to the Blue Beta Priority Two message from the Poconos—stating that the KGB has launched an invasion fleet of XB3 shuttles into space (besides borrowing our technology, they borrowed our alphabet). You are to destroy them—and just guess how you do that. Go ahead, put the magazine down; I'll wait.

You guessed correctly, as usual: you shoot them and they blow up. To be more specific, you'll see "yourself" running through the airlock (not at warp speed, mind you), looking like a cross between *Tron* and the cable TV installation man, accompanied by fanfare that sounds like it was taped from "The Price is Right." Then you see the first-person view from your fighter ship's cockpit (a blue cardboard box), and it's off to fight perhaps the most redundantly boring battle in modern space games.

The first thing you might notice is that, even though your torpedo fire appears to miss, the little U-shaped XB3s blow up much of the time, anyway. The second thing you *will* notice is that *every* time you're hit by enemy fire, you see a close-up portrait of "yourself" in the cockpit, staring into space like Mr. Bill contemplating the steamroller Mr. Sluggo is driving toward his Play-Doh hide.

Another one of your jobs in space is satellite repair. Upon first hearing this, I imagined pictures of the repair mission the Space Shuttle crew undertook in '85, with the lone astronaut in his tetherless EVA suit stopping the satellite's rotation with one hand.

Apparently, the **S.D.I.** designers' idea of technological wizardry is the remote control box on their VCR. The entire satellite-repair sequence consists of pressing the *R* key, centering the bit-mapped satellite image in the middle of a green rectangle, and pressing the joystick trigger.

Were this an actual game, in which your choice of strategies were arbitrary, logic might dictate (as Spock would say) that the best strategy would be to attack the source of the XB3s: the Lenin Defense Station in orbit over the Soviet Union. There are several reasons why you would not want to do that right away, among them: first, it would end the game too soon; second, you'd "lose" if you attacked at the wrong time, since your girlfriend—who, by matter of convenience, is also the Lenin Station commander—has barricaded herself within the station—and if Lenin goes, she goes with it.

There are two other game sequences in **S.D.I.** One is the missile-defense sequence, which begins with a two-minute warning from the KGB, which you receive while out in space blowing things up. In order to dock with your defense station (as well as with Lenin Station, later), you have to engage the docking procedure, which begins like the repair procedure but with the *D* key. You center your station—a large, grey Tylenol tablet—within the green rectangle.

Thus begins the "Computer-Aided Docking Sequence," which assumes the computer that's aiding you is a 1977 Apple II. Alongside graphics in this game—which range in quality from fair to excellent—this part is *atrocious*. You have to fly your ship so that this expanding green rectangle in green space meets flush with the borders of your screen.

The missile-defense sequence itself is a relief to see. The graphics are *fine*, as orange trails of Soviet missiles creep onto the map of the mainland U.S., seeking to construct rather large holes. You have to center your target pointer over the missile you wish to take aim at, and then select the weapon for knocking it down. But if the satellite over the region the missile's in is disabled, you certainly can't fire at it. This sequence is the fun part, and assuredly deserves honorable mention. It requires a high level of concentration, patience over that purposefully slow target pointer, and some fast reflexes.

When it comes time to attack Lenin Station itself, you'll receive a four-minute warning, during which time you have to fly across the Atlantic, "dock" with Lenin, and try to blast your way through the corridor into the control room, where KGB agents have seized your girlfriend. If you

liked the antique slot-machine game where you shoot metal bandits with a B-B gun, then you'll *love* the Lenin Station sequence.

The creative premise behind **S.D.I.** certainly must've been complete and imaginative, but, somewhere during the production of the final product, there evidently was a snag. The three major game sequences seem to have been produced by three different people, with the connecting element between them—the docking sequence—not so much programmed as stumbled upon, or arrived at by default.

In the long run (and even in the short run), a space game with graphics should feature an arbitrary choice of strategies. There is only one way to "score" in **S.D.I.**: by "winning," which is achieved only by having performed, in my opinion, merely adequately, or by the book. With an effort at restructuring the plot concept, interactive movies *can* work, perhaps in future releases. . .or the inevitable sequel. As for **S.D.I.** itself, I have but this suggestion: just say no. //

D.F. Scott is an artist, writer, educator and programmer living in Oklahoma City. He is currently engaged in the study of quantum physics, computing and other ways in which elementary particles interact with each other. Otherwise, he fills infinite pieces of paper.

Flash

by Joe Chiazzese and Alan Page
THE CATALOG
Antic Publishing
524 Second Street
San Francisco, CA 94107
(800) 443-0100, ext. 133
High or medium resolution $39.95

by Matthew J.W. Ratcliff

Flash, a terminal program for the ST, has numerous features. It is fast, smart and flexible. Everything can be accessed from GEM pull-down menus, from a brief command line, or from a keyboard command. This makes the program fairly easy to use for those who are novices with the GEM interface. At the same time, it retains a fast user interface, via keyboard commands, for the more experienced user.

I have very few complaints about this product, except for its manual. While well done, I found it too brief. This would be especially telling for the novice. The documentation does not have an index, but includes a handy quick reference to all the **Flash** commands.

You can begin by making a backup copy of **Flash**. It is not copy protected, so please don't exchange it with friends.

This 100K-plus program loads, then displays a title screen. A click of the left mouse button gets you to the terminal display. This has a status line at the bottom of the screen, which also serves double duty as a command or type-ahead window.

You may press the right mouse button at any time (on- or off-line), to get to the capture buffer. Here, you have the following pull-down menus available: Desk, File, Edit, Upload, Download, Block and Options. If any information is in the capture buffer, it's displayed in the GEM window below. A click of the right mouse button will put you back on-line, with the original display undisturbed.

The capture buffer may be edited from **Flash**. All the cut-and-paste block control functions may be handled with the function keys, or from the block pull-down window. The capture buffer editor is quite simple to use. Though not a full-blown word processor, it's adequate in preparing short documents for uploading to Email or to a message base, minimizing connect time. Blocks of text may be marked and "cut" to a disk file or to the printer, for later reference. With TOS in ROM on a 520ST, you have around 300K of capture space, so you won't be running out anytime soon. You can call up your favorite information service and capture huge amounts of information for off-line perusal, at an hour when the rates are just a bit lower.

A percentage is displayed on the status line at the bottom of the **Flash** screen, indicating how full the buffer is (when it's in use). Other stats on this line include line feed, echo, translation, strip bit, printer spooling, duplex and baud rate. At the bottom right is the current time, which also may be reset and used as an on-line time meter.

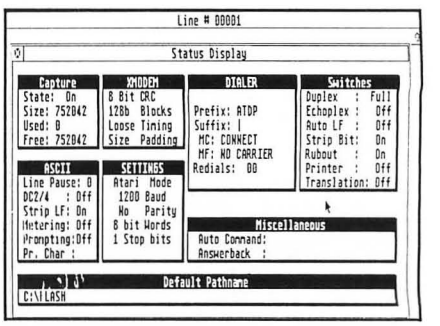
For those with monochrome monitors, **Flash** supports the 48-line text mode, similar to that of **ST Writer**. When in the terminal mode, just press INSERT, followed by the HI ON command, and voilà, you have 48 lines of text on-screen. It takes a little getting used to, but is quite readable and a great way to see large amounts of information at once.

With **Flash**, you have your usual ASCII and Xmodem file transfers, controls for setting up duplex, baud rate and all those other technical goodies. The program supports three terminal modes. The first is ANSI—a VT-100 emulator (a standard, common on VAX systems); next Atari, the default VT-52 mode; and Vidtex, for use with CompuServe graphics. You can read Vidtex or RLE graphics pictures on CompuServe, display them on your screen and save them as **DEGAS**-format pictures (always low resolution for color monitors, and always high-resolution format for monochrome).

The most reliable feature of **Flash** is its Xmodem file transfer—a must for any terminal program, in my book. It seems to support all the different obscure versions of Xmodem floating around these days. You can adjust timing to "loose" for slow-networking systems like CompuServe, for more reliable transfers. You can manually specify the type of Xmodem protocol you'll be using, but why bother? **Flash** can, in most cases, automatically sense what type of protocol is in use and adjust to it automatically.

I have yet to get a bad file transfer with it, or have it do a single "retry" due to a timing glitch, unlike some terminal programs where you must *know* and preset all parameters before attempting a file transfer. **Flash** gets the job done right the first time, minimizing retries and wasted connect time. It's fast, too.

Flash makes it easy to set up a dial directory to call all your favorite boards. A filename may be associated with a particular phone number, separated from the name by the at (@) symbol. This is called a DO file. It may be used to automatically log you onto a system. It's a simple text file,



easily created and saved from the capture buffer editor, made up of text sent to the

computer you're connecting to and commands interpreted by **Flash** itself.

DO files speed up the log-on procedure, though it does take a while to figure out the exact format of all the prompts to set the file up. The documentation for this feature is inadequate for beginners, judging from some of the *frustrated* feedback I've gotten on Delphi.

With **Flash**, you may edit two translation tables, one for input and one for output. For example: on Delphi, the system expects a DELETE character (ASCII 127) and not a BACKSPACE (ASCII 8). You can send the BACKSPACE, but it takes forever to echo. (It goes all the way through the network to Boston, and back. The DELETE is immediately echoed at the local network.)

This has frustrated *most* users new to Delphi. Once you get accustomed to it, it's no real problem, until you go back to your word processor. Then you find yourself pressing DELETE when you meant to press BACKSPACE. With **Flash**, you edit the translation output table, so that every time you press BACKSPACE it sends a DELETE instead. These translation tables can be saved to disk, to be loaded and enabled automatically from your DO files. To return to normal (no translation)—for CompuServe, for example—just put a "translation off" command in your DO file. You could emulate ATASCII mode for 8-bit Atari boards, as well.

Many people have become annoyed with the editor on Delphi, or the poor response when the system is very busy, or the BACKSPACE delays. An easy way to take care of all that is to use your "Type Ahead" buffer. While on-line, press INSERT and enter the TA command. Your status line becomes a one-line typing buffer. It's great for entering messages (you can use left and right arrow keys, INSERT and DELETE characters—all before pressing RETURN). Of course, you could compose a very neat message in your capture buffer editor and do an ASCII upload, if you like.

The Type Ahead is ideal for conference mode, or for "chat" mode on a BBS. While you're frantically typing away on the TA line, any information coming over the modem is echoed and scrolled on the screen directly above. Your typing won't get scrambled by incoming messages.

You can customize all ten function keys, too. You can have one definition, or a different set for each system you call. These can be loaded from your DO files. When hacking away at the message bases, I tend to make more than my share of typing mistakes. Just to be sure my name is spelled correctly, I have that defined under the F1 key. These controls can also include **Flash** commands.

I have yet to find a bug in **Flash**, and there are only a few things that could use improvement. The manual, as above, needs more tutorial information for the novice. When you're looking at the dial directory, you may change the ATP to ATT (pulse to tone) dial command. However, if you save your dial directory, it doesn't get stored. You must save your current "configuration" to save the ATT command. It's in the documentation somewhere, but is a bit confusing.

You cannot format a disk from **Flash**. So if you're on-line and run out of disk space during a download, you'll just have to delete some files or call back later. You may press ALTERNATE-1 through 4 to get a directory and a listing of free disk space on drives A through D. That should have been taken a little further for those who have hard drives, with multiple partitions (e.g., E, F, and so on—up to P, potentially). And—I know I'm getting picky here—the mouse pointer doesn't shut off while editing the capture buffer, a minor annoyance.

Flash is the fastest, most complete, most reliable terminal program I've used to date. Except for the minor updates noted above, about the only things they could throw into this program (besides the kitchen sink) would be a spelling checker and a "learn" function, to figure out your DO files for you. I would recommend **Flash**, especially if you're heavily into telecommunications and have been looking for something powerful. //

True Basic Language System

Over twenty years ago, John Kemeny and Tom Kurtz invented BASIC. Today, more people program with BASIC than with all other languages combined.

Kemeny and Kurtz call their latest version True BASIC. It's easy to use, but has the very latest in structured language design. Named subroutines and advanced control structures make line numbers a thing of the past. And now, with the addition of modules and workspaces, some of the most advanced ideas in programming are part of the language that everyone can understand.

How do I solve this quadratic equation? What does y = sin(x^3) look like? What's the rule behind the Fibonacci series ?

Explore topics like these without getting bogged down in the mechanics. Each package in the Kemeny/Kurtz Math Series is a flexible, menu-driven toolkit that illustrates, as graphically as possible, the major concepts covered in most math courses.

The eight packages in the Series range from Algebra I to Probability, Discrete Math and Statistics.

We're putting our best feet forward.

Hundreds of schools teach with True BASIC, because it's easier to learn than Pascal or unstructured BASIC.

And thousands of professionals and hobbyists have switched to True BASIC for their programming. Maybe for its terrific graphics. Its number-crunching strength, with support for large arrays and matrix algebra. For its speed, and for an editor that makes programming a breeze. And programs are portable between the IBM PC and compatibles, Macintosh, Amiga and Atari ST.

Suggested retail price $99. Libraries $49.95 each.

Teachers use them in the classroom. Students try their own examples. At some schools, Kemeny/Kurtz packages provide the essential "laboratory equipment" for independent study projects. These are a lot more than the same old "drill and tutorial" programs!

There's no programming required, no special syntax for you or your students to learn. Each package is a stand-alone application that can be used with any text, and independent of the rest of the Series. No BASIC or other software is required.

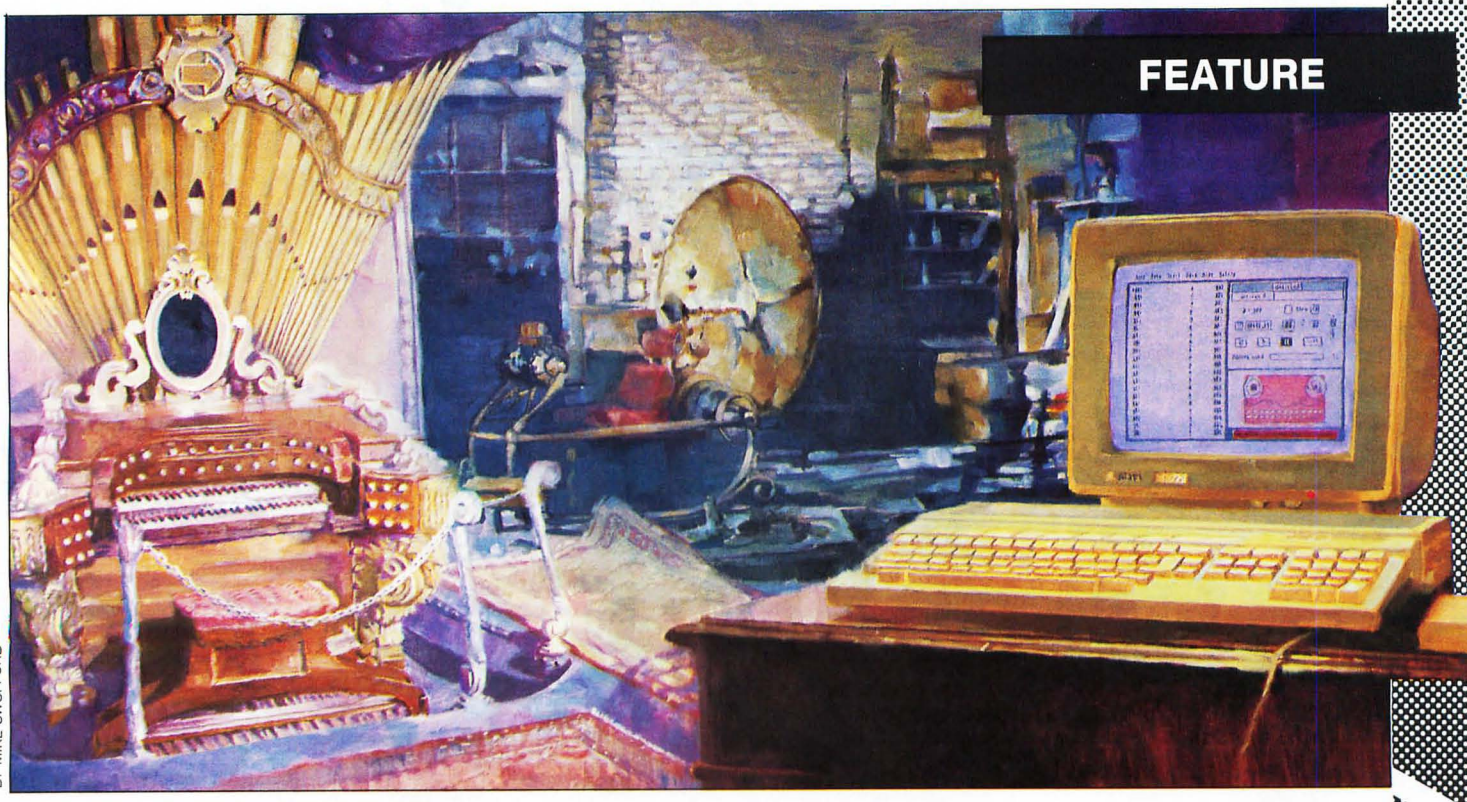
Suggested retail price $49.95 each.

You know us as the inventors of BASIC. As developers of great software on other machines. Now, we've done some of our best work yet for the Atari ST.

For orders or more information, call us at 1-800-TRBASIC. Because for the ST, you'll want to know us better.

Kemeny / Kurtz Math Series

True BASIC, Inc.
39 South Main Street
Hanover, NH 03755 (603) 643-3882



BY MIKE SWOFFORD

Professional Musicians and the Atari ST

The inside story of how Atari won the war against IBM, Apple and Commodore.

by Frank Foster

Most MIDI experts will admit that the Atari ST pretty much took the industry by surprise. . .especially since the change happened so quickly! Now there is widespread agreement that the Atari ST is becoming the international professional standard for the music Industry.

It was two days after the Grammys. . . Peter Gabriel was meeting with me to discuss an ST MIDI system for his new studio. I wondered why someone like Peter Gabriel would be thinking about an ST when he already owned a Fairlight and could probably afford any computer system he wanted. He listened quietly, taping our conversation with a small portable recorder. I explained that the ST was the continuation of the revolution that MIDI started. This technology—previously out of reach, financially, to the average musician—would now be accessible to even the "potential musicians," people who, with the aid of a computer, might find new musical abilities they had never been aware of.

He smiled, and I realized it was probably the politics of MIDI, this new affordable access to technology, which attracted his attention to the ST. That, plus the freedom these recent advances now offer to the artist. Applications like the **Android** programs, capable of generating hundreds of new sounds for the musician to choose from, seemed to interest him the most.

"Whether it's an album or video project, I like to work from as many options as possible," he said. After the meeting was over, I watched him walk toward the parking lot with an armload of ST software, and I couldn't help but think that the Atari ST had finally made it. The competition would not be laughing any more; *1987 would be the year of the ST.*

From our point of view, the war started in June of 1983. At that time, Hybrid Arts was basically two people. This was about six months before the MIDI spec was accepted as the standard. We shared a booth with the IMA (International MIDI Association) at the NAMM (National Association of Music Merchants) convention and showed the first MIDI sequencer. They saw the Atari screen and started laughing, apparently thinking it was some kind of video game. Then Bob Moore hit the space bar and, to their surprise, music filled the place. The crowd stared in disbelief at the Atari 800 playing back through a Prophet 600. MIDI was virtually unheard of at this stage, and the Prophet was the first synth to be interfaced. We picked the Atari because the custom LSI chips made it much more suitable for real-time MIDI processing than the Apple or Commodore. Ironically, the very engineering that made the computer great for fast-action arcade-style games also made it perfect for time-critical applications like MIDI sequencing.

In 1984, Hybrid Arts released **MidiTrack II** and, by now, we had growing competition. Passport Designs had several programs for the Apple II, and Sequential Circuits had a system for the Commodore C-64. That same year Roland released a $350 MIDI interface for the IBM. But it was the C-64 that captured most of the market. Companies like Dr. T, MusicData, Syntech and Cherry Lane Technologies jumped into the market with a variety of good software. This gave customers a real choice in programs for the C-64.

When Apple introduced the Mac, it took almost two years for MIDI software to appear. But when it did, in late 1985, several new companies popped up. First came Opcode with a sequencer and patch librarian, followed by Southworth with Total

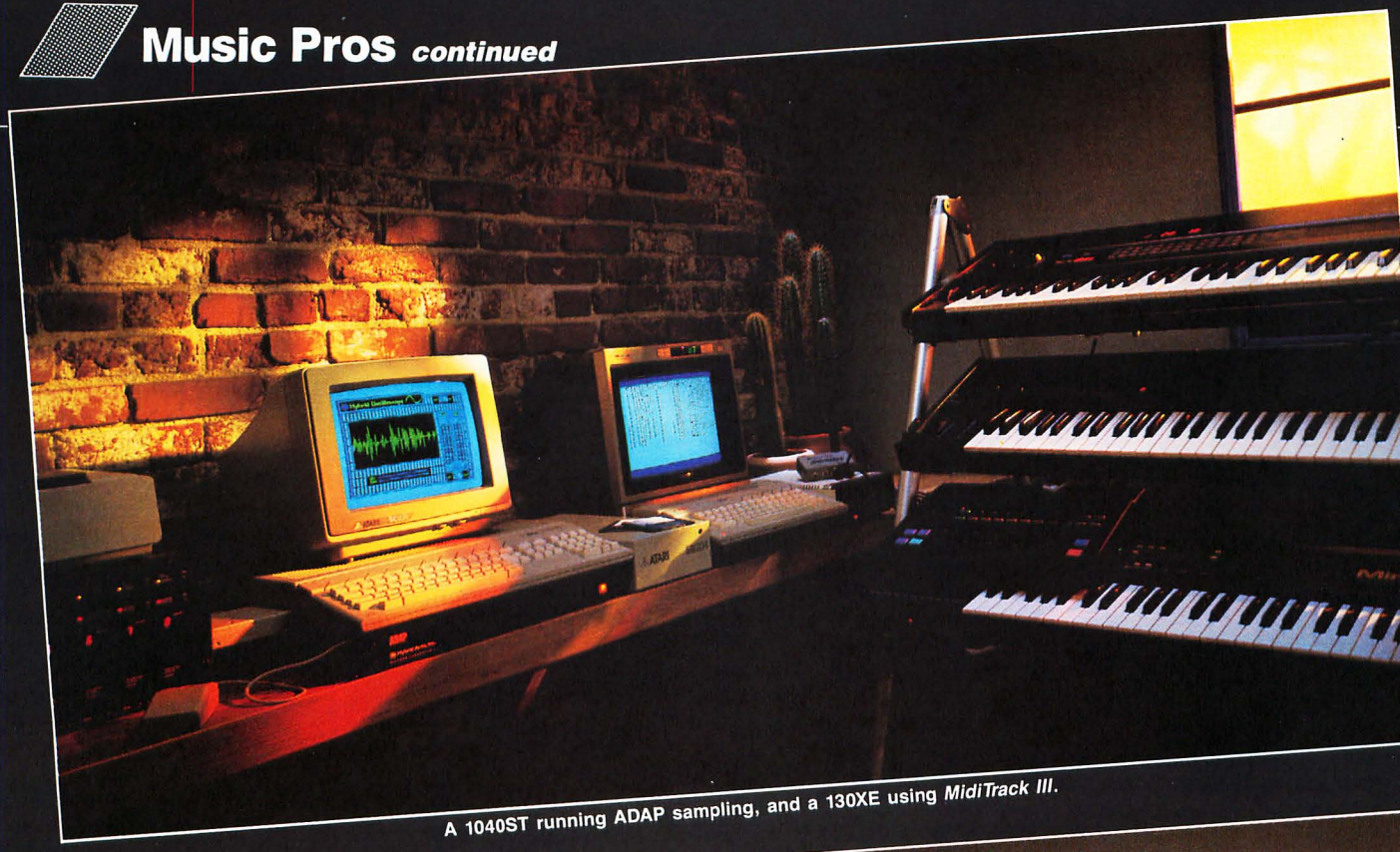


PHOTO BY DAVID SESSIONS

A 1040ST running ADAP sampling, and a 130XE using *MidiTrack III*.

PHOTO BY FRANK FOSTER

Hybrid Arts founder Bob Moore demonstrates the ST to Stevie Wonder.

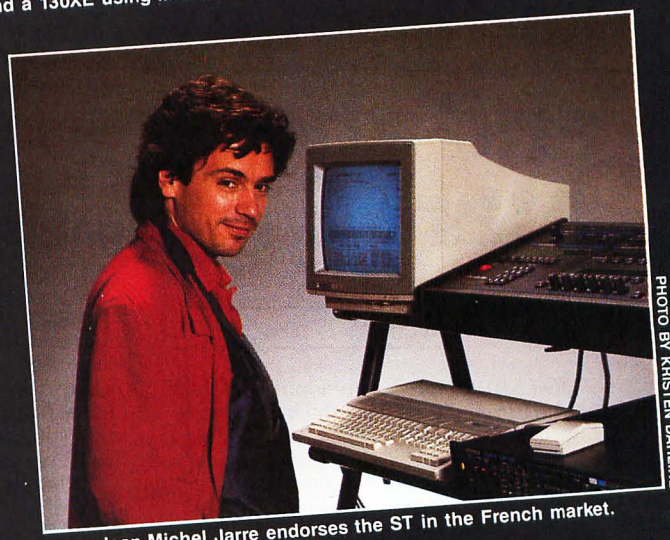


PHOTO BY KRISTEN DAHLINE

Jean-Michel Jarre endorses the ST in the French market.

PHOTO BY DAVID SESSIONS

Don Grusin (left) and Jeff Fair working on the MGM project. The Atari is locked into video with the SMPTE timecode.

Music, and Mark of the Unicorn, when they introduced Professional Composer (a sheet music program) and later addressed MIDI with Professional Performer (their sequencing system).

The Tramiels had taken over Atari in 1985 and given us the 130XE. We released **MidiTrack III** to take advantage of the 128K of RAM this computer offered. Nevertheless, it was a difficult time. Though the press acknowledged **MidiTrack III** as one of the most powerful systems on the market, many people were still laughing at us. To avoid embarrassing questions about the "game computer," some pro musicians actually covered their Atari 130XE labels with gaffer's tape before taking the machines into the studio with them.

By 1986, the Mac was the "computer of choice" with professional musicians. They loved the user interface, and MIDI was becoming an accepted thing—in and out of the studio. For many years, studio people claimed that the MIDI spec was too slow for professional use. But the acceptance of MIDI came more slowly than many predicted, and six of the major software companies went out of business.

It was the beginning of the great MIDI software shakeout, and the sudden introduction of the Amiga and the Atari ST added to the confusion. MIDI developers had to decide which machine to support, or which machine *not* to support. In the February 1986 issue of *Keyboard Magazine*, Dave Kusek of Passport Designs stated, "In that Atari's survival is questionable, we've decided not to invest in software development for that machine. There is very little room in the marketplace for a machine with an independent operating system which is not compatible with anything else." In the same issue, Joe West of Computers and Music (a retail store specializing in MIDI software) reported, "Ninety-five percent of our calls are Amiga related. We're not interested in the ST. The fact that a MIDI port is built in is no big deal, since we all know that adding a MIDI port is easy."

Just as the Amiga was getting started, Cherry Lane Technologies (the company commissioned by Commodore to develop MIDI software for the Amiga) went out of business. That left only one company fighting the war for the Amiga. Their concept was MIDI multi-tasking, but the reality is that, when you're doing time-critical software like MIDI sequencing, the last thing you want is an interrupt to update a different application program.

In the April 1987 issue of *Keyboard Magazine*, the facts about the Amiga's operating system were finally made public. Peter Gotcher revealed that ". . . some computers even have inherent timing problems. Take, for example, the Commodore Amiga. I have spoken to several developers writing software for the Amiga who have been unable to use the Amiga's high-speed clock to time-stamp incoming MIDI events (the clock starts and stops unpredictably when it is interrupted by other processing tasks). The only alternative is to use the Amiga's 60Hz video retrace to time events, resulting in a timing accuracy of 16ms. This produces some pretty spastic playback unless you quantize everything."

So, if we can assume that the IBM interface was too expensive and the Amiga has a clock accuracy difficulty, then what makes the ST *really* better for MIDI than the Mac? Could it be because the ST MIDI is built in? Or, by not buying a Mac, the customer could afford to buy a synth with the money saved? Is it because the ST is noticeably faster than the Mac? Well, the ST does have more flexibility in programming formats and styles—TOS, GEM, GDOS, etc. Sure, color could be a factor, but even in black-and-white, the ST has more display capacity. And, of course, if you really have to use some Mac software, you can run it on the ST using one of the Mac emulator systems. But not vice versa. With MIDI, it's really up to the musicians to decide.

The musicians.

THE POINTER SISTERS — The keyboard players for The Pointer Sisters have been the ST's strongest supporters. Whether at an Atari Expo or the Atari booth at CES, Mark Ritter and Greg Whelchel were there educating people about MIDI and the Atari. So it was only natural that the Sisters would pick an Atari system for their own use. But Atari was even more pleased to hear that the Sisters felt strongly enough about the hardware/software combination to endorse it in an ad campaign.

TANGERINE DREAM — No group is more aware of the ST's popularity in Germany than the Tangerine Dream trio. They are currently using six 1040s in their studio setup. Tangerine Dream is respected not only for their album projects, but for their many film scores, as well, including: *Sorcerer, Risky Business* and *Legend*. So, for their film work, the SMPTE/MIDI system went into immediate use.

MIKE PINDER — As the keyboard player of The Moody Blues, Mike was one of the early innovators designing musical arrangements around available technology. He was one of the first to do analog sampling onstage, with the old Mellotron system (or Pindertron, as the upgraded version was known). So, when MIDI started, he was a strong supporter of the Atari 800 system—later the 130XE and, finally, the 1040ST. And, even in the middle of all this technology, Mike's latest solo project still retains a lot of the "old Moodys" feel—perhaps because he uses MIDI as a composing and arranging tool, with the musical ideas taking priority over the technology.

JAY FERGUSON — Jay has a long history in the industry. He comes, originally, from a classical music background, yet he was extremely successful in the pop music field. He was a founding member of the rock group Spirit, and followed that with Jo Jo Gunne. Later, as a solo artist, he made several hits, including "Thunder Island." For the last several years, he has gotten a reputation for motion picture soundtracks, composing songs for movies such as *Perfect* and *The Terminator*. Jay used Hybrid's Atari 8-bit system for three years and recently upgraded to the ST with a MIDI/SMPTE system. He is currently completing the score for a science fiction film called *Pulse*, in 70mm and 6-channel Dolby sound. Jay feels that the Atari MIDI system and his career as a film composer have helped to combine his backgrounds in both classical and pop music into one common medium.

MARK WOLFSON, SHARON BOYLE and GARY GOETZMAN — The team responsible for musical direction in the film *Stop Making Sense* is currently using the Atari for a series of films. The first, titled *And God Created Women*, is written and directed by Roger Vadim and stars Rebecca DeMornay (from *Risky Business*). The Atari ST, by the way, makes an on-screen appearance, beyond its behind-the-scenes role as a MIDI production tool.

THE JAZZ ARTISTS — Lee Ritenour was one of the very first to really get into MIDI from the guitarist's point of view. With his SynthAxe, he simultaneously records MIDI on the Atari and audio on the multi-track tape deck. He later decides how to mix or quantize the digital audio data. Lee's friend, jazz bassist and keyboardist Abraham Laboriel, never got into computers much before the ST, but had *no problem* with the icon-based interface. He was totally into it, composing at an incredible rate. If he had a question, his friend Don Grusin was there to offer advice. It was no surprise then, when Don's brother—internationally respected jazz artist Dave Grusin—selected the ST for his own system. It seems the Atari ST has enjoyed a word-of-mouth, chain-reaction effect in the jazz session player circle.

And these guys are taking it into movie projects with them. This summer, film buffs will notice a new logo sequence at the beginning of each MGM movie. The soundtrack was programmed entirely on an ST, controlling a whole bank of synths and samplers. Don Grusin and Jeff Fair were able to synchronize the ST to the video with SMPTE timecode, providing the client with instant response to each of the director's suggestions. Since the video image was also created on a computer by Digital Productions, this piece really indicates the amount of control attainable with today's technology.

While assessing the Atari's position in today's music market, it's appropriate to give the last word to one of the trade maga-

zines. *Musician Magazine* has the largest audited paid circulation in the music-making industry. In the April 1987 issue, Editor Jock Baird made the following statement: "Actually, the biggest single winner at winter NAMM was probably not a music manufacturer at all, but a computer maker: Jack Tramiel, the Holocaust survivor who turned Atari around. . ."

To take it further, Baird also stated, "If 1986 was dubbed 'the year of sampling,' my call is for '87 to be 'the year of the Atari.' Nice job, Jack."

When all is said and done, the Atari ST has *proven* itself in the music industry—not just in the professional market, but in the consumer market, as well. In fact, the ST is succeeding in taking the promise of MIDI to the people, where it can do the most good, to the average person with a natural curiosity about learning and music. If the ST can make it just a little easier for a person to create beautiful music or to communicate musical ideas, then maybe it has done something really significant, something that sets it apart from other computers. And maybe this will set Atari apart from other computer companies. We think so. ///

Frank Foster is the President of Hybrid Arts Inc., Los Angeles, California.

SCHEDULED ATARI FAIRS

MAY 8 & 9, 1987
DALLAS, TEXAS*
InfoMart
1950 Stemmons Freeway, Dallas, TX
For further info, write to:
P.O. Box 851872
Richardson, TX 75085-1872

MAY 16 & 17, 1987
SEATTLE, WASHINGTON
Seattle Center, FLAG Pavillion, Seattle, WA
For more information, write to:
15959 NE 1st St., Bellevue, WA 98008

UPCOMING SHOWS

Additional information will be published in later issues.

JUNE 1987
ATLANTA, GEORGIA

JUNE 1987
SAN JOSE, CALIFORNIA

JULY 1987
CHICAGO, ILLINOIS

AUGUST 1987
DETROIT, MICHIGAN

SEPTEMBER 1987
GLENDALE, CALIFORNIA

OCTOBER 1987
WORCESTER, MA

NOVEMBER 1987
PALM BEACH, FLORIDA

NOT YET SCHEDULED:

HONOLULU, HAWAII
NEW YORK CITY, NEW YORK
ST. LOUIS, MISSOURI
SAN JOSE, CALIFORNIA

***ANALOG Computing** will be attending these shows

ST news

Easy-Draw companion.

In addition to the updates mentioned last issue, Migraph has just announced a bit-image loader for its **Easy-Draw version 2.0**. The **Easy-Draw Supercharger** allows **Easy-Draw** to load bit-images from **DEGAS** or **Neo-Chrome**, giving users the ability to add clip art into their work. Scanned images and portions of any screen display can also be added in. The **Supercharger** will retail for under $50.

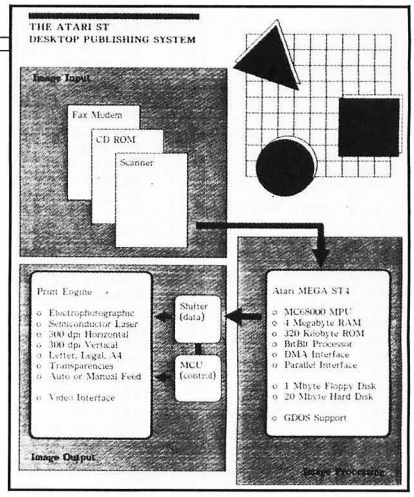
Also available to support **Easy-Draw** owners is **Font Pack 1**, a package containing two fonts: Rocky and HiTech. Another package, **Draw Art 1**, contains over 100 predrawn object-oriented images—vehicles, symbols and borders, to name a few.

A second art package, **Technical Draw Art 1**, is aimed at those who create technical illustrations, floor plans, wiring diagrams, and so on. Included are symbol libraries for electrical schematics, hydraulic and piping layouts, flow charts, wiring diagrams, etc.

Font Pack 1 retails for $39.95 and is available in 9- or 24-pin versions. The two art programs list for $29.95 each.

Additional printer drivers are also available for the HP Laserjet Plus and Series II printers, as well as for 24-pin printers like the Star NB series, NEC P series and Epson LQ series. These go for $19.95 each.

In early summer, Migraph will release a Postscript I driver enabling all **Easy Draw** files to be printed directly on Postscript-compatible printers. For additional information, contact Migraph, Inc. at 720 South 333rd Street, Suite 201, Federal Way, WA 98003 — (206) 838-4677. Reader Service #101.



Powerful desk accessory.

DeskCart! from QMI is a plug-in cartridge containing GEM desk accessories. These utilities can be used in any GEM application, without having to exit the program in which you're working. Because it's a cartridge, less RAM is used up, and the programs are permanently available—since they're in ROM form.

Also contained in the cartridge is a battery-operated clock. The fourteen accessories consist of a calendar, notebook, card file, calculator, typewriter, address book, VT-52 terminal, control panel, screen dump, memory test, RAMdisk control, disk utilities, print spooler and screen dump.

Another plus: GEM only allows you up to six accessories in memory at once. With **DeskCart!**, five accessories can be loaded in addition to the cartridge. A detailed manual explains all of the functions.

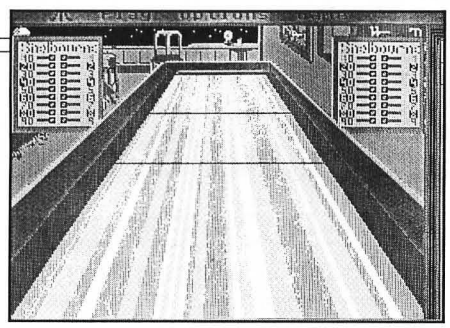
Retail is $99.95. Quantum Microsystems, P.O. Box 179, Liverpool, NY 13088 — (315) 451-7747. Reader Service #102.

ST-Shuffleboard!

This 3-D shuffleboard table simulator mimics the game which can be found in pubs or on many college campuses. Sharp, colorful graphics with superb game play and many options give you a new form of entertainment for your ST. This program is from the same designers as **ST Pool**.

Two views are displayed simultaneously: a 3-D view down the table, as well as an overhead view. Play against the computer or another player. The computer's skill level is adjustable, as are the foul line and mouse speed.

Cost is $29.95, color systems only. Shelborne Software Systems, Inc., 7221 Rising Sun Avenue, Suite 191, Philadelphia, PA 19111 — (215) 884-2656. Reader Service #103.



Electronic drawing boards.

EI/O Products is currently shipping the **ARTablet**—a graphics tablet available in three sizes. The drawing boards, each with an attached stylus, allow anyone to enter data or draw by hand, rather than with a mouse. **The Freelancer**, a 6x9-inch tablet, sells for $395; the **Illustrator**, which is 12x12 inches, goes for $539; and the **Director**, 12x18 inches, lists for $877.

Some of many applications these tablets can be used for include: art design, engineering, architectural and educational environments—and whenever computer graphics are required. In addition, the graphics tablets work with any Atari ST program which utilizes mouse input. No software modifications are needed! Compatible programs include: **DEGAS** and **DEGAS Elite**, **Neo-Chrome**, **Publishing Partner**, **CAD 3D**, **First CAD**, **Aegis Animator**, and many more. Included with the tablet are: a drawing stylus, power supply, RS232 cable for connection to the modem port, a device driver for STs with TOS in ROM, and installation instructions.

The tablets work in monochrome or color. For further information, contact EI/O Products, 1559 Rockville Pike, Rockville, MD 20852, or call Jeffrey Siegel at (301) 869-5984. Reader Service #104

COMPUTER CREATIONS

YOUR ATARI 520ST SUPPORT CENTER

ABACUS	
Paint Pro	33
Text Pro	33
Data Trieve	33
Powerplan	54
PC Board Designer	139
Assem. Pro	43
Paint Pro Data Lib.	21
ACADEMY	
Typing Tutor/ Word Invaders	23
ACCESS	
Leaderboard Golf	26
Tournament Disk	14
Tenth Frame	27
ACCOLADE	
Mean 18 Golf	29
Sundog	26
Mean 18 Course Dsk	14
ACTION	
Action Pak	28
Solapak	28
ACTIVISION	
Mindshadow	33
Borrowed Time	33
Hacker	29
Music Studio	39
Paintworks	47
Little Comp. People	33
Hacker II	33
Tass Times/Tonetown	29
Champ. Basketball	29
Shanghai	29
ANTIC	
C.O.L.R. Obj. Editor	23
Maps & Legends	27
Macro Assembler	60
Lattice C	114
Disk Doctor	23
A-Calc	45
Meta Pascal	75
CAD-3D	38
A-Ram	15
A-Seka	27
GST C Compiler	60
GST-ASM	45
Expert Opinion	75
Flash	30
Star Struck	19
Red Alert	19
Kermit & Remote Co.	19
Crystal	19
Fonts, Primitires	19
Cartographer	27
G.I.S.T.	27
K-Switch	23
K-Resource	38
ARTWORX	
Bridge 4.0	19
Compubridge	19
Maillist	12
Strip Poker	25
Peggammon	12
Hole in One Golf	19
ATARI	
DB Man	99
DB Master	33
Joust	19
Star Raiders	19
BATTERIES INCLUDED	
Degas	26
Degas Elite	49
Timelink	33
Thunder	26
I/S Talk	33
I/S Paperclp Elite	Call
Isgur Portfolio	139
BECKEMEYER	
Micro C-Shell	39
Micro C-Tools	19
Micro Make	27
RTX	53
Micro MTC Shell	99
Hard Disk Accelerate	30
CENTRAL POINT	
Copy II	27
DAC	
Easy Accounting	49
Easy Payroll	34
DATASOFT SOFTWARE	
Mercenary	26
Alternate Reality	23
EIDERSOFT	
ST Protector	22
ST Karate	22
ELECTRONIC ARTS	
Financial Cookbook	36
Ultima III	39
Coloring Book	15
Golden Oldies	23
Arctic Fox	29
Skyfox	33
Starfleet I	32
Ogre	29
EPYX	
Winter Games	26
Rogue	26
Temple of Apshai	26
World Games	26
Champion. Wrestling	26
Super Cycle	26
FIREBIRD	
The Pawn	29
Starglider	29
FIRST BYTE	
Kid Talk	33
Speller Bee	33
Math Talk	33
First Shapes	33
HABA	
Habacom	19
Phonebook	39
Business Letters	29
Habawriter II	39
HYBRID ARTS	
EZ Track	39
CZ Android	59
LOGICAL DESIGN WORKS	
LDW Basic Compiler	49
MARKSMAN TECHNOLOGY	
Phaser	69
MASTERTRONICS	
Renegade	14
MEGASOFT	
A Copier	27
X-Rated Library	17
MICHTRON	
DOS Shell	26
The Animator	26
Personal Money Mgr	33
Utilities	39
M-Disk	26
Softspool	26
Flip Side	26
Calendar	19
Mi-Term	33
Time Bandits	26
Cornerman	33
Mighty Mail	33
Cards	26
Business Tools	33
Major Motion	26
Mi-Dupe	20
BBS 2.0	54
Pro Football Wizard	27
Pinball Factory	27
Your Finance Future	26
Mi-Print	20
Trivia Challenge	27
GFA Basic	54
Trim Base	69
Match Point	27
Raid	27
Super Directory	27
MICRODEAL	
Karate Kid II	29
MICROPROSE	
Silent Service	27
MI-GRAPH	
Easy Draw	54
Fast	34
MILES	
Harrier Strike	34
MINDSCAPE	
Brataccus	33
S.D.I.	33
Balance of Power	33
MONOGRAM	
Dollars & Sense	69
NAVARONNE	
Astrology	29
Timekeeper	34
OSS	
Personal Pascal	50
PROCO	
Pro-Copy	24
PROGRESSIVE COMP. APPL.	
Graphic Artist	139
Font Editor	54
Font Pak I	34
PROGRESSIVE PERIPHERALS	
Picasso's Revenge	40
PSYGNOSIS	
Deep Space	34
Arena	26
QUANTUM	
ST-Talk	14
QUICKVIEW	
Zoomracks II	99
REGENT	
Regent Spell	34
Regent Base	69
Regent Word II	69
SHANNER	
Soundwave SW-1	15
ST-Key	10
Macro-Manager	15
Macro-Desk	10
DB Calc II	15
Colorwriter	10
SHELBOURNE	
Pool	24
Shuffleboard	20
SIERRA	
Kings Quest II	33
Ultima II	20
Winnie the Pooh	17
Black Cauldron	26
K/Q Hint Book	6
B/C Hint Book	6
Donald Duck	17
Cash Disbursements	67
Space Quest	33
Kings Quest III	33
Kings Quest I	33
SOFTLAB	
Oscar	32
SOFTLOGIK	
Publish. Partner	99
SPECTRUM HOLOBYTE	
Gato	27
SPINNAKER	
Homework Helper:Math	34
Treasure Island	27
SSI	
Phantasie	27
Phantasie II	27
SUBLOGIC	
Flight Simulator	38
TDI	
Modula-2	59
Modula-2 Develp.Kit	99
UCSD Pascal	59
TIMEWORKS	
Datamanager	53
Swiftcalc	53
Wordwriter	53
UNICORN	
Fraction Action	27
Decimal Dungeon	27
Animal Kingdom	27
UNISON WORLD	
Printmaster	26
Art Gallery I	19
Art Gallery II	19
VIP	
VIP Professional	149
VIP Lite	69
X-LENT	
Typesetter Elite	29
Rubber Stamp	25
Music Box	32
Megafont	25
Write 90 Degrees	19
PM Interface	19
Typeset Color Icon	13

PRINTERS

STAR MICRONICS	
NP-10	189
NX-10	229
PANASONIC	
KX-P 1080i	199
KX-P 1091i	259
KX-P 3131	259
KX-P 1592	399
KX-P 1595	579
KX-P 1092i	329
MODEMS	
Supra 300 ST	59
Supra 1200 ST	149
Avatex 1200	99
Avatex 1200 HC	139
Compuserve Start Kt	24
Microstuffer	59
Omega Terminal	20

ST ACCESSORIES

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

ST BOOKS

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

All books are $16 ea.

CABLES

ST to Printer	15
ST to Modem	15

ATARI 520 ST HARDWARE:

CALL

To order call TOLL FREE

1-800-824-7506

ORDER LINE ONLY

No extra charge for MasterCard or Visa

COMPUTER CREATIONS, Inc.

P.O. BOX 493 - DAYTON, OHIO 45459

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

VISA® MasterCard

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

EDUCATIONAL FEATURE

Low and medium resolution

Music Steps and Triads

This program helps music students learn their keyboards.

by Regena

When I taught piano lessons, I used the computer to help with some of the repetitious drill work. I've written programs to help beginning piano students learn the names of the keys on the keyboard, then learn to read notes written on the treble and bass staffs.

Music Steps and Triads is a tutorial plus drill-and-practice system. It's designed to supplement the teacher's instructions for distinguishing between half steps and whole steps on the keyboard, counting steps between two notes, and using a counting steps method to distinguish types of triads. There are several options in the program.

Half Steps.

A half step is from one key to the very next key on the piano or organ keyboard. Although a half step may be a rise or a fall in pitch, this program only considers upward movement. The program draws a keyboard, on which examples of half steps are illustrated with red arrows. A quiz is then presented which asks if the arrows represent a half step. Notes are chosen randomly and the tones sounded, so the student will see and hear the difference between the two notes. The student presses *1* for yes or *2* for no. There are ten questions.

Whole Steps.

A whole step is equal to two half steps. A keyboard is drawn with examples of whole steps. The quiz for this section asks the student to press *1* for a half step and *2* for a whole step. Arrows are drawn and tones sounded for the ten questions.

Count the Steps.

This section is a quiz of ten questions. Two keys are chosen randomly and the corresponding tones played. The student must indicate the correct number of steps between the two notes: 1=a half step; 2=a whole step; 3=one and a half steps; 4=two steps; and 5=two and a half steps. If the answer is correct, an arpeggio is played. If the answer is incorrect, the correct answer is shown. Arrows showing each half step are drawn on the keys, so the student can count the proper number of steps.

Identifying Triads.

I use the counting steps method to identify the type of triad a root chord is. The C chord is used for an example, because it's easy to remember. For each triad, the keys are marked on the keyboard with the number of steps between notes. The chord is also played, so the student can learn to hear differences among the triads.

Start with the major chord. The steps between the notes are 2 steps then 1½ steps— for example, for the C Major chord, there are 2 steps between C and E, then 1½ steps between E and G. To make the chord minor, the middle note is lowered one half step to E-flat. The number of steps now are 1½ then 2. The augmented chord starts with the major chord (2, 1½), then "augments" or enlarges the chord, so the order is 2 steps then 2 steps. The diminished chord starts with the minor chord (1½,2), then "diminishes" or reduces the chord to 1½ steps and 1½ steps.

The quiz randomly chooses a beginning note, a middle note either 1½ or 2 steps higher than the first note, and a top note either 1½ or 2 steps higher than the middle note. The three notes are sounded separately, then together. The student chooses whether the chord is major, minor, augmented or diminished. If the answer chosen is correct, an arpeggio is played. If the answer is incorrect, the number of steps between each note is illustrated and the correct answer is given. The quiz consists of ten triads.

Music Steps and Triads *continued*

Triads Quiz.

If the student wants more practice in identifying triads, this choice presents the quiz without the tutorial explanation.

That's it. **Music Steps and Triads** should be a help in your—or your students'—musical learning experience.

Regena got her first home computer (TI-99/4) for Christmas in 1980. Ideas for her hundreds of published BASIC programs (for various computers) come from her six children. A regular columnist in COMPUTE!, *her latest book is* Elementary ST BASIC, *from COMPUTE! Publishers, Inc.*

Listing 1.
ST BASIC listing.

```
10 'MUSIC STEPS AND TRIADS
20 'by REGENA
30 '   P. O. Box 1502
40 '   Cedar City, Utah  84720
50 DIM C$(3),H(16),N1(16),N2(16),YY(16
)
60 L1=80:L2=100
70 H$=CHR$(171):A$=CHR$(3)
80 L$=CHR$(169)+"-":R$="-"+CHR$(170)
90 C$(1)=L$+"1"+H$+R$:C$(2)=L$+"-2-"+R
$
100 C$(0)=L$+"1"+H$+"-"+R$:C$(3)=L$+"-
2--"+R$
110 FOR C=0 TO 16
120 READ H(C),N1(C),N2(C)
130 YY(C)=L2
140 NEXT C
150 DATA 37,1,3,50,2,3,62,3,3,75,4,3
160 DATA 87,5,3,112,6,3,125,7,3,137,8,
3
170 DATA 150,9,3,162,10,3,175,11,3
180 DATA 187,12,3,212,1,4,225,2,4
190 DATA 237,3,4,250,4,4,262,5,4
200 YY(1)=L1:YY(3)=L1
210 YY(6)=L1:YY(8)=L1:YY(10)=L1
220 YY(13)=L1:YY(15)=L1
230 COLOR 1,1,1,1,1
240 FULLW 2:CLEARW 2:GOTOXY 8,1
250 PRINT "MUSIC STEPS AND TRIADS"
260 PRINT:PRINT "CHOOSE:"
270 PRINT:PRINT "  1  Half Steps"
280 PRINT:PRINT "  2  Whole Steps"
290 PRINT:PRINT "  3  Count the Steps"
300 PRINT:PRINT "  4  Identifying Tria
ds"
310 PRINT:PRINT "  5  Triads Quiz"
320 PRINT:PRINT "  6  End Program"
330 E=INP(2):IF E<49 OR E>54 THEN 330
340 CLEARW 2:GOTOXY 0,0:WAVE 1
350 ON E-48 GOSUB HALF,WHOLE,COUNT,IDE
NT,QUIZ,ENDPROG
360 GOTO 230
370 STOP
380 F1:
390 GOTOXY 20,17:PRINT "<Press F1>";
400 EE=INP(2):IF EE<>187 THEN 400
410 GOTOXY 20,17:PRINT SPACE$(10);
420 RETURN
430 BLACK:
440 COLOR 1,1,1,1,1
450 LINEF X-W,Y,X-W,Y+L
460 LINEF X-W,Y+L,X+W,Y+L
470 LINEF X+W,Y+L,X+W,Y
480 LINEF X-W,Y,X+W,Y
490 FILL X-2,Y+2
500 FILL X+2,Y+2
510 RETURN
520 KEYBOARD:
530 Y=30:L=80:COLOR 1,1,1,1,1
540 LINEF 25,Y,275,Y
550 LINEF 25,Y,25,Y+L
560 LINEF 25,Y+L,275,Y+L
570 LINEF 275,Y+L,275,Y
580 FOR X=50 TO 250 STEP 25
590 LINEF X,Y,X,Y+L
600 NEXT X
610 RESTORE 620
620 DATA 50,75,125,150,175,225,250
630 W=7:L=60
640 FOR C=1 TO 7
650 READ X:GOSUB BLACK
660 NEXT C
670 RETURN
680 UP:
690 LINEF K2-15,L2,K2,L1
700 LINEF K2,L1,K2-4,L1+3
710 LINEF K2,L1,K2-1,L1+6
720 RETURN
730 DN:
740 LINEF K2-13,L1,K2,L2
750 LINEF K2,L2,K2,L2-5
760 LINEF K2,L2,K2-5,L2-1
770 RETURN
780 WH:
790 LINEF K2-25,L2,K2,L2
800 LINEF K2,L2,K2-5,L2-3
810 LINEF K2,L2,K2-5,L2+3
820 RETURN
830 BL:
840 LINEF K2-25,L1,K2,L1
850 LINEF K2,L1,K2-5,L1-3
860 LINEF K2,L1,K2-5,L1+3
870 RETURN
880 UP2:
890 LINEF K2-37,L2,K2,L1
900 LINEF K2,L1,K2-4,L1
910 LINEF K2,L1,K2-2,L1+5
920 RETURN
930 DN2:
940 LINEF K2-37,L1,K2,L2
950 LINEF K2,L2,K2-2,L2-5
960 LINEF K2,L2,K2-5,L2-1
970 RETURN
980 E1:
990 COLOR 1,1,1:GOSUB UP
1000 COLOR 1,0,0:LINEF K2-15,L2,K2-8,L
1+11
1010 RETURN
1020 E2:
1030 COLOR 1,0,0:GOSUB DN
1040 COLOR 1,1,1:LINEF K2-13,L1,K2-6,L
1+10
1050 LINEF K2-13,L1,K2-7,L1+10
1060 RETURN
1070 E3:
1080 COLOR 1,0,0:GOSUB WH
1090 COLOR 1,0,1:LINEF K2-12,L1,K2-12,
L2+5
1100 RETURN
1110 E4:
1120 COLOR 1,1,1:GOSUB BL
1130 COLOR 1,0,0:LINEF K2-17,L1,K2-8,L
1
1140 RETURN
1150 E5:
1160 COLOR 1,1,1:GOSUB UP2
1170 COLOR 1,0,0:LINEF K2-37,L2,K2-8,L
1+4
1180 LINEF K2-37,L2,K2-8,L1+5
1190 COLOR 1,0,1:LINEF K2-25,L1,K2-25,
L2
```

```
1200 RETURN
1210 E6:
1220 COLOR 1,0,0:GOSUB DN2
1230 COLOR 1,1,1:LINEF K2-12,L1,K2-12,
L2
1240 LINEF K2-37,L1,K2-30,L1+4
1250 RETURN
1260 CORRECT:
1270 WAVE 1
1280 SOUND 1,15,1,4,3
1290 SOUND 1,15,5,4,3
1300 SOUND 1,15,8,4,3
1310 SOUND 1,15,1,5,6
1320 SOUND 1,0,1,1,0
1330 RETURN
1340 NT:
1350 WAVE 1
1360 SOUND 1,15,5,3,2
1370 SOUND 1,15,1,3,2
1380 SOUND 1,0,1,1,50:SOUND 1,0,1,1,0
1390 RETURN
1400 REWARD:
1410 RESTORE 1460
1420 FOR P=1 TO 18
1430 READ J,K:SOUND 1,15,J,K,5
1440 NEXT P
1450 SOUND 1,0,1,1,0
1460 DATA 1,4,5,4,8,4,1,5
1470 DATA 8,4,1,5,5,4,8,4,1,5
1480 DATA 5,5,1,5,5,5,8,4
1490 DATA 1,5,5,5,8,5,5,5,8,5
1500 RETURN
1510 HALF:
1520 PRINT:PRINT TAB(10);"H A L F   S
T E P S"
1530 GOSUB KEYBOARD
1540 GOTOXY 0,14
1550 PRINT "   A ";
1560 COLOR 2:PRINT "half step";
1570 COLOR 1:PRINT " is from one key"
1580 PRINT "   to the very next key."
1590 COLOR 1,1,2
1600 K2=H(1):GOSUB UP
1610 K2=H(5):GOSUB WH
1620 K2=H(9):GOSUB DN
1630 K2=H(12):GOSUB WH
1640 GOSUB F1
1650 CLEARW 2:GOTOXY 0,0
1660 PRINT "IS THIS A HALF STEP?"
1670 GOTOXY 0,14:PRINT TAB(10);"1 YES"
1680 PRINT TAB(10);"2 NO"
1690 GOSUB HALFQUIZ
1700 RETURN
1710 HALFQUIZ:
1720 GOSUB KEYBOARD
1730 RANDOMIZE 0
1740 FOR P=1 TO 10
1750 COLOR 1,1,2
1760 ANS=INT(2*RND+1)
1770 IF ANS=2 THEN 1810
1780 CH=INT(RND*16+1):K2=H(CH)
1790 ON CH GOSUB UP,DN,UP,DN,WH,UP,DN,
UP,DN,UP,DN,WH,UP,DN,UP,DN
1800 GOTO 1830
1810 CH=INT(RND*15+2):K2=H(CH)
1820 ON CH-1 GOSUB WH,BL,WH,DN2,UP2,WH
,BL,WH,BL,WH,DN2,UP2,WH,BL,WH
1830 WAVE 1:C1=CH-ANS
1840 SOUND 1,15,N1(C1),N2(C1),6
1850 SOUND 1,15,N1(CH),N2(CH),6
1860 SOUND 1,0,1,1,0
1870 E=INP(2):IF E<49 OR E>50 THEN 187
0
1880 GOTOXY 20,13+E-48:COLOR 2:PRINT "
*"
1890 IF E=ANS+48 THEN GOSUB CORRECT EL
SE GOSUB NT
1900 ON ANS GOTO 1910,1930
1910 ON CH GOSUB E1,E2,E1,E2,E3,E1,E2,
E1,E2,E1,E2,E3,E1,E2,E1,E2
1920 GOTO 1940
1930 ON CH-1 GOSUB E3,E4,E3,E6,E5,E3,E
4,E3,E4,E3,E6,E5,E3,E4,E3
1940 GOTOXY 20,13+E-48:COLOR 1:PRINT "
 "
1950 NEXT P
1960 GOSUB REWARD
1970 RETURN
1980 WHOLE:
1990 PRINT TAB(7);"W H O L E   S T E P
 S"
2000 GOSUB KEYBOARD
2010 GOTOXY 0,14:PRINT "   A ";
2020 COLOR 2:PRINT "whole step ";
2030 COLOR 1:PRINT "is equal to"
2040 PRINT "   two half steps (1 = "H$
" + "H$")."
2050 COLOR 1,0,2
2060 K2=H(2):GOSUB WH
2070 K2=H(6):GOSUB UP2
2080 K2=H(12):GOSUB DN2
2090 K2=H(15):GOSUB BL
2100 GOSUB F1
2110 CLEARW 2:GOTOXY 0,0
2120 PRINT "WHAT KIND OF MUSICAL STEP?
"
2130 GOTOXY 0,14
2140 PRINT "PRESS 1 FOR HALF STEP"
2150 PRINT "PRESS 2 FOR WHOLE STEP"
2160 GOSUB HALFQUIZ
2170 RETURN
2180 COUNT:
2190 SC=0
2200 PRINT "COUNT HOW MANY STEPS THERE
 ARE"
2210 PRINT "BETWEEN TWO NOTES."
2220 GOTOXY 0,13:PRINT "    1  HALF ST
EP"
2230 PRINT "    2  WHOLE STEP"
2240 PRINT "    3  1"H$" STEPS"
2250 PRINT "    4  2 STEPS"
2260 PRINT "    5  2"H$" STEPS";
2270 GOSUB KEYBOARD
2280 RANDOMIZE 0
2290 FOR P=1 TO 10
2300 CH=INT(RND*9+1):X1=H(CH):Y1=YY(CH
)
2310 SOUND 1,15,N1(CH),N2(CH),10
2320 COLOR 2,2,2:PCIRCLE X1,Y1,5
2330 ST=INT(RND*4+1):C2=CH+ST
2340 X2=H(C2):Y2=YY(C2)
2350 SOUND 1,15,N1(C2),N2(C2),10
2360 PCIRCLE X2,Y2,5
2370 SOUND 1,0,1,1,0
2380 E=INP(2):IF E<49 OR E>53 THEN 238
0
2390 E=E-48:GOTOXY 2,12+E:PRINT "*";
2400 IF E=ST THEN GOSUB CORRECT:SC=SC+
1:GOTO 2500
2410 COLOR 3,3,3
2420 FOR J=1 TO ST
2430 K=CH+J:K2=H(K)
2440 ON K GOSUB UP,DN,UP,DN,WH,UP,DN,U
P,DN,UP,DN,WH,UP,DN,UP,DN
2450 SOUND 1,15,N1(K),N2(K),10
2460 GOTOXY 1,12+J:PRINT A$;
2470 NEXT J
2480 SOUND 1,0,1,1,0
2490 FOR DELAY=1 TO 2000:NEXT DELAY
2500 IF Y1=L1 THEN COLOR 1,1,1 ELSE CO
LOR 1,0,0
2510 PCIRCLE X1,Y1,5
2520 IF Y2=L1 THEN COLOR 1,1,1 ELSE CO
LOR 1,0,0
2530 PCIRCLE X2,Y2,5
2540 GOTOXY 2,12+E:PRINT " ";
```

Music Steps and Triads *continued*

```
2550 IF E=ST THEN 2610
2560 FOR J=1 TO ST
2570 GOTOXY 1,12+J:PRINT " ";
2580 K=CH+J:K2=H(K)
2590 ON K GOSUB E1,E2,E1,E2,E3,E1,E2,E
1,E2,E1,E2,E3,E1,E2,E1,E2
2600 NEXT J
2610 NEXT P
2620 CLEARW 2:GOTOXY 0,0
2630 PRINT "COUNTING STEPS"
2640 PRINT:PRINT
2650 PRINT "SCORE = ";STR$(SC);"0 PERC
ENT"
2660 IF SC=10 THEN GOSUB REWARD
2670 GOSUB F1
2680 RETURN
2690 IDENT:
2700 PRINT TAB(5);"IDENTIFYING TRIADS"
2710 ?:? " A triad consists of 3 notes
."
2720 ?:? " It is in root position if a
ll"
2730 ? " three notes are on lines or i
f"
2740 ? " all three notes are on spaces
"
2750 ? " on the staff."
2760 COLOR 1,0,4
2770 FOR P=90 TO 150 STEP 15
2780 LINEF 10,P,270,P
2790 NEXT P
2800 RESTORE 2840:COLOR 1,0,2
2810 FOR P=1 TO 6
2820 READ X,Y:ELLIPSE X,Y,12,7
2830 NEXT P
2840 DATA 100,150,100,135,100,120
2850 DATA 150,142,150,127,150,112
2860 GOSUB F1
2870 CLEARW 2:GOTOXY 0,0
2880 ? " To identify a chord, first ma
ke"
2890 ? " sure the notes are in root po
sition"
2900 ? " (invert if necessary).":PRINT
2910 ? " The name of the chord is the"
2920 ? " bottom note of the root chord
."
2930 GOSUB F1
2940 CLEARW 2:GOTOXY 0,0
2950 ? "The basic triads are"
2960 COLOR 2
2970 ? TAB(10);"major"
2980 ? TAB(10);"minor"
2990 ? TAB(10);"augmented"
3000 ? TAB(10);"diminished":PRINT
3010 COLOR 1
3020 ? " The type of triad may be dete
rmined"
3030 ? " by counting steps between not
es"
3040 ? " of the root chord."
3050 GOSUB F1
3060 CLEARW 2:GOTOXY 0,0
3070 PRINT "   MAJOR TRIAD"
3080 GOSUB KEYBOARD:WAVE 7
3090 GOTOXY 0,13:? "The C Major triad
consists of C, E, G"
3100 COLOR 1,2,2:PCIRCLE H(0),L2,5
3110 SOUND 1,15,1,4,50
3120 PCIRCLE H(4),L2,5
3130 SOUND 2,15,5,4,50
3140 PCIRCLE H(7),L2,5
3150 SOUND 3,15,8,4,50
3160 PRINT " 2  steps from C to E"
3170 PRINT " 1"H$" steps from E to G"
3180 COLOR 4:GOTOXY 4,2:PRINT C$(2);C$
(1)
3190 WAVE 0:COLOR 1
3200 GOSUB F1
3210 GOTOXY 0,0:PRINT "CHANGE TO MINOR
 TRIAD"
3220 GOTOXY 0,13:? "Lower the middle n
ote a half step.     "
3230 ? " 1"H$" steps from C to E-flat"
3240 ? " 2  steps from E-flat to G"
3250 WAVE 7
3260 SOUND 1,15,1,4,50
3270 PCIRCLE H(3),L1,5
3280 COLOR 4,0,0:PCIRCLE H(4),L2,5
3290 SOUND 2,15,4,4,50
3300 GOTOXY 3,2:PRINT C$(1);C$(3)
3310 SOUND 3,15,8,4,50
3320 WAVE 0:COLOR 1
3330 GOSUB F1
3340 CLEARW 2:GOTOXY 0,0
3350 PRINT "   AUGMENTED TRIAD":PRINT
3360 ?:? " Start with the major triad.
"
3370 ?:? " 'Augment' the triad by movi
ng"
3380 ?:? " the top note up one half st
ep."
3390 GOSUB F1
3400 CLEARW 2:GOTOXY 0,0
3410 PRINT "   AUGMENTED TRIAD"
3420 GOTOXY 0,13:PRINT " C Augmented T
riad--C, E, G#"
3430 ? "  2  steps from C to E"
3440 ? "  2  steps from E to G#"
3450 GOSUB KEYBOARD:WAVE 7
3460 COLOR 1,2,2:PCIRCLE H(0),L2,5
3470 SOUND 1,15,1,4,50
3480 PCIRCLE H(4),L2,5
3490 SOUND 2,15,5,4,50
3500 PCIRCLE H(8),L1,5
3510 SOUND 3,15,9,4,50
3520 COLOR 4:GOTOXY 4,2
3530 PRINT C$(2);C$(3)
3540 COLOR 1:WAVE 0
3550 GOSUB F1
3560 CLEARW 2:GOTOXY 0,0
3570 PRINT "   DIMINISHED TRIAD":PRINT
3580 ?:? " Start with the minor triad.
"
3590 ?:? " 'Diminish' the triad by low
ering"
3600 ?:? " the top note one half step.
"
3610 GOSUB F1
3620 CLEARW 2:GOTOXY 0,0
3630 PRINT "   DIMINISHED TRIAD"
3640 GOSUB KEYBOARD
3650 GOTOXY 0,13:PRINT "  C Diminished
 triad"
3660 ? "   1"H$" steps from C to E-fla
t"
3670 ? "   1"H$" steps from E-flat to
G-flat"
3680 COLOR 4,2,2:PCIRCLE H(0),L2,5
3690 WAVE 7
3700 SOUND 1,15,1,4,50
3710 PCIRCLE H(3),L1,5
3720 SOUND 2,15,4,4,50
3730 PCIRCLE H(6),L1,5
3740 SOUND 3,15,7,4,50
3750 GOTOXY 3,2
3760 PRINT C$(1);C$(0)
3770 WAVE 0:COLOR 1
3780 GOSUB F1
3790 CLEARW 2:GOTOXY 0,0
3800 PRINT TAB(13);"STEPS BETWEEN NOTE
S"
3810 PRINT "TRIAD";TAB(13);"1st     2n
d     3rd"
3820 ?:? "MAJOR";TAB(18);"2";SPC(7);"1
";H$
```

```
3830 ?:? "MINOR";TAB(18);"1";H$;SPC(6)
;"2"
3840 ?:? "AUGMENTED";TAB(18);"2";SPC(7
);"2"
3850 ?:? "DIMINISHED";TAB(18);"1";H$;S
PC(6);"1";H$
3860 GOSUB F1
3870 QUIZ:
3880 CLEARW 2:GOTOXY 0,0
3890 PRINT "NAME THE TYPE OF TRIAD"
3900 GOSUB KEYBOARD:SC=0
3910 GOTOXY 0,13
3920 PRINT "     1  MAJOR"
3930 PRINT "     2  MINOR"
3940 PRINT "     3  AUGMENTED"
3950 PRINT "     4  DIMINISHED"
3960 RANDOMIZE 0
3970 FOR P=1 TO 10
3980 WAVE 0:SOUND 1,0:SOUND 2,0:SOUND
3,0
3990 WAVE 7:COLOR 2,2,2,1,1
4000 CH=INT(RND*9)
4010 X1=H(CH):Y1=YY(CH)
4020 PCIRCLE X1,Y1,5
4030 SOUND 1,15,N1(CH),N2(CH),50
4040 S1=INT(2*RND+1)+2
4050 C2=CH+S1:X2=H(C2):Y2=YY(C2)
4060 PCIRCLE X2,Y2,5
4070 SOUND 2,15,N1(C2),N2(C2),50
4080 S2=INT(2*RND+1)+2
4090 C3=C2+S2:X3=H(C3):Y3=YY(C3)
4100 PCIRCLE X3,Y3,5
4110 SOUND 3,15,N1(C3),N2(C3),50
4120 IF S1=4 AND S2=3 THEN ANS=1
4130 IF S1=3 AND S2=4 THEN ANS=2
4140 IF S1=4 AND S2=4 THEN ANS=3
4150 IF S1=3 AND S2=3 THEN ANS=4
4160 WAVE 0
4170 E=INP(2):IF E<49 OR E>52 THEN 417
0
4180 E=E-48:GOTOXY 3,E+12
4190 WAVE 7:PRINT "*";
4200 IF E=ANS THEN SC=SC+1:GOTO 4320
4210 COLOR 4:GOTOXY 1,12+ANS:PRINT A$;
4220 GOTOXY CH*1.75+3,2
4230 IF CH<=2 OR CH=5 THEN PRINT " ";
4240 SOUND 1,15,N1(CH),N2(CH),50
4250 IF S1=3 THEN PRINT C$(1);:GOTO 42
70
4260 IF CH=0 OR CH>4 THEN ? C$(2); ELS
E ? C$(3);
4270 SOUND 2,15,N1(C2),N2(C2),50
4280 IF S2=3 THEN PRINT C$(1):GOTO 430
0
4290 IF C2=5 OR C2=6 OR C2=7 OR C2=8 O
R C2=12 THEN ? C$(2) ELSE ? C$(3)
4300 SOUND 3,15,N1(C3),N2(C3),50
4310 GOTO 4350
4320 SOUND 1,15,N1(CH),N2(CH),50
4330 SOUND 2,15,N1(C2),N2(C2),50
4340 SOUND 3,15,N1(C3),N2(C3),50
4350 WAVE 0
4360 IF E=ANS THEN 4410 ELSE GOSUB F1
4370 GOTOXY 2,2:PRINT SPACE$(40)
4380 FOR J=1 TO ANS
4390 GOTOXY 1,12+J:PRINT " ";
4400 NEXT J
4410 GOTOXY 3,12+E:PRINT " ";
4420 IF Y1=L1 THEN COLOR 1,1,1 ELSE CO
LOR 1,0,0
4430 PCIRCLE X1,Y1,5
4440 IF Y2=L1 THEN COLOR 1,1,1 ELSE CO
LOR 1,0,0
4450 PCIRCLE X2,Y2,5
4460 IF Y3=L1 THEN COLOR 1,1,1 ELSE CO
LOR 1,0,0
4470 PCIRCLE X3,Y3,5
4480 NEXT P
4490 CLEARW 2:GOTOXY 0,0
4500 PRINT "IDENTIFYING TRIADS"
4510 PRINT:PRINT "SCORE = ";STR$(SC);"
0 PERCENT"
4520 IF SC=10 THEN GOSUB REWARD
4530 ?:?:? "TRY AGAIN?  Y/N"
4540 E=INP(2)
4550 IF E=89 OR E=121 THEN 3870
4560 IF E=78 OR E=110 THEN 230 ELSE 45
40
4570 ENDPROG:
4580 END
```

●

ST CHECKSUM DATA.

(see page 8)

```
 10 data  464, 577, 912, 38, 208, 611,
 849, 651, 992, 299, 5601
 110 data  968, 610, 441, 263, 658, 98
8, 505, 382, 327, 524, 5666
 210 data  743, 634, 27, 916, 351, 254
, 655, 738, 450, 170, 4938
 310 data  679, 625, 299, 633, 85, 406
, 110, 757, 638, 808, 5040
 410 data  30, 342, 165, 34, 983, 180,
 986, 676, 941, 910, 5247
 510 data  341, 677, 298, 447, 463, 80
9, 744, 11, 443, 303, 4536
 610 data  888, 510, 387, 840, 831, 27
9, 361, 839, 848, 901, 6684
```

```
 710 data  901, 348, 773, 833, 798, 92
4, 363, 839, 854, 912, 7545
 810 data  913, 350, 767, 839, 923, 92
4, 365, 1, 860, 760, 6702
 910 data  906, 352, 935, 847, 924, 92
8, 367, 766, 976, 548, 7549
 1010 data  438, 827, 887, 548, 158, 4
43, 834, 899, 487, 440, 5961
 1110 data  833, 888, 274, 444, 839, 6
9, 524, 141, 322, 443, 4777
 1210 data  840, 42, 308, 106, 448, 58
6, 250, 205, 218, 221, 3224
 1310 data  206, 135, 449, 915, 251, 2
15, 204, 27, 455, 466, 3323
 1410 data  36, 56, 466, 366, 141, 528
, 990, 557, 984, 452, 4576
 1510 data  114, 71, 408, 917, 911, 35
2, 375, 677, 726, 492, 5043
 1610 data  503, 465, 488, 463, 38, 16
5, 574, 763, 470, 458, 4387
 1710 data  848, 413, 997, 59, 728, 51
6, 568, 611, 666, 579, 5985
 1810 data  609, 427, 506, 306, 376, 1
54, 375, 762, 160, 955, 4630
 1910 data  613, 587, 261, 750, 382, 2
20, 471, 373, 619, 392, 4668
 2010 data  484, 449, 354, 173, 706, 4
83, 598, 620, 468, 446, 4781
 2110 data  21, 33, 906, 209, 564, 454
, 449, 395, 944, 805, 4780
 2210 data  960, 842, 431, 231, 954, 4
19, 405, 989, 51, 753, 6035
 2310 data  493, 867, 635, 576, 431, 4
75, 142, 366, 194, 259, 4438
 2410 data  727, 176, 117, 118, 362, 4
34, 353, 146, 453, 455, 3341
 2510 data  474, 459, 478, 462, 536, 1
83, 468, 125, 248, 352, 3785
 2610 data  371, 37, 612, 98, 135, 780
, 468, 465, 347, 832, 4145
 2710 data  175, 196, 547, 203, 321, 7
32, 177, 232, 382, 509, 3474
 2810 data  28, 735, 379, 300, 337, 47
3, 48, 764, 844, 816, 4724
 2910 data  46, 752, 473, 48, 574, 359
, 760, 777, 624, 722, 5135
 3010 data  328, 569, 189, 338, 450, 2
5, 238, 349, 62, 867, 3415
 3110 data  297, 480, 312, 488, 324, 6
24, 916, 850, 289, 451, 5031
 3210 data  916, 889, 350, 224, 270, 3
05, 483, 879, 318, 914, 5548
 3310 data  326, 288, 457, 32, 627, 89
2, 423, 832, 463, 31, 4371
 3410 data  743, 77, 52, 163, 358, 882
, 312, 495, 327, 495, 3904
 3510 data  335, 812, 269, 295, 465, 4
0, 822, 928, 950, 231, 5147
 3610 data  464, 39, 876, 416, 579, 28
4, 179, 893, 286, 314, 4330
 3710 data  492, 326, 500, 338, 831, 2
74, 305, 474, 49, 948, 4537
 3810 data  602, 657, 625, 841, 762, 4
75, 193, 51, 321, 43, 4570
 3910 data  928, 689, 719, 276, 384, 1
0, 72, 22, 79, 165, 3344
 4010 data  630, 464, 498, 452, 617, 4
70, 437, 458, 588, 469, 5083
 4110 data  438, 712, 716, 721, 723, 2
49, 355, 923, 596, 220, 5653
 4210 data  666, 738, 267, 505, 746, 2
27, 443, 563, 810, 443, 5408
 4310 data  579, 506, 442, 447, 254, 2
39, 979, 289, 468, 350, 4553
 4410 data  462, 458, 477, 462, 481, 4
66, 485, 376, 42, 95, 3804
 4510 data  15, 777, 167, 407, 458, 49
4, 607, 940, 3865
```

●

ATTENTION PROGRAMMERS!

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

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

Send programs to:
Editor, **ST-Log**, P.O. Box 23, Worcester, MA 01603.

CZ-101 Patch Librarian

Expand your keyboard horizons— save every sound you'd like.

by Kirk Stover

With the addition of a MIDI synthesizer to your 520 system, you have opened a new door. The Casio **CZ-101** keyboard is capable of many unique sounds—from brass ensembles and electric guitars to laser guns.

CZ Patch Librarian is a program which allows you to save sounds to disk for later use. **CZ Patch** will greatly increase your capacity to create and save sounds from your keyboard.

The **CZ-101** has sixteen preset sounds with which it came from the factory, and room for sixteen internal memory sounds. The internal sounds are the ones you create; they can be whatever your imagination is capable of. With **CZ Patch**, you're no longer limited to keeping just sixteen of your own sound creations, but can save every sound you want.

Using it.

To use the program, type in Listings 1 and 2. Check your typing with **ST-Check** (see page 8), and run each program separately under ST BASIC; there are no resolution dependent features. The files CZPATCH.PRG and CZPATCH .RSC will be created on drive A. If you want to change the destination drive or the filename, change the assignment to *filename$* in Line 100 (the first line in each program). When each program ends, it will print the message *file written* in the output window. Make sure the destination disk for the files has space for a 6606-byte (CZPATCH.PRG) and a 552-byte (CZPATCH.RSC) file.

To begin, simply double click on CZPATCH.PRG, after you've verified that CZPATCH.RSC is on the same disk. This will present the dialog box you use to send and receive sounds. The number at the top of the box represents the sound you want to work with. Click the mouse on the up and down arrows to change tone numbers. These numbers encompass the MIDI standard, as shown below.

1 - 32 . . . . . Preset
33 - 64 . . . . . Internal
65 - 96 . . . . . Cartridge
97 . . . . . Sound Area

First, select which tone you want to receive from the keyboard. Tones being received can come from any of the four MIDI areas. Get the counter to represent that tone's number and click on "Receive." The program will then request the keyboard to send the tone data to the computer. If successful, you'll be presented with the file selector dialog box, and asked to name the sound file you're saving to disk.

At times, you'll want to send a sound you've previously saved to disk, in order to have it in your keyboard's memory. To do so, select the tone number which represents where you want your sound to be stored. This can be in any of the areas except preset. After clicking on "Send," a file selector dialog box will be displayed. You must choose the sound file you want to send. The program will then request the keyboard to listen for incoming tone data. Number 97, the sound area, is useful for testing sounds without overlaying any of the sounds currently stored in the keyboard. It will only remain in the sound area as long as you don't select any other sounds.

How it works.

Now that you know how to use **CZ Patch**, you may be interested in how the program communicates with the keyboard. First, the computer sends a "system exclusive" command to the keyboard, utilizing the BIOS BCONOUT command with device = MIDI (3). The computer then receives the MIDI's response through BIOS BCONIN, which is known as the "handshake." If the handshake is

successful, data transfer takes place. The example below illustrates the communications that are taking place:

RECEIVE REQUEST (Transmit tone data from **CZ-101** to computer)

COMPUTER		CZ-101		COMPUTER		CZ-101		COMPUTER
F0 44 00 00 70 10 tone#	↘	F0 44 00 00 70 30	↗	70 31	↘	[tone data]	↗	F7

SEND REQUEST (Transmit tone data from computer to **CZ-101**)

COMPUTER		CZ-101		COMPUTER		CZ-101
F0 44 00 00 70 20 tone#	↘	F0 44 00 00 70 30	↗	[tone data] F7	↘	F7

It is important to note that tone data is sent and received in high and low nibble order. ///

Kirk Stover is a Systems Analyst at an insurance company in Minnesota. He enjoys working on his 520ST in both C and assembly language. His special interest is writing time-saving utilities.

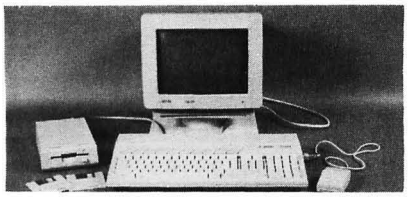
Listing 1.
ST BASIC listing.

```
100 filename$="a:\czpatch.prg"
110 fullw 2:clearw 2:gotoxy 0,0:print
"creating file..."
120 option base 0
125 dim a%(16000):def seg=1:v$=""
130 p=varptr(a%(0)):bptr=p+1
140 for i%=1 to 6606
150 read v$:code%=val("&H"+v$)
160 poke p, code%:print ".";
170 p=p+1
180 next
190 bsave filename$,bptr,6606
200 print "file written":end
1000 data 60,1A,00,00,18,D0,00,00,00,0
0,00,00,00,00,00,00
1010 data 00,00,00,00,00,00,00,00,00,0
0,00,00,2A,4F,2E,7C
1020 data 00,00,18,B6,2A,6D,00,04,20,2
D,00,0C,D0,AD,00,14
1030 data D0,AD,00,1C,06,80,00,00,01,0
0,2F,00,2F,0D,3F,3C
1040 data 00,00,3F,3C,00,4A,4E,41,DF,F
C,00,00,00,0C,61,00
1050 data 05,44,61,00,00,94,61,00,01,1
8,3F,3C,00,00,4E,41
1060 data 22,3C,00,00,0B,38,30,3C,00,C
8,4E,42,4E,75,22,3C
1070 data 00,00,0B,50,30,3C,00,73,4E,4
2,4E,75,33,FC,00,64
1080 data 00,00,0D,D2,33,FC,00,00,00,0
0,0D,D4,33,FC,00,0B
1090 data 00,00,0D,D8,33,F9,00,00,09,E
8,00,00,0D,DE,20,7C
1100 data 00,00,0D,4A,22,7C,00,00,0E,0
6,30,3C,00,0A,32,D8
1110 data 51,C8,FF,FC,61,B8,20,7C,00,0
0,10,06,22,7C,00,00
1120 data 0D,60,30,3C,00,2C,32,D8,51,C
8,FF,FC,20,7C,00,00
1130 data 11,06,30,3C,00,0B,32,D8,51,C
8,FF,FC,33,F9,00,00
1140 data 0D,DE,00,00,09,E8,4E,75,33,F
C,00,65,00,00,0D,D2
1150 data 33,FC,00,00,00,00,0D,D4,33,F
C,00,00,00,00,0D,D8
1160 data 33,F9,00,00,09,E8,00,00,0D,D
E,61,00,FF,62,4E,75
1170 data 23,FC,00,00,00,00,00,00,0D,F
6,23,FC,00,00,00,00
1180 data 00,00,0D,FA,23,FC,00,00,00,0
0,00,00,0D,FE,23,FC
1190 data 00,00,00,00,00,00,0E,02,33,F
C,00,0A,00,00,0D,D2
1200 data 33,FC,00,00,00,00,0D,D4,33,F
C,00,01,00,00,0D,D6
1210 data 33,FC,00,00,00,00,0D,D8,33,F
C,00,00,00,00,0D,DA
1220 data 61,00,FE,FE,33,F9,00,00,10,0
6,00,00,09,EA,4E,75
1230 data 33,FC,00,13,00,00,0D,D2,33,F
C,00,00,00,00,0D,D4
1240 data 33,FC,00,01,00,00,0D,D6,33,F
C,00,00,00,00,0D,D8
1250 data 33,FC,00,00,00,00,0D,DA,61,0
0,FE,C6,4E,75,33,FC
1260 data 00,2A,00,00,0D,D2,33,FC,00,0
6,00,00,0D,D4,33,FC
1270 data 00,01,00,00,0D,D6,33,FC,00,0
1,00,00,0D,D8,33,FC
1280 data 00,00,00,00,0D,DA,33,F9,00,0
0,09,F0,00,00,0E,06
1290 data 33,FC,00,0A,00,00,0E,08,33,F
9,00,00,09,F2,00,00
1300 data 0E,0A,33,F9,00,00,09,F4,00,0
0,0E,0C,33,F9,00,00
1310 data 09,F6,00,00,0E,0E,33,F9,00,0
0,09,F8,00,00,0E,10
1320 data 23,F9,00,00,0B,64,00,00,12,0
6,61,00,FE,54,30,39
1330 data 00,00,10,06,4E,75,33,FC,00,3
2,00,00,0D,D2,33,FC
1340 data 00,01,00,00,0D,D4,33,FC,00,0
1,00,00,0D,D6,33,FC
1350 data 00,01,00,00,0D,D8,33,FC,00,0
0,00,00,0D,DA,33,FC
1360 data 00,00,00,00,0E,06,23,F9,00,0
0,0B,64,00,00,12,06
1370 data 61,00,FE,0E,30,39,00,00,10,0
6,4E,75,33,FC,00,33
1380 data 00,00,0D,D2,33,FC,00,09,00,0
0,0D,D4,33,FC,00,01
1390 data 00,00,0D,D6,33,FC,00,00,00,0
0,0D,D8,33,FC,00,00
1400 data 00,00,0D,DA,33,C0,00,00,0E,0
6,33,FC,00,00,00,00
1410 data 0E,08,33,FC,00,00,00,00,0E,0
A,33,FC,00,00,00,00
1420 data 0E,0C,33,FC,00,00,00,00,0E,0
E,33,F9,00,00,09,F2
1430 data 00,00,0E,10,33,F9,00,00,09,F
4,00,00,0E,12,33,F9
1440 data 00,00,09,F6,00,00,0E,14,33,F
9,00,00,09,F8,00,00
1450 data 0E,16,61,00,FD,8C,30,39,00,0
0,10,06,4E,75,33,FC
1460 data 00,34,00,00,0D,D2,33,FC,00,0
1,00,00,0D,D4,33,FC
1470 data 00,01,00,00,0D,D6,33,FC,00,0
1,00,00,0D,D8,33,FC
1480 data 00,00,00,00,0D,DA,33,C0,00,0
0,0E,06,23,C8,00,00
1490 data 12,06,61,00,FD,4C,30,39,00,0
0,10,06,4E,75,33,FC
1500 data 00,36,00,00,0D,D2,33,FC,00,0
0,00,00,0D,D4,33,FC
1510 data 00,05,00,00,0D,D6,33,FC,00,0
1,00,00,0D,D8,33,FC
1520 data 00,00,00,00,0D,DA,23,F9,00,0
0,0B,64,00,00,12,06
1530 data 61,00,FD,0E,33,F9,00,00,10,0
8,00,00,09,F2,33,F9
1540 data 00,00,10,0A,00,00,09,F4,33,F
9,00,00,10,0C,00,00
1550 data 09,F6,33,F9,00,00,10,0E,00,0
0,09,F8,4E,75,33,FC
1560 data 00,4D,00,00,0D,D2,33,FC,00,0
0,00,00,0D,D4,33,FC
1570 data 00,05,00,00,0D,D6,33,FC,00,0
0,00,00,0D,D8,33,FC
1580 data 00,00,00,00,0D,DA,61,00,FC,B
8,30,39,00,00,10,06
```

```
1590 data 4E,75,33,FC,00,4E,00,00,0D,D
2,33,FC,00,01,00,00
1600 data 0D,D4,33,FC,00,01,00,00,0D,D
6,33,FC,00,01,00,00
1610 data 0D,D8,33,FC,00,00,00,00,0D,D
A,33,C0,00,00,0E,06
1620 data 23,C8,00,00,12,06,61,00,FC,7
8,30,39,00,00,10,06
1630 data 4E,75,33,FC,00,5A,00,00,0D,D
2,33,FC,00,00,00,00
1640 data 0D,D4,33,FC,00,02,00,00,0D,D
6,33,FC,00,02,00,00
1650 data 0D,D8,33,FC,00,00,00,00,0D,D
A,23,C8,00,00,12,06
1660 data 23,C9,00,00,12,0A,61,00,FC,3
8,30,39,00,00,10,08
1670 data 4E,75,33,FC,00,6E,00,00,0D,D
2,33,FC,00,00,00,00
1680 data 0D,D4,33,FC,00,01,00,00,0D,D
6,33,FC,00,01,00,00
1690 data 0D,D8,33,FC,00,00,00,00,0D,D
A,23,FC,00,00,0B,6C
1700 data 00,00,12,06,61,00,FB,FA,30,3
9,00,00,10,06,4E,75
1710 data 33,FC,00,6F,00,00,0D,D2,33,F
C,00,00,00,00,0D,D4
1720 data 33,FC,00,01,00,00,0D,D6,33,F
C,00,00,00,00,0D,D8
1730 data 33,FC,00,00,00,00,0D,DA,61,0
0,FB,C6,30,39,00,00
1740 data 10,06,4E,75,33,FC,00,70,00,0
0,0D,D2,33,FC,00,02
1750 data 00,00,0D,D4,33,FC,00,01,00,0
0,0D,D6,33,FC,00,00
1760 data 00,00,0D,D8,33,FC,00,01,00,0
0,0D,DA,33,FC,00,00
1770 data 00,00,0E,06,33,FC,00,00,00,0
0,0E,08,61,00,FB,82
1780 data 30,39,00,00,10,06,23,F9,00,0
0,13,06,00,00,0B,64
1790 data 4E,75,61,00,FC,1C,33,C0,00,0
0,09,EA,61,00,FE,80
1800 data 33,C0,00,00,09,E6,33,C0,00,0
0,09,E8,61,00,FB,6E
1810 data 3F,3C,00,19,4E,41,54,8F,06,4
0,00,41,20,7C,00,00
1820 data 0A,90,10,C0,10,FC,00,3A,10,B
C,00,00,3F,3C,00,00
1830 data 2F,3C,00,00,0A,92,3F,3C,00,4
7,4E,41,50,8F,20,7C
1840 data 00,00,0A,90,22,7C,00,00,0B,3
0,61,00,02,36,30,3C
1850 data 00,00,61,00,FE,5E,61,00,FE,D
A,4A,40,66,00,00,16
1860 data 30,3C,00,01,20,7C,00,00,0B,7
8,61,00,FD,72,58,8F
1870 data 60,00,00,20,61,00,FF,2E,20,7
9,00,00,0B,64,30,3C
1880 data 00,02,C0,FC,00,18,D0,C0,20,6
8,00,0C,23,D0,00,00
1890 data 0B,68,4E,75,61,00,FF,5C,61,0
0,00,08,61,00,FE,D2
1900 data 4E,75,61,00,FD,7A,30,3C,00,0
0,61,00,FC,B0,30,3C
1910 data 00,01,61,00,FC,A8,33,FC,00,0
0,00,00,09,F0,61,00
1920 data FB,DE,61,00,FC,52,0C,40,00,0
4,66,00,00,08,61,00
1930 data 00,62,60,EA,0C,40,00,03,66,0
0,00,08,61,00,00,6A
1940 data 60,DC,0C,40,00,08,67,00,00,2
4,0C,40,00,09,66,00
1950 data 00,0C,61,00,00,18,61,00,01,C
4,60,A6,0C,40,00,07
1960 data 66,C0,61,00,00,08,61,00,02,B
0,60,96,30,3C,00,02
1970 data 61,00,FC,4A,30,3C,00,03,61,0
0,FC,42,4E,75,20,79
1980 data 00,00,0B,64,C0,FC,00,18,D0,C
0,02,68,FF,FE,00,0A
1990 data 4E,75,30,39,00,00,09,EE,06,4
0,00,01,0C,40,00,61
2000 data 6E,00,00,1E,60,00,00,10,30,3
9,00,00,09,EE,04,40
2010 data 00,01,67,00,00,0C,33,C0,00,0
0,09,EE,61,00,00,AC
2020 data 33,FC,00,02,00,00,09,F0,4E,7
5,3F,3C,00,03,3F,3C
2030 data 00,01,4E,4D,58,8F,4A,40,4E,7
5,33,FC,00,00,00,00
2040 data 09,EC,61,E6,66,00,00,28,06,7
9,00,01,00,00,09,EC
2050 data 0C,79,4E,20,00,00,09,EC,6D,E
8,30,3C,00,01,20,7C
2060 data 00,00,0C,1D,61,00,FC,38,58,8
F,60,00,00,12,3F,3C
2070 data 00,03,3F,3C,00,02,4E,4D,58,8
F,02,40,00,FF,4E,75
2080 data 02,40,00,FF,3F,00,3F,3C,00,0
3,3F,3C,00,08,4E,4D
2090 data 58,8F,4A,40,67,F0,3F,3C,00,0
3,3F,3C,00,03,4E,4D
2100 data 5C,8F,4E,75,61,84,67,00,00,0
6,61,8E,60,F6,4E,75
2110 data 30,3C,00,01,20,7C,00,00,0B,E
1,61,00,FB,E2,0C,40
2120 data 00,01,67,00,00,04,58,8F,4E,7
5,20,79,00,00,0B,68
2130 data 30,39,00,00,09,EE,56,88,32,3
C,00,01,48,C0,81,FC
2140 data 00,0A,48,40,11,00,06,10,00,3
0,48,40,51,C9,FF,EE
2150 data 4E,75,13,FC,00,00,00,00,0A,D
0,20,7C,00,00,0A,90
2160 data 22,7C,00,00,0A,D0,61,00,FC,A
A,4A,40,66,00,00,08
2170 data 58,8F,60,00,00,2C,0C,39,00,0
0,00,00,0A,D0,67,D2
2180 data 20,7C,00,00,0A,90,22,7C,00,0
0,0A,E0,61,00,00,20
2190 data 20,7C,00,00,0A,E0,22,7C,00,0
0,0A,D0,61,00,00,04
2200 data 4E,75,4A,18,66,FC,53,88,10,D
9,66,FC,4E,75,2F,09
2210 data 42,41,42,42,10,18,12,C0,67,0
0,00,18,0C,00,00,3A
2220 data 67,00,00,0A,0C,00,00,5C,66,0
0,00,04,34,01,52,41
2230 data 60,E2,20,5F,11,BC,00,00,20,0
1,4E,75,30,3C,00,09
2240 data 61,00,FE,5C,61,00,FF,6C,3F,3
C,00,00,2F,3C,00,00
2250 data 0A,E0,3F,3C,00,3D,4E,41,50,8
F,33,C0,00,00,0A,8E
2260 data 4A,40,6A,00,00,14,30,3C,00,0
1,20,7C,00,00,0B,9F
2270 data 61,00,FA,EC,60,00,00,C0,2F,3
C,00,00,0A,0E,2F,3C
2280 data 00,00,00,80,3F,39,00,00,0A,8
E,3F,3C,00,3F,4E,41
2290 data DF,FC,00,00,00,0C,3F,39,00,0
0,0A,8E,3F,3C,00,3E
2300 data 4E,41,58,8F,61,00,FE,BE,30,3
9,00,00,09,EE,04,40
2310 data 00,01,13,C0,00,00,0A,07,26,7
C,00,00,0A,01,36,3C
2320 data 00,06,10,1B,61,00,FE,7A,51,C
B,FF,F8,36,3C,00,05
2330 data 26,7C,00,00,0A,08,61,00,FE,2
2,B0,1B,67,00,00,08
2340 data 61,00,FE,8E,60,BE,51,CB,FF,E
E,26,7C,00,00,0A,0E
2350 data 36,3C,00,7F,18,1B,10,04,E8,0
8,61,00,FE,44,10,04
2360 data 02,00,00,0F,61,00,FE,3A,51,C
B,FF,EA,10,3C,00,F7
```

```
2370 data 61,00,FE,2E,61,00,FD,E4,0C,0
0,00,F7,67,00,00,0A
2380 data 61,00,FE,4E,60,00,FF,7E,30,3
C,00,01,20,7C,00,00
2390 data 0C,EB,61,00,FA,2A,4E,75,30,3
C,00,07,61,00,FD,60
2400 data 61,00,FE,22,30,39,00,00,09,E
E,04,40,00,01,13,C0
2410 data 00,00,0A,00,26,7C,00,00,09,F
A,36,3C,00,06,10,1B
2420 data 61,00,FD,DE,51,CB,FF,F8,36,3
C,00,05,26,7C,00,00
2430 data 0A,08,61,00,FD,86,B0,1B,67,0
0,00,08,61,00,FD,F2
2440 data 60,B6,51,CB,FF,EE,10,3C,00,7
0,61,00,FD,B4,10,3C
2450 data 00,31,61,00,FD,AC,36,3C,00,7
F,26,7C,00,00,0A,0E
2460 data 61,00,FD,58,18,00,61,00,FD,5
2,E9,0C,D0,04,16,C0
2470 data 51,CB,FF,EE,10,3C,00,F7,61,0
0,FD,86,61,00,FD,F4
2480 data 3F,3C,00,00,2F,3C,00,00,0A,E
0,3F,3C,00,3D,4E,41
2490 data 50,8F,4A,40,6B,00,00,20,3F,0
0,3F,3C,00,3E,4E,41
2500 data 58,8F,30,3C,00,02,20,7C,00,0
0,0C,96,61,00,F9,70
2510 data 0C,40,00,02,67,C6,3F,3C,00,0
0,2F,3C,00,00,0A,E0
2520 data 3F,3C,00,3C,4E,41,50,8F,33,C
0,00,00,0A,8E,4A,40
2530 data 6A,00,00,14,30,3C,00,01,20,7
C,00,00,0B,C0,61,00
2540 data F9,3E,60,00,FF,12,2F,3C,00,0
0,0A,0E,2F,3C,00,00
2550 data 00,80,3F,39,00,00,0A,8E,3F,3
C,00,40,4E,41,DF,FC
2560 data 00,00,00,0C,2F,00,3F,39,00,0
0,0A,8E,3F,3C,00,3E
2570 data 4E,41,58,8F,20,1F,0C,80,00,0
0,00,80,67,00,00,14
2580 data 30,3C,00,01,20,7C,00,00,0B,C
0,61,00,F8,F2,60,00
2590 data 00,10,30,3C,00,01,20,7C,00,0
0,0D,17,61,00,F8,E0
2600 data 4E,75,00,00,00,00,00,00,00,0
0,00,01,00,00,00,00
2610 data 00,00,00,00,00,00,F0,44,00,0
0,70,10,00,F0,44,00
2620 data 00,70,20,00,F0,44,00,00,70,3
0,00,00,00,00,00,00
2630 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2640 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2650 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2660 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2670 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2680 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2690 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2700 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2710 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2720 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2730 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2740 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2750 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2760 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2770 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2780 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2790 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
2800 data 00,00,00,00,00,00,00,00,00,0
0,00,00,5C,2A,2E,53
2810 data 4E,44,00,00,00,00,0D,D2,00,0
0,0D,E8,00,00,0E,06
2820 data 00,00,10,06,00,00,12,06,00,0
0,13,06,00,00,0D,D2
2830 data 00,00,0E,06,00,00,0F,06,00,0
0,10,06,00,00,11,06
2840 data 00,00,00,00,00,00,00,00,43,5
A,50,41,54,43,48,2E
2850 data 52,53,43,00,5B,33,5D,5B,55,6
E,61,62,6C,65,20,74
2860 data 6F,20,6C,6F,61,64,20,43,5A,5
0,41,54,43,48,2E,52
2870 data 53,43,5D,5B,41,42,4F,52,54,5
D,00,5B,33,5D,5B,55
2880 data 6E,61,62,6C,65,20,74,6F,20,6
C,6F,61,64,20,70,61
2890 data 74,63,68,2E,5D,5B,20,4F,4B,2
0,5D,00,5B,33,5D,5B
2900 data 55,6E,61,62,6C,65,20,74,6F,2
0,73,61,76,65,20,70
2910 data 61,74,63,68,2E,5D,5B,20,4F,4
B,20,5D,00,5B,33,5D
2920 data 5B,55,6E,61,62,6C,65,20,74,6
F,20,65,73,74,61,62
2930 data 6C,69,73,68,7C,68,61,6E,64,7
3,68,61,6B,65,20,77
2940 data 69,74,68,20,4D,49,44,49,2E,5
D,5B,52,45,54,52,59
2950 data 7C,43,41,4E,43,45,4C,5D,00,5
B,33,5D,5B,54,68,65
2960 data 20,4D,49,44,49,20,69,73,20,6
E,6F,74,20,72,65,73
2970 data 70,6F,6E,64,69,6E,67,21,7C,2
0,08,20,49,73,20,74
2980 data 68,65,20,4D,49,44,49,20,74,7
5,72,6E,65,64,20,6F
2990 data 6E,3F,7C,20,08,20,49,73,20,7
4,68,65,20,63,68,61
3000 data 6E,6E,65,6C,20,73,65,74,20,7
4,6F,20,31,3F,7C,20
3010 data 08,20,49,73,20,74,68,65,20,7
4,6F,6E,65,20,76,61
3020 data 6C,75,65,20,76,61,6C,69,64,3
F,5D,5B,20,4F,4B,20
3030 data 5D,00,5B,33,5D,5B,54,68,65,2
0,66,69,6C,65,6E,61
3040 data 6D,65,20,73,65,6C,65,63,74,6
5,64,7C,61,6C,72,65
3050 data 61,64,79,20,65,78,69,73,74,7
3,21,20,20,44,6F,20
3060 data 79,6F,75,7C,77,61,6E,74,20,7
4,6F,20,6F,76,65,72
3070 data 77,72,69,74,65,20,69,74,3F,5
D,5B,20,59,45,53,20
3080 data 7C,20,4E,4F,20,5D,00,5B,31,5
D,5B,50,61,74,63,68
3090 data 20,73,75,63,63,65,73,73,66,7
5,6C,6C,79,7C,73,65
3100 data 6E,74,20,74,6F,20,4D,49,44,4
9,2E,5D,5B,20,4F,4B
3110 data 20,5D,00,5B,31,5D,5B,50,61,7
4,63,68,20,73,75,63
3120 data 63,65,73,73,66,75,6C,6C,79,7
C,72,65,63,65,69,76
3130 data 65,64,20,66,72,6F,6D,20,4D,4
9,44,49,2E,5D,5B,20
3140 data 4F,4B,20,5D,00,00,00,01,00,0
1,00,01,00,01,00,01
```

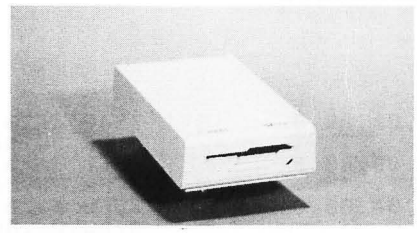
WHEN YOU WANT TO TALK ATARI...YOU WANT TO TALK TO US!



ATARI 520 System

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

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



ATARI 354 Disk Drive

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

ATARI 1040 System

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

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

CALL FOR PRICING ON THE NEW ATARI COMPONENTS

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

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

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

PERIPHERALS

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

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

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

ACCESSORIES

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

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

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

COMPUTER MAIL ORDER

In the U.S.A. and Canada

Call toll-free 1-800-233-8950

Outside the U.S.A. & Canada 717 327-9575 Telex 5106017898
Educational Institutions call toll-free 1-800-221-4283
CMO. 477 East Third Street, Dept. B805, Williamsport PA 17701
All major credit cards accepted.

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

CIRCLE #117 ON READER SERVICE CARD

```
3150 data 00,01,00,01,00,01,00,01,00,0
1,00,02,00,00,00,00
3160 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3170 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3180 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3190 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3200 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3210 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3220 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3230 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3240 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3250 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3260 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3270 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3280 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3290 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3300 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3310 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3320 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3330 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3340 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3350 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3360 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3370 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3380 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3390 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3400 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3410 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3420 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3430 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3440 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3450 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3460 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3470 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3480 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3490 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3500 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3510 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3520 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3530 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3540 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3550 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3560 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3570 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3580 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3590 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3600 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3610 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3620 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3630 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3640 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3650 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3660 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3670 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3680 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3690 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3700 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3710 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3720 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3730 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3740 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3750 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3760 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3770 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3780 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3790 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3800 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3810 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3820 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3830 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3840 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3850 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3860 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3870 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3880 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3890 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3900 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3910 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3920 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
```

```
3930 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3940 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3950 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3960 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3970 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3980 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3990 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4000 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4010 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4020 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4030 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4040 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4050 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4060 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4070 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4080 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4090 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4100 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4110 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4120 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4130 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4140 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4150 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4160 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4170 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4180 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4190 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4200 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4210 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4220 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4230 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4240 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4250 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4260 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4270 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4280 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4290 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4300 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4310 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4320 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4330 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4340 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4350 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4360 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4370 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4380 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4390 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4400 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4410 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4420 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4430 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4440 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4450 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4460 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4470 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4480 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4490 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4500 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4510 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4520 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4530 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4540 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4550 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4560 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4570 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4580 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4590 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4600 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4610 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4620 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4630 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4640 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4650 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4660 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4670 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4680 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4690 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4700 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
```

```
4710 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4720 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4730 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4740 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4750 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4760 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4770 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4780 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4790 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4800 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4810 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4820 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4830 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4840 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4850 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4860 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4870 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4880 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4890 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4900 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4910 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4920 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4930 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4940 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4950 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4960 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4970 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4980 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,04
4990 data 42,0E,10,08,08,06,04,06,06,1
2,06,10,10,04,0A,08
5000 data 08,06,04,10,0A,0A,0A,08,08,0
8,08,08,0A,04,0A,08
5010 data 08,08,08,0E,08,08,08,08,06,0
4,08,06,04,06,04,06
5020 data 04,06,04,06,04,0A,0A,08,08,0
8,08,08,06,04,0A,0A
5030 data 08,08,08,08,08,06,08,08,08,08,0
6,04,06,04,06,04,06
5040 data 04,0A,0A,08,08,08,08,06,06,0
A,0A,08,08,08,08,06
5050 data 04,0A,04,06,04,06,04,06,04,0
A,08,08,08,08,0A,0A
5060 data 08,08,08,08,06,06,0A,0A,08,0
8,08,08,06,06,0A,0A
5070 data 08,08,08,08,06,04,0A,0A,08,0
8,08,08,0A,0A,08,08
5080 data 08,08,08,08,0A,06,04,0C,0A,0
6,16,14,0E,06,20,14
5090 data 14,2C,66,14,16,0E,0C,1A,0E,0
8,0C,56,16,06,24,06
5100 data 06,18,08,06,0A,06,56,0E,10,0
E,0C,12,12,0A,06,18
5110 data 1A,42,18,0A,06,18,2E,2A,22,1
4,0E,10,0E,0C,14,1E
5120 data 12,01,5E,04,04,04,04,04,04,0
4,04,04,04,00
5130 data *
```

ST CHECKSUM DATA.

(see page 8)

```
 100 data  599, 948, 117, 614, 503, 22
7, 410, 427, 14, 109, 3968
 190 data  659, 357, 562, 578, 780, 77
5, 796, 665, 774, 779, 6725
 1080 data  839, 864, 713, 895, 937, 9
44, 902, 827, 916, 743, 8580
 1180 data  862, 684, 824, 839, 898, 8
28, 832, 983, 854, 836, 8440
 1280 data  753, 775, 793, 780, 751, 8
20, 832, 847, 630, 800, 7781
 1380 data  840, 839, 755, 727, 810, 7
71, 748, 840, 845, 842, 8017
 1480 data  740, 813, 843, 843, 702, 8
59, 680, 904, 866, 848, 8098
 1580 data  775, 846, 837, 850, 727, 8
41, 843, 843, 733, 852, 8147
 1680 data  845, 911, 766, 875, 845, 8
64, 845, 844, 860, 763, 8418
 1780 data  654, 956, 915, 737, 832, 8
13, 673, 881, 811, 755, 8027
 1880 data  877, 925, 914, 836, 854, 7
21, 719, 728, 759, 873, 8206
 1980 data  18, 737, 675, 752, 811, 79
0, 804, 832, 775, 821, 7015
 2080 data  790, 854, 888, 824, 698, 8
61, 819, 780, 834, 778, 8126
 2180 data  709, 741, 996, 674, 630, 7
79, 864, 872, 701, 877, 7843
 2280 data  740, 856, 896, 666, 941, 7
66, 65, 802, 989, 910, 7631
 2380 data  843, 924, 781, 702, 8, 911
, 58, 833, 955, 151, 6166
 2480 data  804, 786, 791, 787, 838, 6
93, 840, 894, 723, 722, 7878
 2580 data  816, 752, 533, 662, 614, 4
76, 477, 478, 479, 480, 5767
 2680 data  481, 482, 476, 477, 478, 4
79, 480, 481, 482, 483, 4799
 2780 data  484, 485, 581, 729, 595, 5
56, 639, 836, 828, 852, 6585
 2880 data  866, 879, 837, 860, 863, 9
07, 849, 867, 827, 845, 8600
 2980 data  845, 818, 837, 782, 858, 8
51, 868, 772, 900, 818, 8349
 3080 data  836, 876, 856, 796, 892, 8
36, 574, 472, 466, 467, 7071
 3180 data  468, 469, 463, 464, 465, 4
66, 467, 468, 469, 470, 4669
 3280 data  471, 472, 466, 467, 468, 4
69, 470, 471, 472, 473, 4699
 3380 data  474, 475, 469, 470, 471, 4
72, 473, 474, 475, 476, 4729
 3480 data  477, 478, 472, 473, 474, 4
75, 476, 477, 478, 479, 4759
 3580 data  480, 481, 475, 476, 477, 4
78, 479, 480, 481, 482, 4789
 3680 data  483, 484, 478, 479, 480, 4
81, 482, 483, 484, 485, 4819
 3780 data  486, 487, 481, 482, 483, 4
84, 485, 486, 487, 488, 4849
 3880 data  489, 490, 484, 485, 486, 4
87, 488, 489, 490, 491, 4879
 3980 data  492, 493, 459, 460, 461, 4
62, 463, 464, 465, 466, 4685
 4080 data  467, 468, 462, 463, 464, 4
65, 466, 467, 468, 469, 4659
```

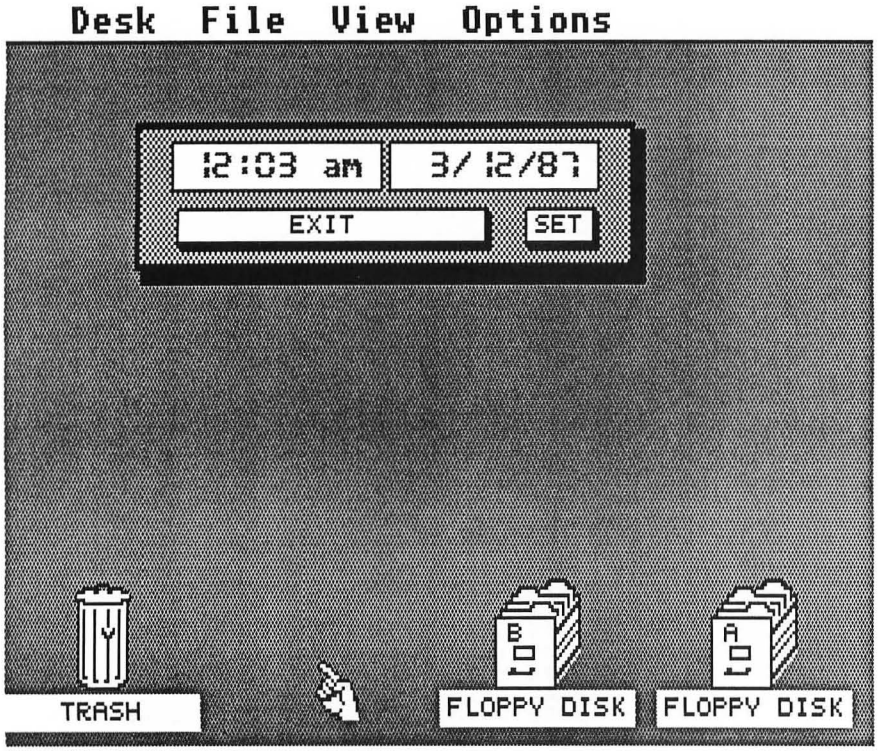
```
 4180 data  470, 471, 465, 466, 467, 4
68, 469, 470, 471, 472, 4689
 4280 data  473, 474, 468, 469, 470, 4
71, 472, 473, 474, 475, 4719
 4380 data  476, 477, 471, 472, 473, 4
74, 475, 476, 477, 478, 4749
 4480 data  479, 480, 474, 475, 476, 4
77, 478, 479, 480, 481, 4779
 4580 data  482, 483, 477, 478, 479, 4
80, 481, 482, 483, 484, 4809
 4680 data  485, 486, 480, 481, 482, 4
83, 484, 485, 486, 487, 4839
 4780 data  488, 489, 483, 484, 485, 4
86, 487, 488, 489, 490, 4869
 4880 data  491, 492, 486, 487, 488, 4
89, 490, 491, 492, 493, 4899
 4980 data  498, 617, 619, 583, 605, 5
70, 619, 604, 623, 626, 5964
 5080 data  626, 702, 647, 704, 988, 2
01, 3868
```

Listing 2.
ST BASIC listing.

```
100 filename$="a:\czpatch.rsc"
110 fullw 2:clearw 2:gotoxy 0,0:print
"creating file..."
120 option base 0
125 dim a%(16000):def seg=1:v$=""
130 p=varptr(a%(0)):bptr=p+1
140 for i%=1 to 552
150 read v$:code%=val("&H"+v$)
160 poke p, code%:print ".";
170 p=p+1
180 next
190 bsave filename$,bptr,552
200 print "file written":end
1000 data 00,00,01,1C,00,74,00,74,00,7
4,00,74,00,24,00,74
1010 data 00,74,02,24,00,0B,00,01,00,0
6,00,00,00,00,00,00
1020 data 00,00,02,28,20,30,31,20,00,0
0,54,6F,6E,65,20,4E
1030 data 75,6D,62,65,72,00,00,20,20,5
2,65,63,65,69,76,65
1040 data 20,20,00,00,20,20,20,51,55,4
9,54,20,20,20,20,00
1050 data 00,20,20,20,53,65,6E,64,20,2
0,20,20,00,00,20,43
1060 data 5A,20,50,61,74,63,68,20,4C,6
9,62,72,61,72,69,61
1070 data 6E,20,00,00,00,00,00,24,00,0
0,00,00,00,00,00,29
1080 data 00,03,00,06,00,00,11,80,00,0
0,00,FF,00,05,00,01
1090 data 00,00,00,2A,00,00,00,00,00,0
0,00,36,00,03,00,06
1100 data 00,00,11,80,00,00,00,FF,00,0
C,00,01,00,00,00,37
1110 data 00,00,00,00,00,00,00,43,00,0
3,00,06,00,00,11,80
1120 data 00,00,00,FF,00,0C,00,01,00,0
0,00,44,00,00,00,00
1130 data 00,00,00,50,00,03,00,06,00,0
0,11,80,00,00,00,FF
1140 data 00,0C,00,01,00,00,00,51,00,0
0,00,00,00,00,00,5D
1150 data 00,03,00,06,00,00,11,80,00,0
0,00,FF,00,0C,00,01
1160 data 00,00,00,5E,00,00,00,00,00,0
0,00,73,00,03,00,06
1170 data 00,00,11,80,00,00,00,FF,00,1
5,00,01,FF,FF,00,01
1180 data 00,0A,00,14,00,00,00,20,00,0
2,11,11,00,00,00,00
1190 data 00,20,00,11,00,06,00,02,00,0
5,00,14,00,00,00,20
1200 data 00,FF,11,00,00,08,00,04,00,0
F,00,04,00,03,FF,FF
1210 data FF,FF,00,16,00,00,00,00,00,0
0,00,74,00,05,00,02
1220 data 00,04,00,01,00,04,FF,FF,FF,F
F,00,1B,00,40,00,00
1230 data 02,FF,11,00,00,01,00,02,00,0
4,00,01,00,05,FF,FF
1240 data FF,FF,00,1B,00,40,00,00,01,F
F,11,00,00,09,00,02
1250 data 00,04,00,01,00,01,FF,FF,FF,F
F,00,15,00,00,00,00
1260 data 00,00,00,90,00,02,00,00,00,0
B,00,01,00,0A,00,07
1270 data 00,09,00,14,00,00,00,20,00,F
F,11,00,00,08,00,09
1280 data 00,0F,00,07,00,08,FF,FF,FF,F
F,00,16,00,15,00,20
1290 data 00,00,00,AC,00,02,00,03,00,0
B,00,01,00,09,FF,FF
1300 data FF,FF,00,16,00,15,00,20,00,0
0,00,C8,00,02,00,05
1310 data 00,0B,00,01,00,06,FF,FF,FF,F
F,00,16,00,15,00,20
1320 data 00,00,00,E4,00,02,00,01,00,0
B,00,01,00,00,FF,FF
1330 data FF,FF,00,16,00,20,00,20,00,0
0,01,00,00,05,00,01
1340 data 00,14,00,01,00,00,01,1C
1350 data *
```

ST CHECKSUM DATA.
(see page 8)

```
 100 data  591, 948, 117, 614, 503, 65
, 410, 427, 14, 109, 3798
 190 data  497, 357, 611, 516, 662, 75
9, 587, 623, 798, 530, 5940
 1080 data  592, 509, 608, 509, 582, 5
99, 531, 606, 531, 765, 5832
 1180 data  504, 501, 768, 677, 855, 7
45, 776, 830, 537, 597, 6790
 1280 data  879, 747, 733, 863, 729, 6
69, 257, 201, 5078
```



Try a digital clock accessory as an

by John Brochu

One of the things that impressed me when I first heard about the Atari 520ST was the amount of RAM in the machine. After using the original 800 since 1982 with its then plentiful 48K, followed by the 130XE with its bank-switched 128K, 512-thousand bytes sounded like a lot of RAM!

Of course, the first release of the ST had to have TOS booted from disk, which ate up nearly half that RAM. With a full load of desk accessories, an application was sometimes left with less available RAM than the 8-bitters! When TOS on ROM came along, promising to free another 200K, the system was finally given some breathing room. Still, using the Atari Developer's Kit to write C programs almost requires the use of a large RAMdisk to make the compile and link process at least bearable. I found myself scrounging for every little byte I could save.

The Atari control panel is a convenient accessory, allowing you to tailor the system to your specific tastes. However, the price for this convenience is that it eats up over 17K of RAM—which could be put to better use. The only time I ever used it (after initial set-up) was to enter the time and date. But, since the control panel was also needed to set up the system defaults, I was stuck with it. So what I set out to do was write an accessory that would duplicate the most-used features of the control panel, while freeing up some of the unused RAM.

The new accessory, upon boot-up, loads the DESKTOP.INF file into a temporary buffer, reads the printer and control panel settings, and sets up the system defaults accordingly. All parameters the control panel handles will be set by the **ST Clock**. These include the color palette, mouse double-click speed, key-click and console bell toggles, key delay and repeat rates, and printer configuration.

Using the *ST Clock*.

The C source code for **ST Clock** is included on the disk version of this issue and on the Atari Users' Group on Delphi.

To install **ST Clock**, first load ST BASIC and type in Listing 1. Check it with **ST-Check**, then run it. The BASIC program will create the file CLOCK.ACC on drive A. Rename it DESK1.ACC if you still have the original TOS on disk.

Remember, this is to be used *in place of* the control panel, so delete or rename the panel accessory, if necessary.

Once the system is booted, the **ST Clock** can be displayed by selecting it from the "desk" menu on the desktop or GEM application. The clock can be set easily, by clicking on the time or date window, or on the SET button. In either case, the SET button will be drawn inverted, or "selected," and the text cursor will appear in the selected field.

The time and date can then be typed in using the GEM editor, which includes the following special keys: ESC - clear the field and position the cursor at the beginning of the field; TAB or DOWN-ARROW - move to the next field; UP-ARROW - move to the previous field; LEFT- or RIGHT-ARROW - move the cursor within the current field. BACKSPACE and DELETE can also be used to edit the field. After typing in the new time and/or date, pressing RETURN or clicking the STORE button will set the clock if the entered values are indeed valid time and date entries. If either is invalid, that field will be ignored.

Time can be entered in either 12- or 24-hour format, the latter being assumed if you leave off the am/pm specifier. If the entry is accepted, the hardware IKBD clock—as well as the GEMDOS clock—will be set, so that the time won't be lost on a RESET or resolution-change. Seconds are cleared when a new time setting is accepted, so that the

ST Clock

A SPECIAL INCLUSION
SOURCE CODE ON DISK AND
ON DELPHI ATARI USERS' GROUP

alternative to the control panel.

ST Clock can be synchronized with an accurate time source, if desired. Set mode can be aborted by clicking SET a second time to deselect it.

When not in SET mode, the accessory checks once a second to see if GEMDOS's time is different from the displayed time. If so, the time and date are updated from the GEMDOS clock, which, incidentally, has only a two second resolution. Therefore, the displayed time is always accurate to plus or minus two seconds, even after a SYSTEM RESET.

Some useful functions.

Most of the C functions in the **ST Clock** are structured so that you can use them in your own programs with little or no modification. Let's go through them now, one at a time.

The find() routine searches the string at address st_adr for a null-terminated string of any length. If the string is found between st_adr and end_adr, its start address is returned; otherwise, the null pointer is returned. The dpeek() routine will return the integer value found at any address in system memory.

The get_asc() routine converts an integer value to a string representing "normal" or LED-type digits. The following values are passed: num is the integer value to convert; string is a pointer to the output string; max is the maximum number of characters (*1* through *5*) allowed in the output string (the output string is right-justified in this field, and excess characters on the left are ignored); ofs is the offset to character *0* (should be *48 [0x30]* for "normal" ASCII digits, or *16 [0x10]* for LED-type digits); and typ specifies whether leading zeros are converted to spaces (*1*), or left as zeros (*0*).

The next routine, get_num(), performs the opposite operation, converting an ASCII string of the specified length to an integer. The functions copy_str() and str_len() are similar to their C-library equivalents, except that our copy_str() has its input arguments reversed (i.e., copy_str(from,to)).

The C listing.

The actual clock object is defined in the TEDINFO and OBJECT structures at the top of the source code listing (again, this is on disk and on Delphi, for those who are interested). These structures are declared in the obdefs.h header file supplied with the Developer's Kit, and were produced with the Resource Construction Set using the "C output" option.

Tracing through the main() listing, the first call is appl_init(), which supplies us with a unique application ID, gl_apid, used in the following menu_register call to display our accessory's title in the desk menu. (It should be noted here that the docs state an ap_id is returned by the appl_init() function, but due to a bug in the bindings, this is not the case. The application ID *can* be accessed directly from the system array with a gl_apid external declaration.)

Returned from the menu_register() call is a menu ID number, used later on to determine when our accessory is started from the desk menu. Next, we read the time and date from the IKBD's hardware clock. If the month and date are both zero here, we can assume the system has just been powered up and proceed to set the power-up default date (*MO*, *DAY* and *YR*) defined at the beginning of the listing. (These definitions may be changed to any desired date at compile time.)

An attempt is then made to open the DESKTOP.INF file; first on the current drive, then on the boot drive if they're not one and the same. If the file is found, a memory block is reserved with a Malloc() call, and the file is read into the allocated memory. The file is then processed in the

get_b_hdr() and get_c_hdr() functions, which parse the file's "#b" (printer configuration) and "#c" (control panel) headers for the system parameters. We then free the 1K buffer we Malloc()'d earlier, to complete system initialization.

After setting up the object structures, we enter the main accessory loop and make an evnt_mesag() call, to tell the AES that we're waiting for the user to open our accessory from the desk menu. The system transfers control to the desktop until this occurs. At this point, our application ID is moved from the "ready-list" to the "not-ready-list." The AES's dispatcher maintains these lists automatically. Each application, whether program, accessory or the desktop, is placed on the not-ready-list whenever a call is made to one of the AES's event library routines (evnt_mesag() in this case), and moved back to the ready-list when the requested event has occurred. Each of the processes on the ready-list, is in this way, handled in turn.

When the **ST Clock** is finally selected, we change the mouse pointer to a pointing hand and take over control of the mouse with a wind_update() call. After initializing some variables, the **ST Clock** is drawn and we enter the do_clock() function, where we continually monitor the user's interaction with the timekeeper. The form_do() call is only made when the clock is in SET mode. Otherwise, we use a custom routine (similar to form_do() itself), so the time can be kept up to date.

Once an object has been selected in the wait_button() function and its object number returned, we determine which object was selected and take the appropriate action. Clicking on the time or date will cause the SET button to become selected, change the EXIT button text to STORE, redraw the time and date in standard rather than LED digits, and switch to SET mode, using the standard form_do call to edit the time and date.

If the SET button was clicked, its state is toggled, the EXIT/STORE button text is changed, and we switch modes. If STORE was clicked, the set button is deselected, STORE is changed to EXIT, time and date are verified in the set_time() function, and the clock is set. (Note that both the GEMDOS and IKBD clocks are set with the new time/date. This protects us in case of a SYSTEM RESET or resolution change, since only GEMDOS's software-maintained clock is affected.) If, however, EXIT is clicked, we return to main() and "undraw" the clock, release control of the mouse, and return back to the evnt_mesag() call. This causes the AES to move our process back to the not-ready-list and pass control back to the main application or the desktop.

Another alternative.

Although there are several public domain clock accessories available, none that I've seen also set up the system defaults. This accessory does everything the control panel does, in a much smaller space. In tests I've done, the control panel uses about 17610 bytes, while the **ST Clock** uses only 8200.

The new accessory will use only about 3K and will be completely invisible, since it will only execute once—at boot-up or RESET—to read the DESKTOP file, then "go to sleep." The evnt_timer() call on the last line of the C source code listing is required. Without it, the accessory would never stop running and would appear to lock up the system. ///

John Brochu has been programming on Atari computers exclusively since 1982, beginning with the 400, then the 800 and 130XE, in BASIC and assembly language. Now, he's on to the 520ST, on which he's been using the Atari Developer's Kit since October. He also enjoys doing hardware upgrades and telecommunicating, and is a member of the Atari SIGs on CompuServe, Delphi and GEnie.

Listing 1.
ST BASIC listing.

```
100 filename$="a:\clock.acc"
110 fullw 2:clearw 2:gotoxy 0,0:print
"creating file..."
120 option base 0
125 dim a%(16000):def seg=1:v$=""
130 p=varptr(a%(0)):bptr=p+1
140 for i%=1 to 6774
150 read v$:code%=val("&H"+v$)
160 poke p, code%:print ".";
170 p=p+1
180 next
190 bsave filename$,bptr,6774
200 print "file written":end
1000 data 60,1A,00,00,13,28,00,00,02,E
A,00,00,08,B0,00,00
1010 data 03,48,00,00,00,00,00,00,00,0
0,00,00,2E,7C,00,00
1020 data 1A,16,4E,B9,00,00,02,2E,2E,B
C,00,00,00,00,4E,41
1030 data 22,2F,00,04,30,3C,00,C8,4E,4
2,4E,75,4E,56,FF,F8
1040 data 42,6E,FF,FE,60,04,52,6E,FF,F
E,20,6E,00,10,32,6E
1050 data FF,FE,D1,C9,1D,50,FF,FC,67,1
4,30,6E,FF,FE,22,6E
1060 data 00,08,10,30,98,00,48,80,B0,2
E,FF,FC,67,D8,4A,2E
1070 data FF,FC,67,0E,52,AE,00,08,20,2
E,00,08,B0,AE,00,0C
1080 data 65,BE,20,2E,00,08,B0,AE,00,0
C,66,06,42,80,60,06
1090 data 60,04,20,2E,00,08,4E,5E,4E,7
5,4E,56,FF,F6,42,97
1100 data 3F,3C,00,20,4E,B9,00,00,13,1
8,54,8F,2D,40,FF,FA
1110 data 20,6E,00,08,3D,50,FF,FE,2E,A
E,FF,FA,3F,3C,00,20
1120 data 4E,B9,00,00,13,18,54,8F,30,2
E,FF,FE,4E,5E,4E,75
1130 data 4E,56,FF,EC,41,EE,FF,FA,2D,4
8,FF,F6,42,6E,FF,F2
1140 data 3D,7C,27,10,FF,F0,42,6E,FF,F
4,60,0C,30,2E,FF,F0
1150 data 91,6E,00,08,52,6E,FF,F4,30,2
E,00,08,B0,6E,FF,F0
1160 data 6C,EA,4A,6E,FF,F4,66,0C,4A,6
E,FF,F2,66,06,4A,6E
1170 data 00,12,66,1C,52,6E,FF,F2,30,2
E,FF,F4,12,2E,00,11
1180 data 48,81,D0,41,22,6E,FF,F6,12,8
0,52,AE,FF,F6,60,0C
1190 data 20,6E,FF,F6,10,BC,00,20,52,A
E,FF,F6,30,2E,FF,F0
1200 data 48,C0,81,FC,00,0A,3D,40,FF,F
0,0C,6E,00,01,FF,F0
1210 data 6C,94,20,0E,72,05,92,6E,00,0
E,48,C1,D0,81,5D,80
```

```
1220 data 2D,40,FF,F6,3D,6E,00,0E,FF,F
2,60,16,20,6E,00,0A
1230 data 22,6E,FF,F6,10,91,52,AE,FF,F
6,52,AE,00,0A,53,6E
1240 data FF,F2,4A,6E,FF,F2,66,E4,4E,5
E,4E,75,4E,56,FF,F4
1250 data 42,6E,FF,FE,3D,7C,00,01,FF,F
C,30,2E,00,0C,53,40
1260 data 48,C0,D1,AE,00,08,42,6E,FF,F
A,60,44,20,6E,00,08
1270 data 1D,50,FF,F8,53,AE,00,08,0C,2
E,00,30,FF,F8,6D,10
1280 data 0C,2E,00,39,FF,F8,6E,08,04,2
E,00,30,FF,F8,60,02
1290 data 60,1A,10,2E,FF,F8,48,80,C0,E
E,FF,FC,D1,6E,FF,FE
1300 data 30,2E,FF,FC,C0,FC,00,0A,3D,4
0,FF,FC,52,6E,FF,FA
1310 data 30,2E,FF,FA,B0,6E,00,0C,6D,B
2,42,40,30,2E,FF,FE
1320 data 4E,5E,4E,75,4E,56,FF,FC,20,6
E,00,08,10,10,48,80
1330 data 22,6E,00,0C,12,80,52,AE,00,0
8,52,AE,00,0C,4A,40
1340 data 66,E6,4E,5E,4E,75,4E,56,FF,F
A,42,6E,FF,FE,60,04
1350 data 52,6E,FF,FE,30,6E,FF,FE,22,6
E,00,08,4A,30,98,00
1360 data 66,EE,30,2E,FF,FE,4E,5E,4E,7
5,4E,56,FF,CA,2D,7C
1370 data 00,00,04,46,FF,CE,4E,B9,00,0
0,0F,80,2E,BC,00,00
1380 data 15,FB,3F,39,00,00,1E,90,4E,B
9,00,00,12,38,54,8F
1390 data 3D,40,FF,F6,3E,BC,00,17,4E,B
9,00,00,12,F8,2D,40
1400 data FF,D2,20,2E,FF,D2,C0,BC,01,F
F,00,00,66,18,2D,7C
1410 data 0C,21,00,00,FF,D2,2E,AE,FF,D
2,3F,3C,00,16,4E,B9
1420 data 00,00,12,F8,54,8F,20,2E,FF,D
2,C0,BC,FF,FF,00,00
1430 data 72,10,E2,A0,3D,40,FF,E2,20,2
E,FF,D2,3D,40,FF,E4
1440 data 3E,AE,FF,E2,3F,3C,00,2B,4E,B
9,00,00,13,18,54,8F
1450 data 3E,AE,FF,E4,3F,3C,00,2D,4E,B
9,00,00,13,18,54,8F
1460 data 3E,BC,00,19,4E,B9,00,00,13,1
8,D0,7C,00,41,13,C0
1470 data 00,00,15,8C,42,57,2F,3C,00,0
0,15,8C,3F,3C,00,3D
1480 data 4E,B9,00,00,13,18,5C,8F,2D,4
0,FF,D6,6C,40,2E,AE
1490 data FF,CE,61,00,FD,86,D0,7C,00,4
1,3D,40,FF,FE,10,39
1500 data 00,00,15,8C,48,80,B0,6E,FF,F
E,67,22,30,2E,FF,FE
1510 data 13,C0,00,00,15,8C,42,57,2F,3
C,00,00,15,8C,3F,3C
1520 data 00,3D,4E,B9,00,00,13,18,5C,8
F,2D,40,FF,D6,4A,AE
1530 data FF,D6,6D,00,00,96,20,2E,FF,D
6,3D,40,FF,FC,2E,BC
1540 data 00,00,04,00,3F,3C,00,48,4E,B
9,00,00,13,18,54,8F
1550 data 2D,40,FF,DE,2D,40,FF,D2,4A,A
E,FF,D2,66,12,3E,AE
1560 data FF,FC,3F,3C,00,3E,4E,B9,00,0
0,13,18,54,8F,60,5A
1570 data 2E,AE,FF,DE,2F,3C,00,00,04,0
0,3F,2E,FF,FC,3F,3C
1580 data 00,3F,4E,B9,00,00,13,18,50,8
F,2D,40,FF,DA,3E,AE
1590 data FF,FC,3F,3C,00,3E,4E,B9,00,0
0,13,18,54,8F,2E,AE
1600 data FF,DA,2F,2E,FF,DE,61,00,02,1
2,58,8F,2E,AE,FF,DA
1610 data 2F,2E,FF,DE,61,00,02,8E,58,8
F,2E,AE,FF,D2,3F,3C
1620 data 00,49,4E,B9,00,00,13,18,54,8
F,3D,7C,00,08,FF,FA
1630 data 3E,BC,00,04,4E,B9,00,00,12,F
8,B0,BC,00,00,00,02
1640 data 67,04,42,40,60,02,70,01,E7,4
0,50,40,3D,40,FF,F8
1650 data 42,6E,FF,FE,60,74,30,2E,FF,F
E,C1,FC,00,18,D0,BC
1660 data 00,00,15,14,20,40,30,28,00,1
0,C1,EE,FF,FA,31,40
1670 data 00,10,30,2E,FF,FE,C1,FC,00,1
8,D0,BC,00,00,15,14
1680 data 20,40,30,28,00,12,C1,EE,FF,F
8,31,40,00,12,30,2E
1690 data FF,FE,C1,FC,00,18,D0,BC,00,0
0,15,14,20,40,30,28
1700 data 00,14,C1,EE,FF,FA,31,40,00,1
4,30,2E,FF,FE,C1,FC
1710 data 00,18,D0,BC,00,00,15,14,20,4
0,30,28,00,16,C1,EE
1720 data FF,F8,31,40,00,16,52,6E,FF,F
E,0C,6E,00,05,FF,FE
1730 data 6D,84,30,2E,FF,FA,C1,FC,00,0
6,33,C0,00,00,15,24
1740 data 33,C0,00,00,1E,B4,30,2E,FF,F
8,E5,40,33,C0,00,00
1750 data 15,26,33,C0,00,00,1E,B6,30,3
9,00,00,15,28,5A,40
1760 data 33,C0,00,00,1E,B2,30,39,00,0
0,15,2A,5A,40,33,C0
1770 data 00,00,1E,8E,60,00,00,FC,2E,8
E,06,97,FF,FF,FF,E6
1780 data 4E,B9,00,00,10,4E,0C,6E,00,2
8,FF,E6,66,00,00,E4
1790 data 30,2E,FF,EE,B0,6E,FF,F6,66,0
0,00,D8,42,97,3F,3C
1800 data 00,03,4E,B9,00,00,12,16,54,8
F,3E,BC,00,03,4E,B9
1810 data 00,00,12,DE,3E,B9,00,00,1E,8
E,3F,39,00,00,1E,B2
1820 data 3F,39,00,00,1E,B6,3F,39,00,0
0,1E,B4,42,67,42,67
1830 data 42,67,42,67,42,67,4E,B9,00,0
0,11,8A,DF,FC,00,00
1840 data 00,10,33,FC,00,40,00,00,15,3
4,33,FC,00,40,00,00
1850 data 15,4C,2E,BC,00,00,1E,B8,2F,3
C,00,00,14,98,61,00
1860 data FC,A4,58,8F,70,10,48,80,3E,8
0,61,00,06,CA,3E,BC
1870 data 00,20,42,67,61,00,04,D2,54,8
F,61,00,03,0A,3E,BC
1880 data 00,20,3F,3C,00,03,61,00,04,C
0,54,8F,3E,BC,00,02
1890 data 4E,B9,00,00,12,DE,3E,B9,00,0
0,1E,8E,3F,39,00,00
1900 data 1E,B2,3F,39,00,00,1E,B6,3F,3
9,00,00,1E,B4,42,67
1910 data 42,67,42,67,42,67,3F,3C,00,0
3,4E,B9,00,00,11,8A
1920 data DF,FC,00,00,00,10,42,97,42,6
7,4E,B9,00,00,12,16
1930 data 54,8F,60,00,FF,04,4E,5E,4E,7
5,4E,56,FF,F0,42,6E
1940 data FF,F8,20,2E,00,08,D0,AE,00,0
C,2D,40,FF,F4,2E,BC
1950 data 00,00,16,0B,2F,2E,FF,F4,2F,2
E,00,08,61,00,FA,3E
1960 data 50,8F,2D,40,FF,FC,4A,AE,FF,F
C,63,54,54,AE,FF,FC
1970 data 60,04,52,AE,FF,FC,20,6E,FF,F
C,0C,10,00,20,67,F2
1980 data 42,6E,FF,FA,60,22,E2,EE,FF,F
8,20,6E,FF,FC,10,10
1990 data 48,80,D0,7C,FF,D0,C0,7C,00,0
1,EB,40,81,6E,FF,F8
```

```
2000 data 52,AE,FF,FC,52,6E,FF,FA,0C,6
E,00,06,FF,FA,6D,D6
2010 data 3E,AE,FF,F8,3F,3C,00,21,4E,B
9,00,00,12,F8,54,8F
2020 data 4E,5E,4E,75,4E,56,FF,B6,2D,7
C,00,00,04,84,FF,F2
2030 data 20,2E,00,08,D0,AE,00,0C,2D,4
0,FF,BA,2E,BC,00,00
2040 data 16,0E,2F,2E,FF,BA,2F,2E,00,0
8,61,00,F9,B0,50,8F
2050 data 2D,40,FF,F6,4A,AE,FF,F6,63,0
0,01,E8,54,AE,FF,F6
2060 data 60,04,52,AE,FF,F6,20,6E,FF,F
6,0C,10,00,20,67,F2
2070 data 3E,BC,00,04,4E,B9,00,00,12,F
8,3D,40,FF,EA,0C,6E
2080 data 00,02,FF,EA,6C,00,00,B2,42,6
E,FF,E8,60,6A,20,6E
2090 data FF,F6,10,10,48,80,D0,7C,FF,D
0,3D,40,FF,F0,52,AE
2100 data FF,F6,20,6E,FF,F6,10,10,48,8
0,D0,7C,FF,D0,3D,40
2110 data FF,EE,52,AE,FF,F6,20,6E,FF,F
6,10,10,48,80,D0,7C
2120 data FF,D0,3D,40,FF,EC,52,AE,FF,F
6,30,2E,FF,F0,E1,40
2130 data 32,2E,FF,EE,E9,41,D0,41,D0,6
E,FF,EC,22,4E,34,6E
2140 data FF,E8,D5,CA,D5,FC,00,00,15,9
C,34,52,D5,CA,D3,CA
2150 data 33,40,FF,C8,52,6E,FF,E8,0C,6
E,00,10,FF,E8,6D,8E
2160 data 0C,6E,00,01,FF,EA,66,06,3D,6
E,FF,E6,FF,CE,2E,8E
2170 data 06,97,FF,FF,FF,C8,3F,3C,00,0
6,4E,B9,00,00,12,F8
2180 data 54,8F,3E,BC,00,25,4E,B9,00,0
0,12,F8,3E,BC,00,25
2190 data 4E,B9,00,00,12,F8,60,08,06,A
E,00,00,00,30,FF,F6
2200 data 20,6E,FF,F6,10,10,48,80,D0,7
C,FF,D0,3D,40,FF,C6
2210 data 52,AE,FF,F6,20,6E,FF,F6,10,1
0,48,80,D0,7C,FF,D0
2220 data 1D,40,FF,FC,52,AE,FF,F6,20,6
E,FF,F6,10,10,48,80
2230 data D0,7C,FF,D0,1D,40,FF,FA,52,A
E,FF,F6,20,6E,FF,F6
2240 data 10,10,48,80,D0,7C,FF,D0,C1,F
C,00,0A,22,6E,FF,F6
2250 data 12,29,00,01,48,81,D0,41,D0,7
C,FF,D0,3D,40,FF,C4
2260 data 20,6E,FF,F6,10,28,00,02,48,8
0,D0,7C,FF,D0,C1,FC
2270 data 00,0A,22,6E,FF,F6,12,29,00,0
3,48,81,D0,41,D0,7C
2280 data FF,D0,3D,40,FF,C2,3E,AE,FF,C
2,3F,2E,FF,C4,3F,3C
2290 data 00,23,4E,B9,00,00,12,F8,58,8
F,3E,BC,00,01,3F,2E
2300 data FF,C6,4E,B9,00,00,11,46,54,8
F,42,97,3F,3C,00,20
2310 data 4E,B9,00,00,13,18,54,8F,2D,4
0,FF,BE,20,6E,FF,F2
2320 data 10,10,48,80,C0,7C,00,FA,1D,4
0,FF,FE,10,2E,FF,FA
2330 data 48,80,C0,7C,00,01,12,2E,FF,F
C,48,81,C2,7C,00,01
2340 data E5,41,80,41,12,2E,FF,FE,48,8
1,80,41,22,6E,FF,F2
2350 data 12,80,2E,AE,FF,BE,3F,3C,00,2
0,4E,B9,00,00,13,18
2360 data 54,8F,4E,5E,4E,75,4E,56,FF,F
0,60,00,01,60,2E,8E
2370 data 55,97,2F,0E,55,97,2F,0E,55,9
7,2F,0E,55,97,42,67
2380 data 3F,3C,00,01,3F,3C,00,01,4E,B
9,00,00,0F,F4,DF,FC
2390 data 00,00,00,12,08,39,00,00,00,0
0,15,7F,67,18,3E,AE
2400 data FF,FA,2F,3C,00,00,15,14,4E,B
9,00,00,11,68,58,8F
2410 data 3D,40,FF,FC,60,0E,2E,BC,00,0
0,15,14,61,00,04,B0
2420 data 3D,40,FF,FC,0C,6E,00,01,FF,F
C,67,08,0C,6E,00,02
2430 data FF,FC,66,38,3D,6E,FF,FC,FF,F
A,08,39,00,00,00,00
2440 data 15,7F,66,24,3E,BC,00,21,3F,3
C,00,04,61,00,01,3A
2450 data 54,8F,70,30,48,80,3E,80,3F,3
C,00,48,2F,3C,00,00
2460 data 14,9E,61,00,00,D0,5C,8F,60,0
0,00,C2,0C,6E,00,04
2470 data FF,FC,66,42,08,39,00,00,00,0
0,15,7F,67,1E,3D,7C
2480 data 00,01,FF,FA,70,30,48,80,3E,8
0,3F,3C,00,48,2F,3C
2490 data 00,00,14,9E,61,00,00,9E,5C,8
F,60,16,70,10,48,80
2500 data 3E,80,3F,3C,00,40,2F,3C,00,0
0,14,98,61,00,00,86
2510 data 5C,8F,60,00,00,78,0C,6E,00,0
3,FF,FC,66,6E,08,39
2520 data 00,00,00,00,15,7F,67,62,3E,B
C,00,20,3F,3C,00,04
2530 data 61,00,00,B6,54,8F,61,00,01,0
0,3E,BC,00,2C,4E,B9
2540 data 00,00,13,18,3D,40,FF,F8,3E,B
C,00,2A,4E,B9,00,00
2550 data 13,18,2D,40,FF,F4,20,2E,FF,F
4,72,10,E3,A0,42,81
2560 data 32,2E,FF,F8,D0,81,2E,80,3F,3
C,00,16,4E,B9,00,00
2570 data 12,F8,54,8F,70,10,48,80,3E,8
0,3F,3C,00,40,2F,3C
2580 data 00,00,14,98,61,0E,5C,8F,60,0
2,60,04,60,00,FE,A0
2590 data 4E,5E,4E,75,4E,56,FF,FC,2E,B
C,00,00,1E,B8,2F,2E
2600 data 00,08,61,00,F8,00,58,8F,3E,B
C,00,20,3F,3C,00,03
2610 data 61,36,54,8F,33,EE,00,0C,00,0
0,15,34,33,EE,00,0C
2620 data 00,00,15,4C,3E,AE,00,0E,61,0
0,02,0C,3E,BC,00,10
2630 data 3F,3C,00,01,61,12,54,8F,3E,B
C,00,10,3F,3C,00,02
2640 data 61,06,54,8F,4E,5E,4E,75,4E,5
6,FF,FC,30,2E,00,08
2650 data C1,FC,00,18,D0,BC,00,00,15,1
4,20,40,31,6E,00,0A
2660 data 00,0A,3E,B9,00,00,1E,8E,3F,3
9,00,00,1E,B2,3F,39
2670 data 00,00,1E,B6,3F,39,00,00,1E,B
4,3F,3C,00,01,3F,2E
2680 data 00,08,2F,3C,00,00,15,14,4E,B
9,00,00,12,5A,DF,FC
2690 data 00,00,00,0E,4E,5E,4E,75,4E,5
6,FF,EE,42,6E,FF,F2
2700 data 2E,BC,00,00,14,84,61,00,F7,7
E,3D,40,FF,FE,0C,39
2710 data 00,50,00,00,14,88,67,0A,0C,3
9,00,70,00,00,14,88
2720 data 66,06,3D,7C,00,01,FF,F2,3E,B
C,00,02,2F,3C,00,00
2730 data 14,84,61,00,F6,B8,58,8F,3D,4
0,FF,FC,3E,BC,00,02
2740 data 2F,3C,00,00,14,86,61,00,F6,A
4,58,8F,3D,40,FF,FA
2750 data 0C,6E,00,04,FF,FE,6F,12,4A,6
E,FF,F2,66,0C,0C,6E
2760 data 00,0C,FF,FC,66,04,42,6E,FF,F
C,4A,6E,FF,F2,67,14
2770 data 4A,6E,FF,FC,67,0E,0C,6E,00,0
C,FF,FC,64,06,06,6E
```

```
2780 data 00,0C,FF,FC,0C,6E,00,04,FF,F
E,6D,32,0C,6E,00,18
2790 data FF,FC,64,2A,0C,6E,00,3C,FF,F
A,64,22,42,40,30,2E
2800 data FF,FC,72,0B,E3,68,3E,80,42,4
0,30,2E,FF,FA,EB,48
2810 data D1,57,3F,3C,00,2D,4E,B9,00,0
0,13,18,54,8F,2E,BC
2820 data 00,00,14,8E,61,00,F6,C0,3D,4
0,FF,FE,3E,BC,00,02
2830 data 2F,3C,00,00,14,8E,61,00,F6,1
4,58,8F,3D,40,FF,F8
2840 data 3E,BC,00,02,2F,3C,00,00,14,9
0,61,00,F6,00,58,8F
2850 data 3D,40,FF,F6,3E,BC,00,02,2F,3
C,00,00,14,92,61,00
2860 data F5,EC,58,8F,D0,7C,FF,B0,3D,4
0,FF,F4,42,40,30,2E
2870 data FF,F4,C0,7C,00,03,67,0A,33,F
C,00,1C,00,00,15,BE
2880 data 60,08,33,FC,00,1D,00,00,15,B
E,0C,6E,00,06,FF,FE
2890 data 6D,60,4A,6E,FF,F8,67,5A,0C,6
E,00,0D,FF,F8,64,52
2900 data 4A,6E,FF,F6,67,4C,42,40,30,2
E,FF,F8,53,40,E3,48
2910 data 48,40,42,40,48,40,20,40,22,7
C,00,00,15,BC,30,30
2920 data 98,00,B0,6E,FF,F6,65,2A,42,4
0,30,2E,FF,F4,72,09
2930 data E3,68,3E,80,42,40,30,2E,FF,F
8,EB,48,81,57,42,40
2940 data 30,2E,FF,F6,81,57,3F,3C,00,2
B,4E,B9,00,00,13,18
2950 data 54,8F,4E,5E,4E,75,4E,56,FF,F
6,3E,BC,00,2C,4E,B9
2960 data 00,00,13,18,3D,40,FF,FC,3E,B
C,00,2A,4E,B9,00,00
2970 data 13,18,3D,40,FF,FA,42,40,30,2
E,FF,FC,C0,7C,F8,00
2980 data 72,0B,E2,68,3D,40,FF,FE,13,F
C,00,61,00,00,14,88
2990 data 4A,6E,FF,FE,66,08,3D,7C,00,0
C,FF,FE,60,1E,0C,6E
3000 data 00,0C,FF,FE,65,16,13,FC,00,7
0,00,00,14,88,0C,6E
3010 data 00,0C,FF,FE,67,06,04,6E,00,0
C,FF,FE,3E,BC,00,01
3020 data 10,2E,00,09,48,80,3F,00,3F,3
C,00,02,2F,3C,00,00
3030 data 14,84,3F,2E,FF,FE,61,00,F4,1
8,DF,FC,00,00,00,0A
3040 data 42,57,10,2E,00,09,48,80,3F,0
0,3F,3C,00,02,2F,3C
3050 data 00,00,14,86,42,40,30,2E,FF,F
C,C0,7C,07,E0,EA,48
3060 data 3F,00,61,00,F3,EC,DF,FC,00,0
0,00,0A,42,40,30,2E
3070 data FF,FA,C0,7C,FE,00,72,09,E2,6
8,D0,7C,00,50,3D,40
3080 data FF,FE,3E,BC,00,01,10,2E,00,0
9,48,80,3F,00,3F,3C
3090 data 00,02,2F,3C,00,00,14,92,3F,2
E,FF,FE,61,00,F3,B2
3100 data DF,FC,00,00,00,0A,3E,BC,00,0
1,10,2E,00,09,48,80
3110 data 3F,00,3F,3C,00,02,2F,3C,00,0
0,14,8E,42,40,30,2E
3120 data FF,FA,C0,7C,01,E0,EA,48,3F,0
0,61,00,F3,84,DF,FC
3130 data 00,00,00,0A,3E,BC,00,01,10,2
E,00,09,48,80,3F,00
3140 data 3F,3C,00,02,2F,3C,00,00,14,9
0,3F,2E,FF,FA,02,57
3150 data 00,1F,61,00,F3,5C,DF,FC,00,0
0,00,0A,42,40,13,C0
3160 data 00,00,14,94,13,C0,00,00,14,8
9,4E,5E,4E,75,4E,56
3170 data FF,E4,42,6E,FF,F0,2E,8E,55,9
7,2F,0E,06,97,FF,FF
3180 data FF,EE,2F,0E,55,97,2F,0E,55,9
7,2F,0E,5D,97,2F,0E
3190 data 59,97,42,67,3F,3C,03,E8,42,A
7,42,67,42,67,42,67
3200 data 42,67,42,67,42,67,42,67,42,6
7,42,67,42,67,3F,3C
3210 data 00,01,3F,3C,00,01,3F,3C,00,0
1,3F,3C,00,23,4E,B9
3220 data 00,00,10,68,DF,FC,00,00,00,3
8,3D,40,FF,F4,08,2E
3230 data 00,05,FF,F5,67,66,10,39,00,0
0,14,86,48,80,C0,7C
3240 data 00,0F,C1,FC,00,0A,12,39,00,0
0,14,87,48,81,C2,7C
3250 data 00,0F,D0,41,3D,40,FF,F6,3E,B
C,00,2C,4E,B9,00,00
3260 data 13,18,C0,7C,07,E0,EA,40,3D,4
0,FF,F8,30,2E,FF,F6
3270 data B0,6E,FF,F8,67,26,70,10,48,8
0,3E,80,61,00,FD,F8
3280 data 3E,BC,00,10,3F,3C,00,01,61,0
0,FB,FE,54,8F,3E,BC
3290 data 00,10,3F,3C,00,02,61,00,FB,F
0,54,8F,08,2E,00,00
3300 data FF,F5,67,1A,30,2E,FF,EE,C0,7
C,00,FF,B0,7C,00,0D
3310 data 66,0C,3D,7C,00,03,FF,F2,3D,7
C,00,01,FF,F0,08,2E
3320 data 00,01,FF,F5,67,00,00,9C,3E,A
E,FF,FA,3F,2E,FF,FC
3330 data 3F,3C,00,08,42,67,2F,2E,00,0
8,4E,B9,00,00,12,A4
3340 data DF,FC,00,00,00,0A,3D,40,FF,F
2,0C,6E,00,04,FF,F2
3350 data 67,08,0C,6E,00,03,FF,F2,66,4
0,30,2E,FF,F2,C1,FC
3360 data 00,18,D0,BC,00,00,15,14,20,4
0,3D,68,00,0A,FF,E8
3370 data 30,2E,FF,E8,72,01,B3,40,3D,4
0,FF,EA,3E,AE,FF,E8
3380 data 3F,2E,FF,EA,3F,2E,FF,F2,2F,2
E,00,08,4E,B9,00,00
3390 data 11,E4,50,8F,3D,40,FF,F0,60,2
8,4A,6E,FF,F2,6F,08
3400 data 3D,7C,00,01,FF,F0,60,1A,4A,6
E,FF,F2,6C,14,3E,BC
3410 data 00,07,3F,3C,00,02,3F,3C,00,0
3,4E,B9,00,00,13,08
3420 data 58,8F,4A,6E,FF,F0,67,00,FE,6
E,30,2E,FF,F2,4E,5E
3430 data 4E,75,4E,56,FF,F6,33,EE,00,0
8,00,00,1E,54,30,2E
3440 data 00,08,D0,7C,FF,F6,C1,FC,00,0
3,48,C0,D0,BC,00,00
3450 data 13,28,2D,40,FF,FA,3D,7C,00,0
1,FF,FE,60,1E,20,6E
3460 data FF,FA,10,10,48,80,32,6E,FF,F
E,D3,C9,D3,FC,00,00
3470 data 1E,54,32,80,52,AE,FF,FA,52,6
E,FF,FE,0C,6E,00,04
3480 data FF,FE,6D,DA,2E,B9,00,00,1E,8
A,4E,B9,00,00,00,14
3490 data 42,40,30,39,00,00,1E,7C,4E,5
E,4E,75,4E,56,FF,FA
3500 data 23,FC,00,00,1E,54,00,00,1C,1
E,23,FC,00,00,1E,36
3510 data 00,00,1C,22,23,FC,00,00,1E,5
C,00,00,1C,26,23,FC
3520 data 00,00,1E,7C,00,00,1C,2A,23,F
C,00,00,1E,AA,00,00
3530 data 1C,2E,23,FC,00,00,1E,BE,00,0
0,1C,32,23,FC,00,00
3540 data 1C,1E,00,00,1E,8A,3E,BC,00,0
A,61,00,FF,46,33,F9
3550 data 00,00,1E,7C,00,00,1E,90,70,0
1,4E,5E,4E,75,4E,56
```

```
3560 data FF,FC,3E,BC,00,13,61,00,FF,2
A,70,01,4E,5E,4E,75
3570 data 4E,56,FF,FC,33,EE,00,08,00,0
0,1E,5C,33,EE,00,0A
3580 data 00,00,1E,5E,33,EE,00,0C,00,0
0,1E,60,3E,BC,00,15
3590 data 4E,B9,00,00,0F,16,20,6E,00,0
E,30,B9,00,00,1E,7E
3600 data 20,6E,00,12,30,B9,00,00,1E,8
0,20,6E,00,16,30,B9
3610 data 00,00,1E,82,20,6E,00,1A,30,B
9,00,00,1E,84,42,40
3620 data 30,39,00,00,1E,7C,4E,5E,4E,7
5,4E,56,FF,FC,23,EE
3630 data 00,08,00,00,1E,AA,3E,BC,00,1
7,4E,B9,00,00,0F,16
3640 data 4E,5E,4E,75,4E,56,FF,FC,33,E
E,00,08,00,00,1E,5C
3650 data 33,EE,00,0A,00,00,1E,5E,33,E
E,00,0C,00,00,1E,60
3660 data 33,EE,00,0E,00,00,1E,62,33,E
E,00,10,00,00,1E,64
3670 data 33,EE,00,12,00,00,1E,66,33,E
E,00,14,00,00,1E,68
3680 data 33,EE,00,16,00,00,1E,6A,33,E
E,00,18,00,00,1E,6C
3690 data 33,EE,00,1A,00,00,1E,6E,33,E
E,00,1C,00,00,1E,70
3700 data 33,EE,00,1E,00,00,1E,72,33,E
E,00,20,00,00,1E,74
3710 data 33,EE,00,22,00,00,1E,76,23,E
E,00,24,00,00,1E,AA
3720 data 33,EE,00,28,00,00,1E,78,33,E
E,00,2A,00,00,1E,7A
3730 data 3E,BC,00,19,4E,B9,00,00,0F,1
6,20,6E,00,2C,30,B9
3740 data 00,00,1E,7E,20,6E,00,30,30,B
9,00,00,1E,80,20,6E
3750 data 00,34,30,B9,00,00,1E,82,20,6
E,00,38,30,B9,00,00
3760 data 1E,84,20,6E,00,3C,30,B9,00,0
0,1E,86,20,6E,00,40
3770 data 30,B9,00,00,1E,88,42,40,30,3
9,00,00,1E,7C,4E,5E
3780 data 4E,75,4E,56,FF,FC,33,EE,00,0
8,00,00,1E,5C,33,EE
3790 data 00,0A,00,00,1E,5E,3E,BC,00,1
A,4E,B9,00,00,0F,16
3800 data 4E,5E,4E,75,4E,56,FF,FC,23,E
E,00,08,00,00,1E,AA
3810 data 33,EE,00,0C,00,00,1E,5C,3E,B
C,00,32,4E,B9,00,00
3820 data 0F,16,4E,5E,4E,75,4E,56,FF,F
C,33,EE,00,08,00,00
3830 data 1E,5C,33,EE,00,0A,00,00,1E,5
E,33,EE,00,0C,00,00
3840 data 1E,60,33,EE,00,0E,00,00,1E,6
2,33,EE,00,10,00,00
3850 data 1E,64,33,EE,00,12,00,00,1E,6
6,33,EE,00,14,00,00
3860 data 1E,68,33,EE,00,16,00,00,1E,6
A,33,EE,00,18,00,00
3870 data 1E,6C,3E,BC,00,33,4E,B9,00,0
0,0F,16,4E,5E,4E,75
3880 data 4E,56,FF,FC,23,EE,00,08,00,0
0,1E,AA,33,EE,00,0C
3890 data 00,00,1E,5E,33,EE,00,0E,00,0
0,1E,60,33,EE,00,10
3900 data 00,00,1E,62,3E,BC,00,4B,4E,B
9,00,00,0F,16,4E,5E
3910 data 4E,75,4E,56,FF,FC,33,EE,00,0
8,00,00,1E,5C,23,EE
3920 data 00,0A,00,00,1E,AA,3E,BC,00,4
E,4E,B9,00,00,0F,16
3930 data 4E,5E,4E,75,4E,56,FF,FC,33,E
E,00,08,00,00,1E,5C
3940 data 23,EE,00,0A,00,00,1E,AA,3E,B
C,00,23,4E,B9,00,00
3950 data 0F,16,4E,5E,4E,75,4E,56,FF,F
C,23,EE,00,08,00,00
3960 data 1E,AA,33,EE,00,0C,00,00,1E,5
C,33,EE,00,0E,00,00
3970 data 1E,5E,33,EE,00,10,00,00,1E,6
0,33,EE,00,12,00,00
3980 data 1E,62,33,EE,00,14,00,00,1E,6
4,33,EE,00,16,00,00
3990 data 1E,66,3E,BC,00,2A,4E,B9,00,0
0,0F,16,4E,5E,4E,75
4000 data 4E,56,FF,FC,23,EE,00,08,00,0
0,1E,AA,33,EE,00,0C
4010 data 00,00,1E,5C,33,EE,00,0E,00,0
0,1E,5E,33,EE,00,10
4020 data 00,00,1E,60,33,EE,00,12,00,0
0,1E,62,3E,BC,00,2B
4030 data 4E,B9,00,00,0F,16,4E,5E,4E,7
5,4E,56,FF,FC,33,EE
4040 data 00,08,00,00,1E,5C,3E,BC,00,6
B,4E,B9,00,00,0F,16
4050 data 4E,5E,4E,75,23,DF,00,00,1A,1
A,4E,4E,2F,39,00,00
4060 data 1A,1A,4E,75,23,DF,00,00,1A,1
A,4E,4D,2F,39,00,00
4070 data 1A,1A,4E,75,23,DF,00,00,1A,1
A,4E,41,2F,39,00,00
4080 data 1A,1A,4E,75,00,01,00,02,01,0
1,02,01,01,00,01,01
4090 data 02,01,01,01,01,01,00,00,00,0
0,00,00,00,00,00,00
4100 data 01,00,00,01,00,03,05,00,05,0
5,00,00,01,01,02,01
4110 data 00,10,07,01,02,01,00,00,00,0
0,00,00,00,00,00,00
4120 data 01,01,01,02,01,01,02,01,01,0
2,01,01,01,01,02,01
4130 data 01,01,00,00,00,00,00,00,00,0
0,00,00,00,00,02,01
4140 data 01,01,01,01,06,01,01,04,01,0
1,01,03,01,02,01,01
4150 data 04,02,01,08,01,01,00,00,00,0
0,00,00,01,01,01,09
4160 data 01,01,01,01,01,01,01,00,00,0
5,01,00,00,00,00,00
4170 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4180 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4190 data 00,00,00,00,00,00,00,00,04,0
3,00,08,03,00,06,01
4200 data 00,08,01,00,08,01,00,04,01,0
1,03,01,01,00,05,00
4210 data 01,01,01,00,05,00,00,01,01,0
0,01,01,00,00,00,00
4220 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4230 data 00,00,00,00,00,00,02,02,00,00,0
0,00,00,00,00,00,00
4240 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
4250 data 00,00,05,01,00,05,01,00,01,0
1,00,01,01,00,02,05
4260 data 00,06,01,00,02,01,00,01,01,0
0,06,05,00,00,00,00
4270 data 00,01,01,00,01,00,02,01,00,0
2,01,01,01,01,01,00
4280 data 00,00,00,00,00,00,00,00,00,0
0,00,00,00,00,00,01
4290 data 02,03,01,02,01,01,01,01,01,0
1,00,01,01,00,01,02
4300 data 5F,5F,5F,5F,5F,00,00,00,00,0
0,5F,5F,5F,5F,5F,5F
4310 data 00,00,00,00,45,58,49,54,20,0
0,53,54,4F,52,45,00
4320 data 00,00,14,84,00,00,15,D4,00,0
0,15,DD,00,03,00,06
4330 data 00,02,12,00,00,00,00,00,00,0
6,00,09,00,00,14,8E
```

```
4340 data 00,00,15,E3,00,00,15,EC,00,0
3,00,06,00,02,12,00
4350 data 00,00,00,00,00,07,00,09,00,0
0,1E,B8,00,00,15,F3
4360 data 00,00,15,F4,00,05,00,06,00,0
2,11,00,00,00,FF,FF
4370 data 00,06,00,01,00,00,15,F5,00,0
0,15,F9,00,00,15,FA
4380 data 00,05,00,06,00,02,11,00,00,0
0,FF,FF,00,04,00,01
4390 data FF,FF,00,01,00,04,00,14,00,0
0,00,20,00,02,11,41
4400 data 00,00,00,00,00,17,00,05,00,0
2,FF,FF,FF,FF,00,1E
4410 data 00,40,00,10,00,00,14,A4,00,0
2,00,01,00,09,00,01
4420 data 00,03,FF,FF,FF,FF,00,1E,00,4
0,00,10,00,00,14,C0
4430 data 00,0C,00,01,00,09,00,01,00,0
4,FF,FF,FF,FF,00,16
4440 data 00,07,00,20,00,00,14,DC,00,0
2,00,03,00,0E,00,01
4450 data 00,00,FF,FF,FF,FF,00,16,00,6
1,00,20,00,00,14,F8
4460 data 00,12,00,03,00,03,00,01,23,3
A,5C,44,45,53,4B,54
4470 data 4F,50,2E,49,4E,46,00,00,00,0
0,00,0F,00,01,00,02
4480 data 00,04,00,06,00,03,00,05,00,0
7,00,08,00,09,00,0A
4490 data 00,0C,00,0E,00,0B,00,0D,00,1
F,00,1D,00,1F,00,1E
4500 data 00,1F,00,1E,00,1F,00,1F,00,1
E,00,1F,00,1E,00,1F
4510 data 5F,5F,3A,5F,5F,20,5F,6D,00,3
9,39,39,39,61,00,5F
4520 data 5F,2F,5F,5F,2F,5F,5F,00,39,3
9,39,39,39,39,00,00
4530 data 00,53,45,54,00,00,00,20,20,4
4,69,67,69,74,61,6C
4540 data 20,63,6C,6F,63,6B,00,23,62,0
0,23,63,00,00,5F,63
4550 data 72,79,73,74,61,6C,A2,00,00,0
0,00,14,5F,63,74,72
4560 data 6C,5F,63,6E,A4,00,00,00,13,2
8,5F,74,69,6D,65,00
4570 data 00,00,A4,00,00,00,14,84,5F,6
4,61,74,65,00,00,00
4580 data A4,00,00,00,14,8E,5F,65,78,6
9,74,5F,74,65,A4,00
4590 data 00,00,14,98,5F,73,74,6F,72,6
5,5F,74,A4,00,00,00
4600 data 14,9E,5F,74,78,74,69,6E,66,6
F,A4,00,00,00,14,A4
4610 data 5F,63,6C,6B,5F,74,72,65,A4,0
0,00,00,15,14,5F,66
4620 data 69,6E,64,00,00,00,A2,00,00,0
0,00,20,5F,64,70,65
4630 data 65,6B,00,00,A2,00,00,00,00,7
E,5F,67,65,74,5F,61
4640 data 73,63,A2,00,00,00,00,B4,5F,6
7,65,74,5F,6E,75,6D
4650 data A2,00,00,00,01,70,5F,63,6F,7
0,79,5F,73,74,A2,00
4660 data 00,00,01,E8,5F,73,74,72,5F,6
C,65,6E,A2,00,00,00
4670 data 02,0A,5F,6D,61,69,6E,00,00,0
0,A2,00,00,00,02,2E
4680 data 5F,67,65,74,5F,62,5F,68,A2,0
0,00,00,05,BE,5F,67
4690 data 65,74,5F,63,5F,68,A2,00,00,0
0,06,48,5F,67,65,74
4700 data 5F,74,69,6D,A2,00,00,00,0C,1
A,5F,72,65,64,72,61
4710 data 77,00,A2,00,00,00,0A,2C,5F,6
4,6F,5F,63,6C,6F,63
4720 data A2,00,00,00,08,6A,5F,77,61,6
9,74,5F,62,75,A2,00
4730 data 00,00,0D,72,5F,75,70,64,61,7
4,65,00,A2,00,00,00
4740 data 09,D8,5F,73,65,74,5F,74,69,6
D,A2,00,00,00,0A,7C
4750 data 5F,63,72,79,73,5F,69,66,A2,0
0,00,00,0F,16,5F,61
4760 data 70,70,6C,5F,69,6E,A2,00,00,0
0,0F,80,5F,61,70,70
4770 data 6C,5F,65,78,A2,00,00,00,0F,E
2,5F,65,76,6E,74,5F
4780 data 62,75,A2,00,00,00,0F,F4,5F,6
5,76,6E,74,5F,6D,65
4790 data A2,00,00,00,10,4E,5F,65,76,6
E,74,5F,6D,75,A2,00
4800 data 00,00,10,68,5F,65,76,6E,74,5
F,64,63,A2,00,00,00
4810 data 11,46,5F,66,6F,72,6D,5F,64,6
F,A2,00,00,00,11,68
4820 data 5F,66,6F,72,6D,5F,64,69,A2,0
0,00,00,11,8A,5F,67
4830 data 72,61,66,5F,77,61,A2,00,00,0
0,11,E4,5F,67,72,61
4840 data 66,5F,6D,6F,A2,00,00,00,12,1
6,5F,6D,65,6E,75,5F
4850 data 72,65,A2,00,00,00,12,38,5F,6
F,62,6A,63,5F,64,72
4860 data A2,00,00,00,12,5A,5F,6F,62,6
A,63,5F,66,69,A2,00
4870 data 00,00,12,A4,5F,77,69,6E,64,5
F,75,70,A2,00,00,00
4880 data 12,DE,5F,67,65,6D,64,6F,73,0
0,A2,00,00,00,13,18
4890 data 5F,62,69,6F,73,00,00,00,A2,0
0,00,00,13,08,5F,78
4900 data 62,69,6F,73,00,00,A2,00,00,0
0,12,F8,5F,69,6E,74
4910 data 69,6E,00,00,A1,00,00,00,1A,1
E,5F,69,6E,74,6F,75
4920 data 74,00,A1,00,00,00,1B,1E,5F,6
3,00,00,00,00,00,00
4930 data A1,00,00,00,1C,1E,5F,70,74,7
3,69,6E,00,00,A1,00
4940 data 00,00,1C,36,5F,70,74,73,6F,7
5,74,00,A1,00,00,00
4950 data 1D,36,5F,67,6C,6F,62,61,6C,0
0,A1,00,00,00,1E,36
4960 data 5F,63,6F,6E,74,72,6F,6C,A1,0
0,00,00,1E,54,5F,69
4970 data 6E,74,5F,69,6E,00,A1,00,00,0
0,1E,5C,5F,69,6E,74
4980 data 5F,6F,75,74,A1,00,00,00,1E,7
C,5F,61,64,5F,63,00
4990 data 00,00,A1,00,00,00,1E,8A,5F,6
3,68,00,00,00,00,00
5000 data A1,00,00,00,1E,8E,5F,67,6C,5
F,61,70,69,64,A1,00
5010 data 00,00,1E,90,5F,63,6F,6E,74,7
2,6C,00,A1,00,00,00
5020 data 1E,92,5F,61,64,64,72,5F,69,6
E,A1,00,00,00,1E,AA
5030 data 5F,63,77,00,00,00,00,00,A1,0
0,00,00,1E,B2,5F,63
5040 data 78,00,00,00,00,00,A1,00,00,0
0,1E,B4,5F,63,79,00
5050 data 00,00,00,00,A1,00,00,00,1E,B
6,5F,62,75,74,74,6F
5060 data 6E,00,A1,00,00,00,1E,B8,5F,6
1,64,64,72,5F,6F,75
5070 data A1,00,00,00,1E,BE,00,00,00,0
2,06,82,1C,01,98,06
5080 data 06,06,10,26,2A,10,0C,0A,08,0
A,1E,12,08,0A,26,1E
5090 data 1C,14,2C,12,2A,1C,1C,1C,28,0
6,0C,06,06,08,06,08
5100 data 12,22,0C,06,06,06,06,10,0E,0
8,06,06,38,06,06,06
5110 data 06,12,10,24,6A,24,38,70,36,0
C,0A,A2,10,0E,4A,2E
```

```
5120 data 0E,0C,06,0E,26,20,1A,18,18,1
A,1E,0E,20,12,1A,1E
5130 data 08,36,0E,06,06,06,0E,06,18,1
0,0A,12,14,76,08,12
5140 data 14,14,22,0A,34,32,14,0E,1E,1
E,24,22,44,24,2E,1A
5150 data 06,5A,18,10,16,7E,28,2A,2E,1
E,14,20,18,06,08,0E
5160 data 04,06,04,06,04,06,04,06,04,0
6,04,06,04,0E,04,24
5170 data 08,08,0A,0A,0A,0A,0A,08,10,0
A,10,08,08,08,08,08
5180 data 08,08,08,08,08,08,08,08,08,0
8,08,0A,0A,0A,0A,0A
5190 data 0A,0A,08,10,08,0A,10,08,0A,1
0,08,08,08,08,08,08
5200 data 08,08,0A,10,08,08,08,0A,10,0
8,0A,10,08,0A,10,08
5210 data 08,08,08,08,08,08,0A,10,08,08,0
8,08,0A,10,0A,0A,08
5220 data 08,08,08,08,01,84,04,04,14,0
4,04,14,04,04,14,04
5230 data 04,38,18,18,18,00
5240 data *
```

•

ST CHECKSUM DATA.
(see page 8)

```
 100 data  109, 948, 117, 614, 503, 23
6, 410, 427, 14, 109, 3487
 190 data  668, 357, 655, 544, 755, 92
0, 36, 273, 961, 962, 6131
 1080 data  837, 915, 908, 64, 975, 36
6, 137, 41, 170, 956, 5369
 1180 data  104, 191, 71, 894, 993, 11
2, 274, 20, 994, 22, 3675
 1280 data  962, 331, 271, 113, 922, 8
06, 162, 49, 197, 899, 4712
 1380 data  845, 998, 118, 55, 100, 18
3, 985, 990, 875, 754, 5903
 1480 data  6, 125, 75, 785, 985, 197,
 733, 250, 964, 93, 4213
 1580 data  986, 16, 219, 139, 920, 88
5, 884, 245, 871, 994, 6159
 1680 data  892, 998, 177, 826, 138, 9
86, 976, 745, 800, 98, 6636
 1780 data  934, 143, 857, 855, 821, 9
05, 723, 824, 16, 821, 6899
 1880 data  803, 879, 863, 801, 849, 4
7, 134, 920, 336, 110, 5742
 1980 data  273, 198, 283, 26, 41, 947
, 975, 263, 58, 11, 3075
 2080 data  36, 206, 162, 235, 334, 19
1, 270, 158, 184, 74, 1850
 2180 data  934, 915, 156, 214, 175, 3
72, 130, 43, 126, 933, 3998
 2280 data  320, 885, 911, 45, 107, 95
9, 78, 938, 980, 873, 6096
 2380 data  947, 693, 904, 913, 8, 122
, 775, 800, 826, 880, 6868
 2480 data  910, 773, 742, 921, 740, 8
61, 916, 61, 976, 870, 7770
 2580 data  838, 75, 814, 824, 783, 79
2, 966, 864, 836, 813, 7605
 2680 data  860, 66, 998, 670, 895, 55
, 0, 88, 184, 93, 3909
 2780 data  31, 52, 173, 926, 2, 960,
838, 901, 210, 988, 5081
 2880 data  955, 110, 106, 738, 70, 29
, 925, 118, 941, 124, 4116
 2980 data  987, 129, 892, 24, 690, 12
, 728, 1, 910, 98, 4471
 3080 data  917, 937, 831, 723, 208, 6
98, 868, 885, 789, 220, 7076
 3180 data  13, 888, 795, 749, 912, 87
8, 854, 988, 144, 63, 6284
 3280 data  995, 805, 177, 997, 203, 7
94, 69, 92, 880, 231, 5243
 3380 data  73, 106, 102, 710, 174, 92
8, 97, 35, 198, 107, 2530
 3480 data  25, 948, 788, 783, 769, 84
4, 919, 796, 40, 981, 6893
 3580 data  808, 812, 771, 741, 16, 81
8, 996, 810, 787, 784, 7343
 3680 data  815, 826, 796, 829, 824, 8
65, 767, 752, 792, 799, 8065
 3780 data  30, 814, 29, 865, 973, 830
, 787, 784, 808, 919, 6839
 3880 data  24, 808, 857, 26, 858, 4,
898, 976, 872, 793, 6116
 3980 data  790, 922, 994, 794, 775, 4
0, 795, 835, 806, 788, 7539
 4080 data  577, 475, 487, 477, 484, 4
70, 493, 496, 481, 469, 4909
 4180 data  470, 496, 499, 478, 467, 4
72, 469, 493, 494, 485, 4823
 4280 data  474, 493, 875, 660, 674, 5
45, 652, 652, 745, 735, 6505
 4380 data  674, 690, 864, 562, 931, 8
69, 601, 943, 658, 651, 7443
 4480 data  538, 655, 671, 858, 825, 7
19, 756, 803, 839, 697, 7361
 4580 data  870, 814, 895, 848, 729, 8
00, 887, 845, 853, 745, 8286
 4680 data  910, 844, 846, 859, 850, 7
52, 907, 848, 865, 941, 8622
 4780 data  928, 883, 796, 850, 893, 8
64, 893, 844, 873, 840, 8664
 4880 data  856, 773, 857, 843, 668, 8
12, 777, 835, 913, 898, 8232
 4980 data  874, 696, 859, 787, 886, 7
40, 716, 775, 870, 717, 7920
 5080 data  673, 691, 599, 756, 713, 6
39, 698, 719, 569, 643, 6700
 5180 data  645, 622, 611, 622, 579, 7
24, 205, 4008
```

•

Keyboard Controlled Sequencer (KCS)

DR. T's MUSIC SOFTWARE
66 Louise Road
Chestnut Hill, MA 02167
(617) 244-6954
$195.00
520ST with TOS in ROM

Midisoft Studio

MIDISOFT CORPORATION
Box 1000
Bellevue, WA 98009
(206) 827-0750
$99.00
520ST with TOS in ROM

SMPTE-Track

by Stefan Daystrom
HYBRID ARTS INC.
11920 Olympic Blvd.
Los Angeles, CA 90064
(213) 826-3777
$575.00 (with *SMPTEMate* interface)
520ST with TOS in ROM
External sync requires *SMPTEMate*

by Charles F. Johnson

After a long wait, MIDI sequencing software is beginning to appear for the Atari ST, in droves. It seems as if every week brings an announcement of another company entering the MIDI sweepstakes with their idea for the perfect sequencer. As a professional musician and an Atari enthusiast, the explosion of new MIDI programs pleases me greatly, but I must ask the difficult question: "Is the MIDI software for the ST truly of professional quality yet?"

The ST, with its built-in MIDI ports, shows great promise as a computer for musicians. Yet, in the past few years, there has been more excellent MIDI software released for the Apple Macintosh than for any other personal computer. The editing capabilities of the best Mac sequencers (such as Performer and Total Music) are very sophisticated, allowing musicians to transform their compositions in ways that Mozart could never have envisioned. I'm sorry to report that the ST software I'm examining in this review is not quite up to its Macintosh competition in terms of flexibility and ease of use. However, some of the ST products show definite signs of becoming real contenders as they're developed further.

Dr. T's *KCS*.

The **Keyboard Controlled Sequencer (KCS)** from Dr. T's Music Software is a powerful and complex program. It comes on a copy-protected disk, which makes it difficult to install on a hard disk; you can run it from the hard drive, but the original master program disk must be in drive A.

The first big weakness of the **KCS** is its manual. I found this nearly incomprehensible to the average musician/computer user, full of tables of numbers, and with little or no introductory material for MIDI novices. It assumes a great deal of prior MIDI knowledge and experience. This is unfortunate, because the program really *needs* a good manual, perhaps more than any of the other sequencers in this review.

Most of the ST sequencers to date have relied on the visual metaphor of the multitrack tape recorder, to the extent of usually supplying on-screen buttons for functions like Play, Record, Rewind, Fast Forward, Auto-Locate, etc. **KCS** departs from the crowd here. The main program options of **KCS** are accessed from screens full of text, although GEM dialog boxes are used for some input. The program has three modes of operation: Track Mode, Song Mode and Open Mode.

Note: As I was finishing this review, I received a copy of version 1.2 of the **KCS**. The new program does have a more tape-recorder-like method of operation in Track Mode, with buttons for the common functions. However, the operation of the program is essentially the same.

It's possible to do quite a bit with **KCS** without ever leaving Track Mode. There are options for sending MIDI data through to different channels, muting individual tracks, setting cue points (places to start and stop) for each track, and setting the punch-in and punch-out times for each track.

The Track Mode screen displays 48 tracks, each of which can be given an eight-character name. In my opinion, that's not really enough for descriptive track names, but the advantage of this limit is that all 48 tracks can be visible on-screen at once.

Track Mode is the one most similar to a tape recorder, except that there are no play or record buttons (see above). Instead, the sequencer is started either from the computer keyboard, or by playing something on a MIDI synth connected to the ST. This can be a real problem, since, anytime you play a note on your synth, the **KCS** will start recording. You can disable this function and require a keypress to begin recording, but every time you switch to another mode, the Auto-Record function will be enabled again. When recording, the **KCS** will always save its data to the lowest-numbered free track, which means that you have no control over which tracks are used. I found this annoying, since I like to group different instruments on tracks and separate them with blank spaces as a visual aid.

Another problem with the **KCS** Track Mode is that the length of all tracks is set by the first track you record, although there is an option to set all the tracks to a specified length after first deleting them.

Song Mode lets you construct sequences out of the material you record in Track Mode. The tracks can be combined in any fashion, and you can specify transpositions, tempo changes and repetitions for a song. Song Mode on the **KCS** is similar to programming a song on a drum machine. You build small segments of music for each part of a song, then chain them together in a sequence. The **KCS** lets you

edit the order (and other events) of a sequence quickly and easily. In addition, you can have more than one song in memory at once, and arrange them to play in any order. There are cut-and-paste, delete, move and copy commands in Song Mode, and these can work on a specified range of time in any sequence, allowing you to easily transfer musical material between different songs.

Open Mode is probably the **KCS**'s most difficult mode to use. Essentially, Open Mode allows you to chain and play up to 128 sequences. . . at the same time, if you choose. Sequences can overlap, and contain "sequence start" and "sequence stop" events, which will cause other sequences to begin and end. Data from sequences can be freely intermingled in a very powerful and flexible way.

The **KCS** is the only sequencer reviewed here that supports full editing of each and every MIDI event recorded. Its Edit screen displays all events in a user-alterable window. For instance, say you play a wrong note, an A-flat that should have been a G-natural. You can pop into the Edit screen, find A-flat in the list of MIDI events, and change it to a G, just by typing in the letter G. In the same way, you can also quickly delete notes (with an option to leave the rest of the data in the same place, or move it back to fill the deleted spot), change note duration and velocity, and even add notes. In fact, it's possible to enter an entire piece of music from the event editing screen—if you're a masochist!

The **KCS**'s quantizing (time-correction) feature deserves special mention. The traditional quantization found in most sequencers lets you automatically correct the timing in a piece of recorded music to the nearest eighth note, quarter note, etc. In addition to this type of quantizing, the **KCS** allows you to quantize durations and velocities, as well, and specify a start and end point for quantization. This can be a powerful tool to achieve subtle phrasing and dynamic changes.

Unfortunately, the **KCS** makes you enter all your quantizing values as numbers. You can't simply point at a picture of an eighth note and click the mouse; you have to figure out how many MIDI clocks an eighth note amounts to, based on your current internal resolution setting. For example, if you have the internal "Steps per Beat" set at *96*, you would use a value of *24* to quantize a track to the nearest sixteenth note. Sound complicated? That's because it *is* complicated!

The program seems to have trouble using both the GEM interface (for its dialog and alert boxes) and its own method for checking the mouse. Some of the menu functions cause a GEM dialog box to appear, and, often, GEM thinks you're holding a mouse button down when you're not. To be fair, this is probably due to the limitations of GEM in the ST; if you use both AES and VDI input calls, the ST can get confused. This can usually be fixed just by moving the mouse or clicking one of its buttons. Another problem: occasionally, for no discernible reason, the keyboard will begin to Auto-Repeat even though you're not pressing a key. Neither of these problems is fatal, however.

The power of the **KCS** comes at the price of much complexity. Many commands require you to deal with screens full of numbers, to set such things as internal clock resolution, tempo, time correction, etc. If numbers scare you, I'd advise you to look at another sequencer; **KCS** is nothing but numbers, numbers and *more* numbers! But if you make the effort to learn Dr. T's method of sequencing, you'll be rewarded; this program is capable of amazing things, in spite of its lack of user-friendliness and few small bugs.

Midisoft Studio.

Midisoft Studio was first released under the name **Metatrak;** it comes on an unprotected disk, and can be easily backed up or installed on a hard disk. It's a fully GEM-based 32-track sequencer, with drop-down menus and dialog boxes controlling much of the action.

Interestingly, of the programs reviewed here, **Midisoft Studio** is the only one that lets you use the menus and dialogs even as music is being played. The action of a GEM program is normally suspended when a menu or dialog box is on-screen, so most ST sequencers disable these features while a sequence is playing. **Studio** uses an interrupt-driven method to send data to the MIDI port, allowing both GEM and MIDI to function concurrently.

Studio's main screen consists of a track display on the left, with drop-down menus above (as usual). Underneath the track display is a row of tape-recorder-style buttons: Pause, Rewind, Stop, Play and Record. These buttons act just like their counterparts on a real tape recorder. On the right side of the screen is a window displaying the current track and region you've marked for editing, the amount of free memory, the tempo and current location, and the on-screen metronome.

Studio has both real-time and step-time note entry (as do all of the sequencers reviewed here). The step-time feature is particularly well implemented; you can adjust the timing to 240 steps for every quarter note, and enter a velocity for each note. This last feature is nice for those of us who don't have a velocity-sensitive keyboard, but have synths that will *respond* to velocity information.

Midisoft Studio does not have full MIDI event editing (like the **KCS**), but you can define a section of a track and copy, move, delete or erase it, or insert it into another track at a different point in time. The editing marks are always displayed at the upper right, and you can click on the numbers with the mouse, to set editing points as a sequence is playing.

The program has both an internal metronome (which sounds a click through the monitor speaker) and the ability to send MIDI start/stop commands to a drum machine. A problem arises when you have a drum machine playing and also leave the internal metronome on; the metronome click tends to go just a bit faster than the MIDI clock the program sends to the drum machine, leading to a very confusing situation. However, if you turn off the metronome, there are no timing problems.

Almost all the options of **Studio** are accessed via the mouse and drop-down menus. The function keys F1 through F6 can be used for Pause, Stop, Rewind, Fast Forward, Play and Record; F9 and F10 will decrement and increment the tempo. The user interface is well done for the most part. I do wish there were more keyboard alternatives for some of the menu commands, like MIDI through (it would be nice to be able to change MIDI channels by typing a key, especially in a multi-synth setup) and track erasing.

Every time you start **Studio** with the Record button, what you play is saved on the currently active track. (If you don't put a track into "Record" mode, **Studio** automatically records on the lowest-numbered free track.) If you make a mistake and decide to start over, instead of fixing it with editing commands, you must first delete the track you just recorded—which means you must move to the "Edit" drop-down menu, select "Erase Track," then enter the track number into a dialog box and click on the OK button. This is a lot to go through just to try something again. In contrast, **MidiTrack ST Professional** lets you start over simply; nothing is saved until you tell the program to do so.

Studio's biggest failing, in my opinion, is its lack of a looping feature. Looping is used to construct long songs from short segments, or to repeat a part over and over so you can try different things against it. You can simulate looping in **Studio** by using the "Move" feature to repeat parts, but this is a halfway measure at best.

Overall, **Midisoft Studio** works well, and has enough features to make it a useful musical tool. The program comes with a reasonably well written manual that does a good job of explaining its operation. Its low price makes it even more attractive if you're on a budget but still want a reasonably powerful sequencer. Midisoft states that updated versions of **Studio** will include musical-notation-driven MIDI event editing, programmable tempo and time-signature changes, looping, and more.

SMPTE-Track.

Hybrid Arts is the only company (to my knowledge) with a MIDI sequencer for the

8-bit Atari computer, **MidiTrack III**. Last year, they released a scaled-down "consumer" version (called **EZ-Track**) of what was to be their professional ST sequencer. **SMPTE-Track** is the new top-of-the-line sequencer in the Hybrid Arts **MidiTrack ST** series, and it bears a certain resemblance to their 8-bit program. Many of the same key commands are used for identical functions, so if you're familiar with **MidiTrack III**, you'll find **SMPTE-Track** easy to learn.

But the new program also uses the mouse and drop-down menus. Almost all functions can be accessed either by keyboard or mouse. The program manual is attractively done, and well written although a little sketchy in places.

SMPTE-Track is shipped with a special custom interface box, which allows it to directly read many types of external sync sources, including the tape sync used by the 8-bit **MidiTrack**. The **SMPTEMate** box is also capable of reading and writing SMPTE (pronounced "simpty," it stands for Society of Motion Picture & Television Engineers) time code, an industry standard sync signal which has been used in movies and TV for years. SMPTE time code is generally considered the most reliable and flexible sync source available. Using SMPTE, you can instantly locate any point in a sequence and sync it with tracks that have already been recorded. You can also vary the speed of the music. Unlike most other types of tape sync, SMPTE does not force you to always use the same tempo. **SMPTE-Track**'s SMPTE capability is a powerful tool, although perhaps it's of most value to the professional musician rather than the computer/music hobbyist.

SMPTE-Track has a full range of MIDI control features, like MIDI through (to control several synths from a master keyboard), and MIDI start/stop and song pointer (which let the sequencer synchronize to an external drum machine or tape sync track). There's a terrific option called "Auto-Install," which allows you to automatically load synths with the proper patches whenever a song is loaded. This feature works with patch files created by the Hybrid Arts **GenPatch** program.

Of the three sequencers reviewed here, **SMPTE-Track** wins hands down in terms of user friendliness. The recording process is almost effortless and every important control (such as tempo, MIDI through, track muting, punch-in and -out, MIDI channel assignments, etc.) can be adjusted with the mouse while a sequence is playing or being recorded—without leaving the main screen. However, the sequencer does stop when you access a drop-down menu. If you wish, you can set the program so that the menus won't drop without a mouse click.

To mute, unmute and solo tracks, you simply click on the track display next to the track. Custom characters show the current status of each track. . .which MIDI channel it's assigned to, mute/unmute, protect status, plus a special series of indicators that flash when the sequence is playing, to give you some visual feedback.

The looping feature of **SMPTE-Track** is particularly well implemented, and a joy to use. You define up to ninety "sections," which can include any number of tracks. Each section has a starting and an ending time. You then string these sections together, to create what's called a "chain track." This contains all the looping commands, and when the sequencer plays (and the chain track isn't muted), it will control which tracks are heard. To work on individual sections again, simply mute the chain track—no need to switch between program modes (as with the **KCS**).

SMPTE-Track also has a nice feature called "registers." Registers remember settings for mutes, tempo, punch-in/punch-out and auto-locate points, and can be instantly selected by a click of the mouse button, at any time. I found registers to be an invaluable tool. You can set one register to a slow speed for careful recording and another to the real tempo of the song, then switch back and forth with ease. Another use for registers is to set auto-locate points to different sections of a song. Then you can play from any point simply by changing the active register. There are twenty-seven registers available for each song.

SMPTE-Track comes on a copy protected disk. However, the copy-protection is ignored by the program if it determines that there is a **SMPTEMate** box connected to the computer, making backup copies and hard disk installation possible. The only trouble with the **SMPTEMate** box is that it connects to both the RS-232 port (modem port) and the second joystick/mouse port. So, if you want to use a modem as well, you have to constantly switch cables around. I solved this problem by getting an RS-232 switch box; they're available for about $50 to $60.

I have one gripe with **SMPTE-Track**: the program needs more editing features. The only types of editing currently available in the program are punch-in and -out and step editing. I find this unacceptable in a product costing almost $600. According to the people at Hybrid Arts, they're hard at work on a graphics-oriented MIDI event editor, which will be a free update to all registered owners.

This is an excellent program, except for the lack of editing capabilities. The recording process is easy to learn and use, and the SMPTE interface box opens up a world of advanced musical possibilities. I do feel that **SMPTE-Track** is priced a bit high, especially in relation to most of ST software; but the price does include an external sync box (**SMPTEMate**), something no other ST sequencer currently provides. This program is a powerful musical tool.

ATARI USERS' GROUPS

Pasadena Area Local
Atari Computer Enthusiasts (PALACE)
Meetings covering the 8-bit computers.
For information, contact the BBS at (818) 359-1697.

California Original Atari ST User Group (COAST)
1015 Esther Dr., Pleasant Hill, CA 94523
(415) 945-1949
Meetings; newsletter: *FirST*. President: Chuck Thorp.

Santa Barbara Atari Computer Enthusiasts (SBACE)
P.O. Box 3678, Santa Barbara, CA 93130
Meetings; BBS; newsletter. President: Conrad Weiler.

Pueblo Atari ST Enthusiasts (PASTE)
c/o Computer Palace
702 West 10th St., Pueblo, CO 81003
Meetings; BBS. Contact George Heymann or call the BBS at (303) 544-4432 Sun. & Mon. at 300 baud.

Danbury Users Group (D'BUG)
21 Rita Dr., New Fairfield, CT 06812
Meetings; newsletter. President: Dan McNeil.

Atari Computer Owners Resource Network (ACORN)
c/o Ken Helms, 3627 Iowa Ct., Fort Wayne, IN 46815
Meetings; BBS; newsletter: *The Nutshell*.
President: Kim Stahn.

Cascades Atari Computer Enthusiasts (CACE)
P.O. Box 6161, Jackson, MI 49204
Meetings; BBS; newsletter: *Michigan Atari Magazine*.
President: Harold LaRue.

Michigan Atari Computer Enthusiasts (MACE)
P.O. Box 2785, Southfield, MI 48037
Meetings; BBS; newsletter: *The M.A.C.E. Journal*.
President: Tom Sturza.

Dedicated Users of STs (DUST)
P.O. Box 42096, Las Vegas, NV 89116
Meetings; newsletter: *DUST Bin*.
President: Mike Glenn.

Micros of Monmouth
P.O. Box 221, Englishtown, NJ 07726
Meetings; BBS; newsletter.
President: Mike Carovillano.

Metropolitan Atari News & Information Association (MANIA)
P.O. Box 536, Kingsbridge Station, Bronx, NY 10463
(212) 627-5830 Manhattan's only Atari users' group.
Meetings; BBS; newsletter. President: Nevin Shalit.

ST Atari Resource (STAR)
C-55 University Mall, Orem, UT 84508
Meetings; BBS; newsletter.
President: Dr. Thomas K. Martin.

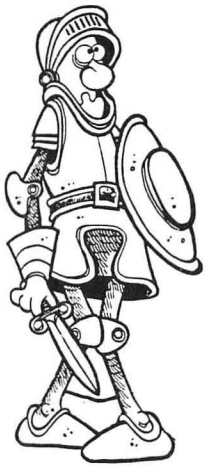
Blue Mountains Homebrew Computer Club
c/o Eric Lindsay
6 Hillcrest Ave., Faulconbridge NSW 2776, Australia
Meetings; newsletter: *The Jackintosh Gazette*.

Atari Computer Users' Group of the Future
c/o Alejandro Lotwin T., Apartado Postal 11-456
06100 Mexico, D.F., Mexico
Contact above for further information.

ATTENTION USERS' GROUPS

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

Ian's Quest



ST news, information and opinion.

by Ian Chadwick

There's a guerilla war going on in Zaire. Bet you didn't know that. I sure didn't. I spent three months last summer studying Zaire: its politics, geography, mineral resources, its ties to the American and Soviet spheres. I never once read about an ongoing guerilla war. Zaire seemed peaceful under the absolute but essentially benign hand of President Mobutu Sese Seko. Now I read the minor country news and am told, in 1986, FLNC soldiers jubilantly occupied Kinshasa—the once stately capital of the country known to many as the Belgian Congo. Curiously enough, by 1987, they were merely on the outskirts of the city. Hmmm . . .

Worse: I'm told Zaire is neutral toward the USA, but friendly toward the USSR—an about-face for Mobutu, who not only kicked the Soviet "advisors" out of his country, but allowed the CIA to use Zaire as their base in the Angola operation. Zaire is also described as an insignificant military power. I'm sure Mobutu would take offense—Zaire was the only sub-Saharan African nation to send troops to Chad when Libyan leader Muammar Qaddafi invaded it (well, his troops did, actually). Mobutu actually takes pride in his self-determined role as southern Africa's policeman.

And the final indignity, if the reappearance of the almost extinct FLNC wasn't bad enough, is that the insurgency is led by: Insurgency. That's right: a nameless, faceless group that goes by the generic name for their kind. A veritable Orwellian concept.

If you haven't guessed, I'm talking about Mindscape's **Balance of Power**—a complex, challenging and demanding simulation of world politics written by Chris Crawford, that old Atari 8-bit whiz kid. As a simulation and a game, I'm mightily impressed with it. And, as you might gather, as a lesson in history and politics, I'm not quite so enthralled.

The idea of the game is to play the leader of the USSR or the USA and, in a period of eight years, manage to prevail over the computer- or human-controlled opponent—diplomatically. Crawford has factored in the immediate end of the game when Defcon 1 is reached. Nuclear war is not the goal but, rather, the antithesis of it. Instead, you have at hand more cunning and subtle weapons: diplomatic relations, military and economic aid, intervention, destabilization, treaties and diplomatic pressure.

If that weren't enough, there is a veritable mountain of data to absorb about every country—from the literacy level, to the number of phones, schools, TVs and telephones, through violence and protests, assassinations, civil rights and riots, crossing investments vs. military spending, consumer spending against the population. A screen of nine bar charts and graphs measures the eight-year cycle of any nation through diplomatic relations, coups, insurgency and the like.

Think that's got you boggled? Wait—there's more. Newspapers proclaim the "headline events" for major and minor countries. You have to be able to pick out the wheat from the chaff: does the small turnout for the Brazilian President, coupled with the drop in military aid from the US, portend his downfall? Is the time ripe for an insurgency thrust? Or destabilization? It's an awesome thing to tackle. You need to be able to read between the lines,

weighing the increase in consumer spending against your economic aid—is it getting through to the people, or is it being pocketed by the higher-ups?

No move can be contemplated until you have examined all the variables. Some elements are worthless—canards, if you will, especially in the beginner and intermediate games. Other items contain the nuggets of critical data you need to make your decisions: the crowds cheer the President and the US opens trade relations—but, on page three, it says he's nationalizing the mines. *Is* he about to swing to the left? How do you react?

Much of the game design intrigues and delights me. It's the ultimate simulation, with a forest of variables through which I can tramp. It's a cacophony of data in which one must struggle to pick out the relevant tunes.

However, it would be closer to the truth to say that several deficiencies almost make me loathe to play it. Let's look at the reasons why.

Zaire is only one of sixty-two nations portrayed in **Balance of Power**. The information about its status, described above, comes from the "Closeup" screen in the beginner's level game. Data like this has the annoying habit of changing in different levels—as if, the harder the game becomes, the more realistic its data. This seems somewhat perplexing to me; why should it change like this?

Take a look at Mozambique, expert level: neutral to the USA but cold toward the USSR; trade relations with the former, none with the latter; and a moderate right government. Are we speaking of the same country? Mozambique is a Marxist state that the Republic of South Africa has been trying for years to destabilize. Mozambique's attempt in 1984 to reach western capitalist nations only fed fire to the Mozambique National Resistance (MNR—not the Makonde as Crawford states). They're still trying to remain Marxist, while encouraging western capital. But moderate right? Sheesh!

Chad (a country central to current African conflicts) doesn't appear on the map. Nor do a lot of other countries—about ninety-odd in all are left off. Some popular hot spots, like Grenada, Cyprus, Ireland, Cambodia, Bangladesh, Salvador, Guatemala, Bolivia, Yemen and the Falklands, are among the most obvious of the missing locales. Forget about sending US troops to Northern Ireland—the game doesn't allow it or several other interesting options. Yet each of these countries has had severe unrest, civil or other wars, and is a pawn in the great power game—equal to many countries included in **Balance of Power** and more than equal to others. (Can you imagine a civil war in Australia?)

The combination of changing, inexact and sometimes basically wrong data is annoying, frustrating and maddening. If the politically innocent play this game, they may get to believe it resembles the real world. It ain't necessarily so, folks. A simple read through such publications as the *African Review* will reveal a host of errata about that continent alone!

One of the problems most likely lies in the age of the data. **Balance** was first released in 1985 for the Macintosh (you know, the Magic Sac computer in its big box. . .). Since then, it doesn't appear that the database has been changed—it's at least two, if not three years old and out of date in many areas. Of twenty books in Crawford's bibliography, nine are dated 1982 or earlier.

But what about play? Well, as the American player, I found the Russians invaded Iran every game. If I questioned this (remember, this isn't sending in the troops, just *protesting*), they would always go to Defcon 1. Big bang—game over. I was never able to invade anyone. And, as the Russian player. . .those nasty Yankees invaded Iran! But when I protested, they meekly backed down! For that matter, I generally found the Russians were aggressive beyond realism (especially with Gorbachev in power, now), while the Americans were wimpy folk who pulled out when threatened. Maybe I've got a leftist microprocessor?

On the other hand, maybe Chris watched *Red Dawn* a few too many times and started to believe it was a documentary. The game has the feel of the paranoid commie-bashing American stereotyped in such movies. The Russkies are the bad guys who go for their guns every time and get all the breaks; the USA is polite and stands back when asked to, and obeys the UN. For that matter, a lot of US messages which indicate that country's acquiescence involve the United Nations—a sore point for many ultra-right Americans, and thrashed to death in the game. Sigh. I can't help but feel Crawford's political point of view is a tad biased, not to mention naive and unrealistic. Or maybe he's trying to teach us a lesson. In which case, class dismissed—you won't learn a lot here!

The action/response mechanism is simplistic as well. It allows only one response: question. Continued questioning, if the opponent doesn't back down, leads to nuclear war. I like to think that neither side would go to that level in a protest over the establishment of trade relations. The possibility of compromise, at least in the solitaire game, does not exist.

Just so that you won't think I'm picking on the game simply for a difference in view, or over a few tiddly details like facts, there are other problems here, too. The main one is the protection scheme on the disk. I don't *mind* protection, but this one saves games onto the *same* disk. A trifle dangerous? Worse; when you quit, it automatically saves the game to the *master* disk. And you can't save on another disk! This simple, convenient little feature was left out to help confound us all.

The game is *very* disk intensive. Why not let us copy it, then check the master once and let us write to a backup disk? This seems merely logical and benign. The current scheme is noticeably user-hostile. And let us save more than one game!

Aha! Here's another little feature they neglected to mention: If you save a game, then decide to go look at another level just for the fun of it, and then quit without even making a move—you guessed it! **Balance**'s software writes over the single saved game on the disk. Yup, you just lost the hours of play you went through! Nice trick. My usual method is to find a situation where I can go to Defcon 1 and die, in order to escape without saving.

And finally, the game is neither as fast nor as smooth as it was on the Mac, although the colors in medium resolution are much nicer than the grey scale on the Mac. The Mac version was ported over almost exactly, which means that some of the ST's features were ignored in order to share identical displays. A pity, really—this is probably the one time where GEM might have had a beneficial effect.

The bottom line: a superb idea, a well-crafted and designed simulation with tremendous potential, flawed by inaccurate data and inadequate disk I/O methods. A real tragedy, because I want to like this game so much. My suggestion: clean up the disk I/O nonsense, let the user save games to another disk and rebuild the database. Better still, create a utility that allows users to enter updated data themselves, thus keeping the game current. //

---

*Ian Chadwick is a Toronto-based writer who currently works for Laser Data Technology, a company which publishes MS-DOS-based desktop publishing software. He is also working on a book on GFA BASIC and an **ST-Log** GFA BASIC column, and is writing a "Battle of Midway" game in GFA.*

Step 1

CUSTOMIZING THE GEM DESKTOP

The fourth in our series for first-time computer users.

by Maurice Molyneaux

Well, we've covered the major points concerning hard and soft wares; we ran through using GEM last time . . .Now what?

You say you wish your computer would boot in medium resolution? And you wish you could set the desktop windows to open in specific spots?

Okay, I can take a hint. A great many ST owners—not all of them new to computers—don't know the first thing about configuring a computer so it's tailored to their needs. The ST owner's manual touches on installing disk drives and changing resolutions, and gives a mercilessly brief description of installing applications. But I'm going to go over these in considerable detail, in addition to giving you some pointers on how to do even more than you may have thought possible—including renaming the trash can and installing multiple applications.

As a reference, dig out your ST owner's manual, because, where applicable, I've used the terms found in it ("palette brightness controls," etc.) Also, keep copies of the previous **Step 1** articles on hand (if you have those issues of **ST-Log**); I will sometimes refer you to them. If you get confused as to what I'm talking about, these references may help.

Creating DESKTOP. INF.

Before saving a desktop configuration, you first need to arrange the GEM desktop. Boot (turn on) your system, making sure all the auto-starting utilities you want to use are present—particularly the RAMdisk-installing AUTO folders (discussed in **Step 1: Soft Wares**, March '87 **ST-Log**) and/or desk accessories.

If you have a color system, decide if you want to default to low or medium resolution when you boot. This is selected with "Set Preferences" under the options menu. (You can also make the ST *not* prompt you for confirmation before copying or deleting files, by clicking in the appropriate boxes. . .although I advise against doing so!) Monochrome system owners can use only high resolution.

Next, move the drive icons and trash can where you'd prefer to have them all the time. The desktop icons work on a "grid" system, so you'll find they will snap into predefined positions. Both floppy drive icons are labeled "Floppy Disk." Maybe you'd rather have them as "Floppy A" and "Floppy B." If you have two drives, set up side by side, you might want to call them "Left Disk" and "Right Disk." Maybe "Top Disk" and "Bottom Disk" if they're stacked, or "Inboard" and "Outboard" if you have a 1040ST and an external drive. Hard disk owners usually partition their hard drives into two or more logical drives, and may want to label each by what they contain, such as "Games," "VIP," or something equally descriptive.

Once you've decided what you wish to call a particular drive, single-click on it with your mouse, then select "Install Drive" under the options menu. Click your mouse on the "Icon Label" line, hit ESCAPE or BACKSPACE to clear the current label, then type your own—up to a maximum of twelve characters. (*Notice:* you users of the original 206K TOS.IMG file for bootup cannot type blank spaces and some other characters in these labels! If this is the case, you can use the underline character to fake a space, etc. Do this for now. Later, we'll edit the DESKTOP.INF file, and can replace the underlines.)

When you're done renaming, click on the "Install" box and look at the disk icon you just edited. If you wish to change any others, do that now.

If you need to install a disk icon for the first time (say, drive D for a RAMdisk), and have never done it before, here's how: single-click on one of the disk icons already present on the desktop, then click on "Install Drive" (op-

tions menu). When the dialog box appears, the cursor (a thin vertical line) should appear on the "Drive Identifier" line (if not, click the mouse on the letter at the end of the line). Press the BACKSPACE key and type the letter for the drive you're about to install (drive D in this example). Make certain you enter an uppercase letter.

Next, click on "Icon Label," hold down BACKSPACE, or just press ESCAPE to clear the field, then type in the description you want. When done, click on the "Install" box (don't press RETURN), and the icon should appear on your desktop. If you can't see the icon, it's probably hidden under a window (if any are open), or else you hit "Cancel" or "Remove" by mistake, instead of "Install."

(Interestingly, if you install a disk drive and make the "Drive Identifier" a lowercase *C*, double-clicking on the resulting icon will open a window to the cartridge slot. No kidding!)

Users of a commercially purchased RAMdisk should consult its enclosed instructions in order to have it automatically initialize. Users of public domain RAMdisks (usually desk accessories) may or may not have instructions, but usually all that's required for auto-initialization is to have the RAMdisk accessory on the boot disk on power-up, and to have had the RAMdisk icon present when the desktop was saved.

Back to business. Your drive icons should now be labeled to your liking, and all the icons positioned where you want. You can't rename the trash can yet. Just position it; we'll fix its name later.

Color system owners, do you like the default colors on the desktop? If not, use the control panel (if present) and change them to what pleases you. Monochrome owners, if you select one of the active "Palette Boxes" on the control panel (white or black in the first vertical column at the bottom left) and pull all three of the "Palette Brightness Controls" (RGB sliders) to the opposite positions (top to bottom, etc.), the system will flip black to white and vice versa.

Using the control panel, you should also select how long it takes for keys to start repeating, how quickly they repeat, how quickly you have to double-click your mouse buttons, and whether you want to hear the warning bells and keyclicks. You can't save the current time/date settings on the clock/calendar. Buy a battery backup clock-card if you want the proper time/date set automatically upon startup.

If you want to save custom colors and other control panel adjustments, you must have the control panel accessory (DESK1.ACC or CONTROL.ACC) on the disk with the DESKTOP.INF file when you boot the ST. Otherwise, it's system default settings as always.

This accessory is also important because it includes the "Install Printer" accessory, with which you can specify your printer's parameters. It is of special significance to owners of printers with 10-inch carriages, because the system default is for a 15-inch printer. Using Install Printer will let you reconfigure, so, when you save the desktop, the printer info will be recorded. Again, if this accessory is not present, the printer settings will be the system default (which can cause havoc in operations like printing the screen).

If you use the DESK2.ACC or EMULATOR.ACC file (click on "Set RS232 Config." and "VT52 Emulator"), you should now adjust its settings, as well.

The next thing to do is set up the directory windows to open in locations and sizes of your choosing. First, select your choices (such as "Show as text," "Sort by Type," etc.) from the view menu. Let's say you'd like to have a directory to drive A open automatically upon booting. To accomplish this, you'll have to open a window to that drive, then size and place it where you want. Next, open three other windows, size and arrange them as you'd like if you had to have four directories on-screen at once. Lay them down in the order that you'd like them to appear. Finally, since initially all you want is the window to drive A that you opened first, close all windows (in reverse order of the way you called them up!) except the first one. This action tells GEM where and what size you want the windows to be from now on.

Time for a test. Put the disk you intend to be your boot disk into drive A. Make sure that all your customizing is done for the moment and that the specified drive A window is open. Go to the options menu and select "Save Desktop." The computer will now create a DESKTOP.INF file on the disk in A. Before we proceed, if you've customized a lot of options, make certain you have the necessary accessories on the disk in A.

Push RESET. Tah dah! The desktop should look exactly as it did before you rebooted. Everything should be positioned, labeled and colored as you left it, and a window containing a directory of drive A should be open—sized and with items sorted as you specified. Now, open a window to any drive. Do this twice more, until four windows are open. They should all be sized and positioned as you had them when you opened them before resetting.

Note: if you had the control panel or some other accessories on the desktop when you saved it, they will not appear, as that information isn't recorded in DESKTOP.INF.

A little surgery.

For many, the above customizing will be sufficient, but there's more you can do. What we're going to cover now includes: permanently installing some applications, renaming the trash can, and even making it so that, when windows are opened, only specific files will be displayed and others will remain invisible! But first, students, let's examine the subject of our exploratory surgery. . .

The DESKTOP.INF file.

Let's review the case of a typical one. Such a file will look something like this:

```
#a000000
#b001000
#c77700070007000700552005055522 2
07705570750555077031111 03
#d
#E FB 02
#W 00 00 26 01 1D 17 07 A:\*.*@
#W 00 00 08 01 1D 17 00 @
#W 00 00 16 01 1D 0B 00 @
#W 00 00 0E 0E 31 09 00 @
#M 00 00 00 FF A DRIVE A:@ @
```

```
#M 00 01 00 FF B DRIVE B:@ @
#T 00 03 02 FF   TRASH@
#F FF 04    @ *.*@
#D FF 01    @ *.*@
#G 03 FF    *.APP@ @
#G 03 FF    *.PRG@ @
#F 03 04    *.TOS@ @
#P 03 04    *.TTP@ @
#G 03 04    NEO.PRG@ *.NEO@
#G 03 04    STWRITER.PRG@ *.STW@
```

[Note that the line beginning with *#c* is split for convenience in our magazine layout. It is one program line, with no spaces.]

Here's a (very) brief, line-by-line rundown of the example file here (the length of this file will vary, depending on changes you make). The first line stores the settings for the "Set RS232 Config." accessory; the second, the settings for the "Install Printer" accessory; the third, the colors and keyboard repeat rates for the control panel; the fourth is apparently reserved for future use; the fifth line contains the settings for how directories are displayed (set with the view menu), the copy/delete confirmation on/off settings, and mouse double-click response rate, as well as the resolution; lines six to nine are window parameters (note our auto-opening window to drive A—as evidenced by the *A:*.*@*). Also: lines ten and eleven in this example contain the coordinates, icon values, identifiers and labels for the disk icons; the twelfth line holds the same type of data as for the disk icons, this time for the trash can icon; the thirteenth and fourteenth lines tell GEM which files and folders to display icons for, when displaying directories with icons (fooling with these can potentially give you the most "explosive"—read: bombs—desktop you could imagine); the fifteenth through eighteenth lines tell GEM which types of files are in fact executable programs and what types of programs they are (GEM, TOS or TTP—see last issue's **Step 1** for more details on program types), while all others are considered nonrunnable data files. Finally: lines nineteen and twenty here contain the data to tell GEM which programs are installed applications and what the document file types are for each.

Now that we know our patient, let's start seeing what we can do for him.

Installing applications.

What this does, in a nutshell, is allow you to set up your system so that, when a specific data file type is selected, the program that uses it is automatically run, and the selected data loaded.

To install an application, you must single-click on a runnable program file, then choose "Install Application" under the options menu. A dialog box will open, displaying the name of the program you're installing and three blank spaces for a "Document Type." Below that are three boxes, one of which will be highlighted (the GEM box if you're installing a .PRG or .APP program; the TOS box if you're installing a .TOS program; or TOS-takes parameters if it's a .TTP program—again, see last issue's **Step 1** for more details).

Let's assume you're installing **ST Writer**. You single-clicked on STWRITER.PRG and selected "Install Application." Since it's a .PRG file, the GEM box should be highlighted. What the computer now awaits is your specification of a document type. This is determined by the three-letter extension attached to the filename (as in *.DOC* or *.NEO*). If you save all your **ST Writer** files with the extension .STW, that means you should type the letters *STW* (no period!) for the document type, then click on the OK box. Voilá!

In case you're confused, what you've just done is tell the computer that if you double-click on (or "open") any file with a .STW extension, it should run **ST Writer** and load the selected .STW file into it.

What good is this? It not only saves you the trouble of having to take the extra step of loading your file into **ST Writer**, but it means that you can remotely trigger the program from inside a subdirectory (folder). An example: my word processing disks are quite packed with a multitude of files, and I've organized certain types of files into folders, leaving only the **ST Writer** program and printer configuration data in the main directory. If I run **ST Writer** itself, I then have to tell it to index the disk files, add the subdirectory name(s), and *then* load the file from the folder. With **ST Writer** installed, all I have to do is open the folder of my choice, double-click on the file I wish to work on, and that's it! The system will get the data from the folder, as well as locate and run STWRITER.PRG from the main (root) directory.

This is the preferable way to use .TTP files. Example: if you have a .TTP program for converting low-res **DEGAS** screens to **NEO-Chrome** format, install the program and tell it that PI1 is the document type. Once that's done, all you have to do is double-click on a .PI1 file, and the conversion program will run, making a .NEO file for you!

If you were installing a program like **DEGAS** (not **Elite**), which creates files with the extensions .PI1, .PI2 and .PI3, it seems you'd be forced to install it three times, with each of these extensions as the document type. Not so. You can use a wildcard to indicate the character that changes. To wit: all the extensions begin with *PI* but end with one of three numbers. What's needed is a symbol to represent any character in that third position, so that all the file types will work. Wildcards are such symbols, and there are two you can use: the * (asterisk) and ? (question mark). (For more information on wildcards, refer to the **Step 1** article in the March **ST-Log**.) The * can represent any number of characters, so the extension .* would mean any extension would fit.

However, this is too general for our example, where we have only one character that changes. The ? represents one single variable character. So, by installing *PI?* as the document type, PI1, PI2 or PI3 files will all work equally well. The trouble is that, if you have .PIC files or the like, those can trigger the installed program, too. Be careful!

Note: not all programs can be successfully installed. In fact, **DEGAS** steadfastly refuses to be installed, but it was the best example I had for multiple document types.

Furthermore, in most cases, the installed program must be on the same disk as the chosen file. Otherwise, a dialog box will open, stating that: *This application can't find*

the folder or file you just tried to access. The way around this problem is documented in the section below, "Pathnames for installed applications."

Think of any programs you'd like to install, and do so. You can install more than one (I have no idea what the limit is. . . if there is a limit) and even install one program several times, so that it will run when a variety of file types are selected. But you should be careful not to install every document type you have, because that will make it impossible for you to double-click on almost anything without the computer trying to run a program!

Once you've installed all the programs you want, put your boot disk in drive A, make sure the system settings (control panel, etc.) and windows are displayed in the manner you want them upon booting, and click on "Save Desktop." This installation data will be saved at the end of the DESKTOP.INF file (see example file above).

Renaming the trash can.

Atari admits that the GEM trash can icon acts more like an incinerator than like its namesake. Why not name it INCINERATOR?

First, you'll need a scalpel. . .er, tool: a text editor that can save files in ASCII (defined in **A Baudy Tale** last issue). **ST Writer** won't work normally (see "Using **ST Writer**" at the end of this article). For this job, I'm using **1st Word** with the WP (Word Processor) mode *off*. If you have a text editor or word processor that has a nondocument or ASCII mode, you can follow along.

Just for safety's sake, copy your DESKTOP.INF file to the same disk (see last month's **Step 1** for an example). The computer will indicate a *NAME CONFLICT DURING COPY,* so change the copy's name to DESKTOP.FOO (don't change it to .BAK). This will ensure that a good copy remains, even if you make a gargantuan goof.

Run your text editor, make sure it's in ASCII mode and load the DESKTOP.INF file. Find the line with the name *TRASH.* Move your cursor until it is exactly on the *T* in *TRASH,* and type in your replacement name, up to eleven characters in length. (INCINERATOR fits exactly. Other good names include: ERADICATOR, ATOMIZER, DISPOSAL, SHREDDER and FILE 13.) If you're using an insert mode, you will now have to delete the word *TRASH,* but leave the @ symbol and the one blank space following it, before the carriage return. They're important! If you've overwritten both *TRASH* and the @ sign, retype the @ and following space. The line might now look like this:

```
#T 00 03 02 FF   INCINERATOR@
```

By the way, if you have TOS in RAM, and had to put underlines in place of spaces in the icon labels, you can now fix them in the same manner as you renamed the trash can—the data for them appears in the lines just above the TRASH label. Just make sure not to overwrite the drive identifier (A, B, etc.) and following blank space, which appear right before the icon label.

Now, save the file as DESKTOP.INF (**1st Word** will not overwrite the old file, but rename it DESKTOP.BAK). When that's done, cross your fingers and hit RESET.

If it turned out all right, the trash icon ought to have the label of your choice. If not, you probably did something wrong. In such a case, delete the current DESKTOP.INF file, and then try the modification again on the DESKTOP.BAK file, saving it as DESKTOP.INF when you finish. When everything's kosher, you can delete any and all DESKTOP backup files (.BAK and .FOO).

Un-installing an application.

If you've installed a program that you don't *want* permanently installed, simply don't use "Save Desktop"; press RESET, and you have nothing to worry about. However, if you did save the desktop, the solution (again) is to edit the DESKTOP.INF file.

Load DESKTOP.INF into your text editor, then go to its end, where the names of all the installed applications are. The last few lines might look like this:

```
#G 03 04   NEO.PRG@ *.NEO@
#G 03 04   STWRITER.PRG@ *.TXT@
```

If STWRITER.PRG is the program you wish to take out, just delete the line it's on, save the DESKTOP.INF file and reboot. That's it.

Or, if you decide you'd like to change the document type of an installed application, delete the old one (just before the @ at the end of the line) and type in the new one (such

as replacing .*TXT@* with .*STW@*). Always check to see if there are any blank spaces after the final character on a line before hitting RETURN, otherwise you could mess things up.

Pathnames for installed applications.

The major problem with installed programs is that the operating system always (with one exception) looks for them in the currently active disk directory. If you, like me, tend to keep a number of utilities constantly on file in one drive (like drive A or a hard disk), you may wish to have the system automatically go to that directory to find your utilities, rather than having to activate another window and then start looking in subdirectories.

For example, say you have the file management program UNSQUEEZ.TTP in a folder called UTILITY on the disk you always leave in drive A (or drive C, D, E, etc., if you use a hard disk). What you need to do, to make the system run it from there, is install the application in question in the normal manner (.?Q? is the document type; since squeezed files always have a *Q* as the second character in their filename's extension, I've used it as the one qualifier and filled the rest of the document type field with wildcards), save the desktop, then edit the newly added line, inserting the pathname and folder name (if any).

The original line would look like this:

```
#P 03 04   UNSQUEEZ.TTP@ *.?Q?@
```

The edited line would appear as:

```
#P 03 04   A:\UTILITY\UNSQUEE
Z.TTP@ *.?Q?@
```

As you can see, my addition consisted of specifying a drive identifier (A:\), then a folder (each folder name followed by a backslash (\) mark), before the name of the installed application. If your application isn't in a folder, you won't need to specify one.

For those of you who want to add such a line from scratch, note that the first three fields of an installed application line vary depending on the type of program:

```
#G 03 04 ..... .PRG or .APP files
#F 03 04 ..... .TOS files
#P 03 04 ..... .TTP files
```

If you mess up somewhere in this process, when you hit RESET, your ST may crash every time it reads the DESKTOP.INF file, and it will try to reset over and over. If this happens, turn off your computer, boot with another disk, then examine the DESKTOP.INF file and see if you can find what you did wrong.

Invisible files.

Well, not really invisible, but let's just say we'll make a selective directory. Suppose you have a demo disk on which are so many .DAT and .RSC files, that, in order to keep things readable, you decided you wanted *only* the .PRG files to show in the directory window. First, you format a disk for these demos, then copy your DESKTOP.INF file and all the demo material to it. Now, edit the DESKTOP.INF file.

In the following example from the DESKTOP.INF file, you can see that the line tells the computer to open a window to drive A. After the backslash (\) is an *.* (which stands for any filename with any extension).

```
#W 00 00 26 01 1D 17 07 A:\*.*@
```

To display only the .PRG files, replace the *.* with *.PRG, making the line look like:

```
#W 00 00 26 01 1D 17 07 A:\*.PRG@
```

And that's all. Or, if you wanted the disk to display only **DEGAS** picture filenames, you could change it to read:

```
#W 00 00 26 01 1D 17 07 A:\*.PI?@
```

This will work only on the specified windows, and will not affect a window that does not automatically open after booting. As always, when you've made your changes, reboot, and see what happens.

New program types.

Just above the lines containing the installed application data are four lines, each with the extension for a program type. If you change the name of one—like .TOS to .COM—you make it possible to run files with different extensions. Sadly, this disables the extension you've replaced. The solution: add another line with a program type indicated. To do so, make a copy of the line for the program type you would've altered, and change the extension (to make .COM programs run like .TOS ones, copy the line for .TOS files). Also, in this way you could run a program with no extension. To make BOINK.PRG just BOINK, you'd add this line, making it runnable under that name.

```
#G 03 FF   BOINK.@ @
```

Using ST Writer.

Not everyone has—or can easily get—a word processor like **1st Word**. If you only have **ST Writer**, it is possible to use that program to edit the DESKTOP.INF file, as I'll explain. First, rename DESKTOP.INF to DESKTOP .BAK. Run **ST Writer**, and set the top and bottom margins to *0*, the left margin to *1* and the right to *80* (refer to your **ST Writer** manual). Load the DESKTOP.BAK file (you'll have to wait a moment as the program "translates" it), and make your changes (as detailed above).

When finished customizing, count the total number of lines in the file (*not* counting the configuration data line at the top of the file) and multiply that figure by *2*. Next, enter the number you calculated after the CTRL-Y character in the configuration line at the top of the file, and delete the old value (probably *132*). Go to the program's main menu, press *P* to print, then *D* for disk output, and enter *DESKTOP.INF* as the output filename. Answer yes when asked to print the whole file. Next, save your **ST Writer** file as DESKTOP.STW, so you can call it back up and make changes if something goes wrong.

Quit the program, double-click on your new DESKTOP.INF file, select "Show" on the dialog box that appears, and make sure that the file contains all the data it should. Furthermore, if it looks compacted and you see the words *DO RUN RUN ST WRITER* at the top of the file, you either saved the file instead of printing it to disk, or you're looking at the DESKTOP.STW file. When ready, RESET your computer and see if your efforts have borne edible fruit. If not, reboot with another disk and try again.

Step-ping out.

Well, that's it for playing Digital Doctor Frankenstein. I certainly hope you're happier with the looks of your desktop than you were when we started. I know I am. For those of you interested in a byte-by-byte breakdown of the DESKTOP.INF file, I'll be uploading a comprehensive listing of the file's structure to the ANALOG Publishing Atari Users' Group on Delphi. One file on the topic already exists. To find my upload quickly, use the keyword *CUSTOMGEM* while searching in the ST programs section. The file should be up there by the time this sees print (I hope).

I've tried to make each **Step 1** a "stand-alone application," so to speak, but, as you've seen in the preceding text, it has become impossible to cover everything without referring to previous articles. There are so many fine points to ST and GEM operations that I have to send you back to previous pieces—just so I don't end up repeating the same information month after month!

Finally, response to **Step 1** has been favorable enough that "the powers that be" have granted my humble request to be allowed to write more "back to basics" articles. Never fear, **Step 1** will be back—as soon as I can figure out which topic to tackle next. If you have suggestions or comments for or about **Step 1**, drop a line to **ST-Log**. We'd love to hear from you.

Well, we've covered hard and soft wares, cracked GEM, and customized the desktop. . .Now what?

Addenda.

Here are a few corrections/updates on previous **Step 1**s:

—In **Cracking GEM**, I mentioned that it was dangerous to hit the ESCAPE key to clear the filename field while saving a picture. This was true of prerelease **NEO-Chrome**s (like version 0.5), but it's not true of the version now for sale (version 1.0).

—Also in **Cracking GEM**, when discussing installed applications, I said if your selected data file for an installed application was on drive B (or C, D, etc.), and the installed program itself was "in the window for drive A. . ." the system would usually find it. Wrong! The program merely has to be in the main (root) directory of the disk in drive A. . .regardless of whether a window is open or not.

—In **Soft Wares**, I stated you can't run software that uses GEM from the AUTO folder, because the ST hasn't run GEM when it runs the programs in AUTO. This is true normally, but now there are some programs which, in fact, make it possible for the ST to run programs using GEM from the AUTO folder. Amazing. Next thing you know, they'll be making Amigas run without crashing. . . //

Allergic to all things Commodore and never bitten by Apples, Maurice Molyneaux first purchased an Atari 800XL for animation work, but upgraded to an ST as soon as they became available. Currently slaving to complete the fifteenth draft of a science-fiction novel, he also masochistically churns out free-lance articles, artwork and animation on his ST, and hopes to dig out of a mountain of pending projects by the year 2000.

Desktop Napoleon

A look at ST combat simulations.

by Bill Kunkel, Joyce Worley and Arnie Katz

Whether on the ground, in the air, or under the sea, the weapons of war are among the most popular subjects for computer simulation. There's nothing the typical computerist enjoys more than piloting jet fighters, commanding submarines and guiding killer tanks through their death-dealing paces—as long as it's all kept within the bounds of a computer recreation.

A trio of recent ST releases falls under this category, representing not only different combat mediums (submarine, jet and futuristic supertank), but three individual approaches to the subject of tactical simulation.

GATO, from Spectrum HoloByte (1050 Walnut, Suite 325, Boulder, CO 80302—303-443-0191), puts the user in charge of a circa World War II attack sub, through a mouse-oriented system which stresses on-screen commands and multiple display screens.

The player first enters the day and month in which the mission is to be set (the year is fixed at 1943). Until the mission has been selected, the ship is in friendly waters, so the user can learn how to command a submarine without destroyers and patrol boats taking potshots at him. Once the player feels comfortable at the helm, he determines the torpedo-to-mine ratio and selects the "new mission" menu command. A coded radio message appears specifying the sub's objective, then the player's GATO-class undersea craft sets off across a 20-quadrant play area of water and islands.

The submarine's console is meticulously detailed. In addition to the usual battery of gauges and indicators (depth, speed, heading), rudder, periscope and torpedo/mine controls, a series of windows displays radar, damage, mine settings and a quadrant map. The primary on-screen display is, of course, devoted to the periscope or bridge view, which is rendered in a clear, if unspectacular, fashion. The ship runs either on its standard diesel engine, or via a battery, which must be recharged when not in use. The diesel is much faster, but doesn't operate below 20 feet, so the battery comes in pretty handy.

There are three types of missions: Destroy, Rescue and Mine. The Destroy scenario is fairly obvious, as is the Mine mission. In the former, the sub is assigned the task of destroying a specific ship or ships, while the latter requires laying mines in a predetermined area. The Rescue mission involves navigating to the position set at the start of the game, then maintaining the slowest possible speed while skirting the perimeter of the target shoal.

GATO differs from other submarine simulations in its emphasis on strategy. The number of torpedoes is limited, so, rather than blasting away at everything in sight, the user is forced to use some discretion in selecting targets. This is also true with regard to fuel and oxygen, which must be similarly conserved. When to fight and when to flee; whether to run silent and deep on battery power or go full blast via diesel power on the surface—these are the decisions which determine the player's success or failure.

Users who prefer a more action-intensive combat experience might prefer **High Roller**, a VTOL fighter simulation from Mindscape (3444 Dundee Road, Northbrook, IL 60062).

High Roller is a nickname for the Harrier Jumpjet, the best known example of VTOL technology. The VTOL (Vertical Take-Off and Landing) or V/STOL (Vertical/Short Take-Off and Landing) is a cross between a jet and a combat helicopter. Its ability to set down and lift off from a standing position gives it invaluable flexibility in combat situations. Its secret? The Harrier's exhaust nozzles are moveable, up to 90 degrees from the normal horizontal position, at which point the exhaust provides the ship with vertical thrust.

In the scenario provided by **High Roller**, a pack of enemy commandos have infiltrated a North African U.S. Marine base, where they've trashed an entire Harrier squadron—with the exception of a single ship which was being fitted with "FOF-TRAC" (Friend Or Foe Tracking Radar), an experimental tracking system. The user must take that final "**Roller**" up for an assault on the terrorist base located 500 miles away, in a sympathetic country well fortified with ground-to-air defenses and MIGs.

As is the case for many combat simulations, much of **High Roller**'s appeal is based on the fantastic array of high-tech pilot aids studding its console. In addition to the standard HUD (Head-Up Display), with its VSI (Vertical Speed Indicator), ASI (Air Speed Indicator), compass, gunsight/roll indicator, altitude, pitch, bomb/missile sight and Homer display, there's an instrument panel plus several keyboard commands. The instrument panel features a Multi-Function Display (thrust/power level, vector thrust, fuel, landing gear,

flaps, brakes and weapons inventory), AAR (Air Attack Radar) and FOFTRAC.

FOFTRAC is a continuously-updating display which tracks even the most subtle enemy target movement. As the Harrier enters new territory, it is necessary to first make a reconnaissance flyover, in order for the system to do its thing.

High Roller combines the most exciting elements of jet and helicopter combat. The Harrier can land on a dime, take off without a runway, hover almost motionlessly, then streak off in the blink of an eye—spitting missiles, bombs and cannonfire as it goes. Control is divided between joystick and keyboard, with a quick reference card supplied for user convenience.

The game's visual component is well handled, with the cockpit window serving as the primary graphic display. Ground targets and MIGs are easily discernible and movement—hover to high-speed—is nicely represented.

High Roller combines the gut-satisfying elements of arcade-style dogfights with the tactical underpinnings necessary to any good combat simulator.

Historically, the most popular format for combat games is the "hex grid," in which pieces representing troop units, artillery, etc., are moved over a board representing the battleground. This type of game format is represented by **Ogre**, a sci-fi contest from Origin Systems (340 Harvey Road, Manchester, NH 03103), in which the player goes up against the ultimate in cybernetic weaponry, the "cybertank," referred to colloquially as an "**Ogre**."

Created by Steve Meuse and Dallas Snell (who designed the **Ogre**'s Artificial Intelligence or AI), **Ogre** posits a future war scenario which has been revolutionized by many technological developments. The most significant is the invention of "biphase carbide armor" (BPC), a nearly indestructible lightweight metal. Just a few centimeters of this can stand up to anything short of a tactical nuclear bomb.

There have been several significant repercussions to the introduction of BPC. For one thing, human beings, encased in 4 centimeters of it, equipped with jets and armed with one tactical nuclear missile, have become virtual walking tanks. The tanks, on the other hand, have become *cyber*tanks. These juggernauts are up to 50 meters long, with atomic-powered treads capable of moving 45 kph or better, and an incredible *3 meters* worth of BPC armor! Worse, the most inhuman of war weapons is literally that: the **Ogre** is run exclusively by artificial intelligence. It is, as the excellent documentation tells us, "the first true war 'robot.' "

The standard battlefield configuration includes craters and piles of rubble—which act as boundaries and obstacles—throughout the top sixteen rows of hexes. The bottom six rows are unobstructed. The player uses the mouse to position pieces, which include a Command Post, Heavy Tanks, Missile Tanks, Howitzers, GEVs (Ground Effect Vehicles) and Infantry. **Ogre** comes with five preset fields, so that the fine art of deployment can be gradually learned.

Origin also includes a separate strategy booklet, which discusses the best battle plans, and features extensive notes on subjects like **Ogre**'s AI and how to exploit terrain.

Supertanks have been popular with computer gamers since Atari's **Battlezone** coin-op hit the arcades a half-decade ago, but **Ogre** is a tank of a different color. The player never gets inside the armored vehicles; he simply directs them, like a chess player. **Ogre** is a game of strategy (*planning* the battles), rather than violence (*fighting* the battles), and offers users yet another perspective on man's eternal folly.

Whether the battleground is on earth, underwater or in the air, whether the simulation is symbolic or realistic, our fascination with conflict is certain to continue. This love of conflict is, after all, the essence of all games, from chess to **Space Invaders**.

Besides, if computer wargaming becomes popular enough, perhaps they'll consider porting the "real" thing over to machines! ///

Famous Course Disk, Vol. II (Mean 18 Ultimate Golf)

ACCOLADE SOFTWARE
20833 Stevens Creek Blvd.
Cupertino, CA 95014
(408) 446-5757
Low resolution $19.95

by Bill Kunkel

Accolade has followed up on its promise to release "Course Disks" to interface with **Mean 18 Ultimate Golf**. Volume II offers a trio of faraway fairways for the armchair duffer.

Vol. II includes: the scenic Harbour Town course, dotted with tiny lakes and rich with maritime ambiance; the classic Inverness, with its eye-popping autumnal foliage; and the massive double-fairways of Turnbery.

These courses can *only* be played if the user already has the **Mean 18 Ultimate Golf** program disk, and it is easily interfaced through the use of extensive, click-on prompts.

The graphics are very nice, though, as with the courses available on the original course disk. The low-resolution graphics are not ultra-realistic, but possess a more stylized—sometimes almost cartoonish—look that may not be to the taste of all users.

Although this reviewer has never been to Scotland's bonny shores, the visual representation seems very faithful to the Inverness, etc., seen in scores of photos and TV transmissions. The courses are plotted in great detail and communicate a real sense of locale. ///

C-manship

The conclusion to our introduction to dialog boxes.

by Clayton Walnum

Last month, we took a look at the process of creating a dialog box. Unfortunately, we ran out of time before we could examine the example program. So everybody get out their issue 13 and turn to the listing. It's time to tie up all those loose ends.

The workings.

At the top of the listing, we include the file SAMPLE.H. This file contains the name of the object tree that represents our dialog box, as well as the names of all the objects within the tree. Each object is given a number. This number is the index used to find the object within the array. (A tree is an array of objects, remember?) For those of you who don't have a resource construction program (RCP), Listing 1 shows what the SAMPLE.H file contains.

If you do have an RCP, and you put together last month's dialog box, you may find that your objects are numbered differently than those in the example. Don't worry about it. That just means we constructed our dialog boxes a little differently. For instance, the example's object numbers don't run in perfect order. Some numbers are missing because, in the course of constructing the dialog, I removed a couple of objects from the screen without actually deleting them from the tree. Those objects were assigned numbers, but since they remain unnamed (and unused), they don't appear in the .H file.

The #defines in last month's listing assign to logical names the values of various parameters we'll be using when handling the dialog. Following that, we have the required GEM array declarations and some global variables.

The character array number_str[] is the string we'll use to change the displayed value of the NUMBERS object, in response to one of the arrows being clicked.

Finally, we get to the two structure types we discussed last month, OBJECT and TEDINFO. Of course, these are not standard C data types, right? These are our own data types, and we told the compiler this by using the typedef keyword in front of the declaration.

We also declare a pointer to our object structure. This pointer does not, at this time, contain an address. We've only told the compiler that when there *is* an address in *tree_addr, it'll be the address of a block of data of the type OBJECT. Also, we don't have an OBJECT or TEDINFO structure in memory yet. We've only stated what they'll look like. If this doesn't make any sense to you, take a little time to review structures and their declarations.

And speaking of the program . . .

Finally, we get to main(). There's not much to discuss here. In keeping with structured style, there's only a general "outline" of the program contained in this function. As you can see, our program must execute seven main steps. We've left the details of those steps to other functions. The function do_dialog() is where the action really begins.

The first thing we must do to get the dialog box on the screen is load into memory the resource file containing all the dialog's information. We do this with the call:

```
rsrc_load ("filename.rsc");
```

Here, filename.rsc is the name of the resource file that was created by the RCP (note that the extension *.RSC* is a convention you should stick to when naming resource files). This call returns a *0* if an error was encountered and a nonzero number if the file loaded okay.

Simple, no? In our case, we've used an if statement to prevent the program from continuing if the .RSC file is missing. If a *0* is returned, the conditional becomes true, in which case we call up an alert box, informing the user that the SAMPLE.RSC file was missing, then skip over the rest of the program and exit to the desktop. . .

Ahem. Well, that's the way it *should* have been. If you've ever run the program without the .RSC file on the disk, then you've surely noticed that when you leave the alert

box, your ST will complain—delivering two nifty little bombs.

This is due to an error on my part (you may send all expressions of sympathy to the editorial address). When program execution drops out of do_dialog(), the next item on the agenda (our "agenda" in main(), that is) is a call to print_results(). Whoops! If we didn't even manage to load the resource file, we certainly don't have any results to print! Silly me (he said, slinking under his desk). The call to print_results() should be moved to immediately follow the last form_dial() call in do_dialog(). Then you'll be able to exit the program cleanly.

Once we've got our resource file loaded, we need to find its address. We do this with the call:

```
rsrc_gaddr (type, index, tree_addr);
```

Here, type is the type of data structure loaded; index is the index of the object within the tree (in our case, the "object" *is* a tree); and tree_addr is the address of the information (the tree) loaded from the .RSC file. The parameter type should be a value from the following table:

0	object tree
1	OBJECT
2	TEDINFO
3	ICONBLK
4	BITBLK
5	string
6	image data
7	obspec (object specification)
8	te_ptext (pointer to text)
9	te_ptmplt (pointer text template)
10	te_pvalid (pointer to text validation string)
11	ib_pmask (pointer to icon image mask)
12	ib_pdata (pointer to icon image data)
13	ib_ptext (pointer to icon text)
14	bi_pdata (pointer to bit image)
15	address of a pointer to a free string
16	address of a pointer to a free image

The next thing we must do is modify the coordinates of our dialog box so that it'll appear in the center of the screen. This is done with the call:

```
form_center (tree_addr,&x,&y,&w,&h);
```

Here, tree_addr is the address of the object tree (returned from rsrc_gaddr()), and &x, &y, &w and &h are the addresses of the integer variables that will contain the dialog's centered X,Y-coordinates, width and height, respectively.

Since we'll be doing some work by hand, as it were, on the NUMBERS object in order to update the number it displays, we must find its eventual position on the screen. We do this with the call:

```
objc_offset (tree_addr,index,&x,&y);
```

Here, tree_addr is the address of the tree that contains the object; index is the object's index within the tree, and &x and &y are the addresses of the integer variables that will contain the object's coordinates returned from the function.

Now we must reserve space on-screen for the dialog. We have to do this so that, when we remove the dialog from the screen, GEM will be able to restore the display. The call:

```
form_dial (flag,s_x,s_y,s_w,s_h,
           l_x,l_y,l_w,l_h);
```

takes care of this, where flag is the operation you wish the function to perform (in this case, it should be *0*); s_x, s_y, s_w and s_h are the X,Y-coordinates, width and height of the smallest rectangle (we'll talk about this in a moment); and l_x, l_y, l_w and l_h are the X,Y-coordinates, width and height of the largest rectangle (the actual size of the dialog). The acceptable values for flag are:

0 FMD_START	reserves screen space
1 FMD_GROW	draws expanding box
2 FMD_SHRINK	draws shrinking box
3 FMD_FINISH	releases screen space and does a redraw

Next, we call form_dial() to perform operation *1*, drawing the expanding box. The call looks exactly the same as above, except the value of flag is *1*. The expanding box is drawn starting with the coordinates and size of the smallest rectangle, and ending with the coordinates and size of the largest rectangle. Note that the drawing of both the expanding and shrinking boxes is optional. If you wish, you can skip over this step and go directly to the call below, which actually draws the dialog. One reason you might want to do this is to bring the dialog up faster.

Finally, we're ready to draw our dialog, with the call:

```
objc_draw (tree_addr,object,depth,
           x,y,w,h);
```

Here, tree_addr is the address of the object tree; object is the number of the object to draw; depth is how many levels deep the object should be drawn; and x, y, w and h are the X,Y-coordinates, width and height, respectively, of the area of the screen in which the object will actually be drawn, also called the "clipping rectangle."

The clipping rectangle is the portion of the display to which all our screen output is limited. For instance, if we print text that'll extend beyond the rectangle's border, the text will be "clipped" to fit; anything that would be drawn outside the clipping rectangle will be ignored. Thus we can protect the integrity of the rest of the display.

When we set object to *0* in the above call, we're asking for the first object in the tree to be drawn. The first object in a tree is the root, in our case, the box containing the rest of the objects that make up our dialog. To be sure all the objects contained within the dialog are drawn, we *must* set depth to the proper value. If we had set it to *0*, only the main box would have been drawn. If we had set it to *1*, only the main box and its children would have been drawn, meaning that our radio button box would be missing its buttons and our number box would be missing its arrows. By setting depth to *2* in the sample program, the main box plus its children and grandchildren (the children of the children) are drawn, thus completing our dialog. If you want to be sure you get everything, just set depth to its maximum value of *8*.

Now that our dialog is on the screen, how do we find out what the user is doing with it? Simple!

```
form_do (tree_adr, object);
```

Here, tree_adr is the address of the object tree, and object is the index of an editable text field (*0* if there are no editable text fields). The value in object tells form_do()

the number of the text editable field we want to be active at the time of the call.

Now that we've made our call to form_do(), GEM will handle the dialog for us, highlighting any selectable fields clicked on and letting us enter text into any editable text field. When an exit button is clicked, the dialog will be terminated and form_do() will return the number of the button clicked. The button number is the *only* piece of information it returns directly. If we want to see what was entered in the strings, we have to hunt.

Obviously, only EXIT buttons will ever have their values returned from the form_do() call, so if you want to know when a button is clicked, make sure, when you design your dialog, that one of its attributes is EXIT or TOUCHEXIT.

Actually, that's not entirely true. There *is* another way to get this information. Each time we click a button or fiddle with the dialog in some other way, the object's status is changed and recorded in the ob_state member of the OBJECT structure. We'll see how to access the OBJECT structure in a while.

Because the only time we want to close the dialog box is when the OK or CANCEL button is clicked, we place the form_do() call within a do. . .while loop. In the body of the loop, we check to see which button was clicked, and perform the necessary action. The loop repeats, continually activating and deactivating the dialog, until OK or CANCEL is clicked. Note that the call to form_do() doesn't *redraw* the dialog; it only notifies GEM to accept more input from the form.

When the user has finished with the dialog, we must remove it from the screen. We do this by performing two more form_dial() calls: one to display the shrinking box (flag=*2*), and one to restore the screen (flag=*3*).

Finding the data.

In our sample dialog, when a button is clicked, all the program does is print the object's name. But when the user clicks on one of the arrows, we have to find a way to change the value shown in the NUMBERS object. This is where things get a little sticky. Your knowledge of pointers and structures is about to be pushed to the limit.

Think back to when you created the dialog. The object NUMBERS is a BOXTEXT object—a graphic box containing a string. It's this string that displays the value of num (the variable that holds the most current value selected by the arrows). When the user clicks on one of the arrows, we must increment or decrement num, then modify the string displayed in the object NUMBERS. In our program, the string we'll be modifying is number_str. But, until we change the pointer contained in te_ptext, the dialog doesn't know anything about *our* string. It's perfectly happy with the string we gave it at the beginning. So. . .

Last month, we discussed the OBJECT and TEDINFO structures. I told you that the OBJECT structure contained a field called ob_spec that holds object-specific information. When the object being described is of type BOXTEXT, ob_spec holds a pointer to a TEDINFO structure. And guess where we'll find that pointer we want to fiddle with? Yep.

So, let's say the user clicks on the up arrow. At that point, program execution jumps to the function do_up(), where num is incremented. The if statement makes sure the displayed value doesn't exceed *9999*. If num gets too big, we reset num to *0* and copy a new string reflecting this change into number_str. We then call edit_object(), the function that'll force NUMBERS to display the string *we* want it to, instead of its own.

Dealing with TEDINFO.

This is where things get tricky, so make sure your thinking caps are in working order.

The first thing we do in edit_object() is to declare a pointer, *ob_tedinfo, to a TEDINFO structure. Also, at the beginning of this function, we declare a ten-character string to temporarily hold the text we'll be setting up.

The first three lines of actual code in edit_object() convert the value in num to string form, then place this new string into number_str. We have to do all that fancy string handling to make sure the numbers are placed in the proper position, retaining any leading zeros, as well as the four spaces before and after the number.

That was the easy part. Now we have to find the pointer that points to the string contained in the NUMBERS object, so that we can change it to point to our own string. Remember that our tree, which is now pointed to by tree_addr, is an array of structures, each structure describing one of the objects within the tree. Just like any other array, it lets us get to a particular element by using an index. The object whose structure we wish to locate is NUMBERS, and, thanks to our handy RCP, NUMBERS has been #defined in the SAMPLE.H header file to the value of the index we need.

The ob_spec member of the structure that describes NUMBERS, contains a pointer to the TEDINFO structure that holds the pointer to our string. Yikes! I feel an illustration coming on. Figure 1 ought to help you sort this tangle out.

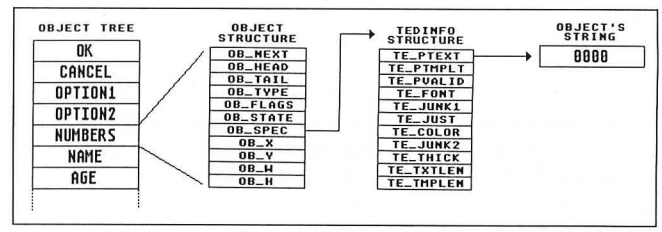


Figure 1.—Locating an object's text.

Got it?

So the address of NUMBERS's TEDINFO structure is:

```
tree_addr[NUMBERS].ob_spec
```

This is the address we store in ob_tedinfo (after casting it into a pointer to TEDINFO).

Now that ob_tedinfo points to NUMBERS's TEDINFO structure, it's a simple matter to get at the address of the string. The address is contained in the element te_ptext, so the statement below changes this pointer to point to our own string.

```
ob_tedinfo->te_ptext = number_str;
```

Once we have the string we created included as part

of NUMBERS, all we have to do is redraw the object with the call:

```
objc_draw (tree_addr,NUMBERS,
           depth,x,y,w,h);
```

Here, the parameters are the same as those described above. The height value of *16* used in the program is for medium resolution. To get the dialog to work properly in high resolution, since the monochrome screen has twice the resolution, you must change this value to *32*. To make your dialog work automatically in any resolution, you can get the values for w and h from the ob_w and ob_h members of the object's OBJECT structure.

Knowing who your friends are.

Last month, I said that the resource construction program was going to be one of your most valued tools. It saves the programmer an immense amount of time, by generating most of the data needed to bring a dialog box (or any other object tree) up on the screen.

But you don't *have* to use an RCP to program a dialog. You can do it directly from C—if you're the type of person who likes outrageously meticulous tasks.

Listing 2 will give you some idea of what I'm talking about. This listing was created by the Resource Construction Set that comes with the Atari Developer's Kit. Compare what you see with all you've learned about dialog boxes, especially with respect to OBJECT and TEDINFO structures. You should be able to identify many of the components of our dialog.

For instance, right at the top of the listing, immediately after the #define statements, are all the strings we need, including templates and string validation fields. Below that is the data for the ANALOG icons. Each icon consists of the actual data plus a mask, so we end up with four arrays, two for each icon. A little farther down, you can see the arrays of TEDINFO and OBJECT structures. Each line of these arrays makes up the data for one of the structures. Compare what you see with the structures illustrated in Figure 1.

Listing 2 appears here in exactly the same form as it was output from the RCS. There's still some information which needs to be filled in by the programmer. For instance, look at the first line of the TEDINFO array. The *11L* is the first TEDINFO's string pointer (te_ptext). Right now, this value is just an offset from the beginning of the strings defined at the top of the listing. If you start at zero and count down to eleven, you'll find that this te_ptext is associated with the string @ □□□□□□□□□□ (each □ represents a space character), which is the blank field for our NAME object.

The array of objects that make up our dialog is found right below the TEDINFOs. Just like the TEDINFO array, each line here consists of the data for each object's structure. Isn't this fun? How'd you like to code all this stuff by *hand*?

Closing up shop.

This finishes up our introduction to dialog boxes. By now, you should have a good idea of how versatile they can be. With a little creativity, you could write an entire program that used nothing but dialog boxes for all its input and output. Don't be afraid to experiment. Get as outrageous as you like!

See ya around. //

Listing 1.
C listing.

```
#define SAMPLE 0
#define OK 4
#define CANCEL 3
#define RADIO2 6
#define RADIO1 8
#define RADIO4 7
#define RADIO3 9
#define RADIO5 10
#define RADIO6 11
#define NAME 17
#define AGE 18
#define OPTION2 14
#define OPTION1 13
#define NUMBERS 19
#define DWNARROW 16
#define UPARROW 15
●
```

Listing 2.
C listing.

```
#define TREEOBJ 0
#define FREEBB 0
#define FREEIMG 4
#define FREESTR 29

BYTE *rs_strings[] = {
"",
"",
"CANCEL",
"OK",
"#2",
"#4",
"#1",
"#3",
"#5",
"#6",
"RADIO BUTTONS",
"@          ",
"NAME: __________",
"aaaaaaaaaa",
"@ ",
"AGE: __",
"99",
"OPTION 1",
"OPTION 2",
"    0000    ",
"",
"",
" ┃ ",
"",
"",
" ┣ ",
"",
"",
"THIS IS A SAMPLE DIALOG BOX"};

WORD IMAGE0[] = {
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
```

```
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF};

WORD IMAG1[] = {
0x7, 0xFFFF, 0xFFFC, 0xF,
0xFFFF, 0xFFFE, 0x1F, 0xFFFF,
0xFFFE, 0x3F, 0xFFFF, 0xFFFE,
0x7F, 0xFFFF, 0xFFFE, 0xFF,
0xFFFF, 0xFFFE, 0xFF, 0xF000,
0xFFE, 0xFF, 0xF000, 0xFFE,
0xFF, 0xF000, 0xFFE, 0xFF,
0xFFFF, 0x8FFE, 0xFF, 0xFFFF,
0xCFFE, 0xFF, 0xFFFF, 0xCFFE,
0xFF, 0xFFFF, 0x8FFE, 0xFF,
0xF000, 0xF1E, 0xFF, 0xF800,
0xE0E, 0xFF, 0xFC00, 0xE0E,
0xFF, 0xFE00, 0xF1E, 0xFF,
0xFF00, 0xFFE, 0xFF, 0xFF00,
0xFFE, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0};

WORD IMAG2[] = {
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF};

WORD IMAG3[] = {
0x7, 0xFFFF, 0xFFFC, 0xF,
0xFFFF, 0xFFFE, 0x1F, 0xFFFF,
0xFFFE, 0x3F, 0xFFFF, 0xFFFE,
0x7F, 0xFFFF, 0xFFFE, 0xFF,
0xFFFF, 0xFFFE, 0xFF, 0xF000,
0xFFE, 0xFF, 0xF000, 0xFFE,
0xFF, 0xF000, 0xFFE, 0xFF,
0xFFFF, 0x8FFE, 0xFF, 0xFFFF,
0xCFFE, 0xFF, 0xFFFF, 0xCFFE,
0xFF, 0xFFFF, 0x8FFE, 0xFF,
0xF000, 0xF1E, 0xFF, 0xF800,
0xE0E, 0xFF, 0xFC00, 0xE0E,
0xFF, 0xFE00, 0xF1E, 0xFF,
0xFF00, 0xFFE, 0xFF, 0xFF00,
0xFFE, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0};

LONG rs_frstr[] = {
0};

BITBLK rs_bitblk[] = {
0};

LONG rs_frimg[] = {
0};
```

```
ICONBLK rs_iconblk[] = {
0L, 1L, 0L, 4096,0,0, 0,0,48,24, 4,21,40,2,
2L, 3L, 1L, 4096,0,0, 0,0,48,24, 4,21,40,2};

TEDINFO rs_tedinfo[] = {
11L, 12L, 13L, 3, 6, 0, 0x1180, 0x0, 255, 11,17,
14L, 15L, 16L, 3, 6, 0, 0x1180, 0x0, 255, 3,8,
19L, 20L, 21L, 3, 6, 2, 0x1180, 0x0, 255, 13,1,
22L, 23L, 24L, 3, 6, 0, 0x1180, 0x0, 255, 4,1,
25L, 26L, 27L, 3, 6, 2, 0x1180, 0x0, 255, 4,1};

OBJECT rs_object[] = {
-1, 1, 20, G_BOX, NONE, OUTLINED, 0x21140L, 0,0, 58,14,
2, -1, -1, G_ICON, NONE, NORMAL, 0x0L, 4,1, 6,3,
3, -1, -1, G_ICON, NONE, NORMAL, 0x1L, 47,1, 6,3,
4, -1, -1, G_BUTTON, 0x41, NORMAL, 0x2L, 48,11, 8,2,
5, -1, -1, G_BUTTON, 0x43, NORMAL, 0x3L, 48,8, 8,2,
13, 6, 12, G_BOX, NONE, SHADOWED, 0xFF1100L, 2,4, 24,9,
7, -1, -1, G_BUTTON, 0x51, NORMAL, 0x4L, 13,3, 8,1,
8, -1, -1, G_BUTTON, 0x51, NORMAL, 0x5L, 13,5, 8,1,
9, -1, -1, G_BUTTON, 0x51, NORMAL, 0x6L, 3,3, 8,1,
10, -1, -1, G_BUTTON, 0x51, NORMAL, 0x7L, 3,5, 8,1,
11, -1, -1, G_BUTTON, 0x51, NORMAL, 0x8L, 3,7, 8,1,
12, -1, -1, G_BUTTON, 0x51, NORMAL, 0x9L, 13,7, 8,1,
5, -1, -1, G_STRING, NONE, NORMAL, 0xAL, 5,1, 13,1,
14, -1, -1, G_FTEXT, EDITABLE, NORMAL, 0x0L, 28,4, 16,1,
15, -1, -1, G_FTEXT, EDITABLE, NORMAL, 0x1L, 28,6, 7,1,
16, -1, -1, G_BUTTON, 0x41, SHADOWED, 0x11L, 27,8, 9,2,
17, -1, -1, G_BUTTON, 0x41, SHADOWED, 0x12L, 37,8, 9,2,
20, 18, 19, G_BOXTEXT, NONE, SHADOWED, 0x2L, 31,11, 12,2,
19, -1, -1, G_BOXTEXT, TOUCHEXIT, NORMAL, 0x3L, 0,0, 3,2,
17, -1, -1, G_BOXTEXT, TOUCHEXIT, NORMAL, 0x4L, 9,0, 3,2,
0, -1, -1, G_STRING, LASTOB, NORMAL, 0x1CL, 15,1, 29,1};

LONG rs_trindex[] = {
0L};

struct foobar {
  WORD  dummy;
  WORD  *image;
} rs_imdope[] = {
0, &IMAG0[0],
0, &IMAG1[0],
0, &IMAG2[0],
0, &IMAG3[0]};

#define NUM_STRINGS 29
#define NUM_FRSTR 0
#define NUM_IMAGES 4
#define NUM_BB 0
#define NUM_FRIMG 0
#define NUM_IB 2
#define NUM_TI 5
#define NUM_OBS 21
#define NUM_TREE 1
```

●

Catching the ST Sound Wave

An overview of MIDI software for your Atari ST.

by Charles F. Johnson

A revolution has been going on in the music industry. You see, a few years back, several manufacturers got together and designed a system by which all of their products could communicate. This meant that synthesizers, audio effects, even lighting systems could share the same data and be controlled by the same commands. The name agreed upon for this elegant data format was *MIDI*, which stands for Musical Instrument Digital Interface.

The MIDI standard allows communication between different machines by specifying sequences of 8-bit bytes. These command sequences can contain information about what notes are being played, whether the player has changed to a different sound, and other data. In addition, every individual synthesizer can send and receive MIDI messages for specific functions, like patch (sound) data, parameter controls, etc. By hooking several synths together, one musician can sound like a symphony orchestra—and still be playing only one keyboard. A synthesizer can be "tuned" to receive only one of sixteen possible MIDI channels.

The MIDI revolution really began when someone realized that a computer could easily manipulate the MIDI data, even to the extent of emulating a multi-track tape recorder. Atari realized the potential of the computer/MIDI marriage and included a MIDI interface—an option which is available only as an expensive add-on with other computers—as standard hardware in the ST series. With the advent of the low-cost Casio CZ-101, Yamaha DX-100, and Korg Poly-800 synthesizers (among others), a MIDI setup is no longer only for professional musicians. Once you've heard the kind of sound possible from a MIDI synth, it's difficult to be satisfied with the ST's internal sound chip.

There are now quite a few MIDI software packages on the market for the ST. I'm going to try to give you a breakdown of the types of software available, and what kinds of features you should look for.

Sequencers.

Sequencers are by far the most common type of MIDI software for the ST to date. A sequencer is a program that lets you enter MIDI data into the computer (either by playing a MIDI keyboard, or by using the computer), and then plays the music on your synthesizer. ST sequencers range from simple "playback" programs that play prerecorded pieces (like a player piano) to high-powered professional systems that can rival a 24-track recording studio in functionality.

When buying a sequencer that will record music, you should see what method it uses to enter data. Many of the less expensive ST programs do not allow you to "capture" music from a synthesizer in real time, yet this feature is a must if you have any keyboard ability at all. The best sequencers will have both real-time and step-time note entry. Step-time entry allows you to enter notes or chords one at a time, specifying the length as you go; this mode is very handy if you have little musical ability.

You should be able to adjust the tempo of the sequencer, to hear what you've recorded at different speeds. This can be one of the most powerful capabilities of a MIDI sequencer; even if you can barely play a keyboard, you can slow the tempo down to a point where the most difficult piece is easy. And, unlike the effect when doing this on a tape recorder, changing speed via MIDI does not affect pitch.

Most programs that allow real-time note entry also have some kind of auto-correct feature (often called "quantizing"). This lets you correct the timing of a recording to the nearest quarter note, eighth note, etc. If possible, you should check out the quantizing of a sequencer before buy-

ing it—not all of the programs out there do an equally good job of auto-correction.

Almost all sequencers for the ST give you more than one "track" to record on. The number of tracks is important . . . more tracks give you more freedom to experiment. You should be able to set each track to play only on a specific MIDI channel, so you can use more than one synth at a time, and you should be able to turn individual tracks off and on ("mute" and "un-mute" them). Along with multiple tracks should be the ability to combine tracks together onto one track, without losing any data from either original.

Editing is very important. Who wants to go back to the beginning and play the whole thing over again if you make one mistake? Sequencers should allow you to easily view and change (edit) the music data you've recorded in the computer. Many of the same editing concepts you find in a word processor are used in sequencers as well, like cut and paste, copy, move, delete, etc. In addition to these, a concept from the recording studio is usually available on sequencers, called "punching in." This lets you put the sequencer in "play" mode, and automatically start and stop recording at specified times. . .a very handy feature to have.

Make sure that your sequencer records all the different types of MIDI events, like pitch wheel, program change, velocity and after-touch. Other nice features would include: transposing of individual tracks or entire songs (shifting the piece to another key); micro-editing (the ability to view and change every single MIDI event that is recorded); copying and moving tracks or sections of tracks to different points in time; and looping functions to quickly build large pieces out of small sections. If you have more than one synthesizer, a MIDI "through" function is very useful. This allows you to designate one synth as the "master" keyboard and redirect its output to any one of the sixteen MIDI channels.

Patch librarians.

Most modern synthesizers have the ability to send patch data out through the MIDI port. This means that you can dump all the parameters which make up the sounds in your synth to another synth, or to a computer. The computer programs that allow saving and loading patches to disk are called "patch librarians," since they let you build libraries of synthesizer sounds on disk.

One of the nice things about synthesizers is that you can alter the sounds they make, usually by manipulating buttons and knobs on the instrument's face. Unfortunately, most synths display their sound parameter data in a one- or two-line window, which means you can only work on one thing at a time—and you can't see how changing one parameter affects another. Again, the computer comes to the rescue. A good patch librarian should have editing capabilities, and should put most (or all, if possible) of the parameter data on the screen at once. You should look for a graphics display of the sound envelopes, as well.

Librarian programs are usually dedicated to one particular synthesizer. There are drawbacks to this—you need to run different programs for each synth you own. You're out of luck if no one makes a patch librarian for your particular synth. A partial answer to this problem is the "generic" patch librarian. This type of program lets you save and restore the patch data from many different synthesizers, and write "configuration files" to handle any new synths that the program may not support. The generic patch librarians don't include editing features, since each synth's sound editing capabilities are unique.

A new breed of patch librarian program has been appearing on the market recently, one that can generate new sounds in the computer and send them to your synth. **DX-Android** and **CZ-Android** (both from Hybrid Arts) are two such programs, for the Yamaha DX and Casio CZ synths. These programs use Artificial Intelligence algorithms to make new sounds in a nonrandom way. In other words, the sounds thus generated follow certain rules, so that you don't end up with sheer noise; in fact, some surprisingly pleasing timbres can be created this way.

Scoring software.

If you play around with your sequencer enough, eventually you're probably going to want to print sheet music from what you've programmed. Scoring software uses a printer's graphics mode to print music. Unfortunately, as I write this I know of no real scoring software for the ST; nonetheless, here are some things to look for when the programs *do* start appearing.

There are lots of rules governing the transcription of music to paper. All professionally printed sheet music follows these rules, and scoring software should, too. When shopping for scoring software, it's not a bad idea to compare the output from the software with some real sheet music and see how it measures up. Things like tempo, crescendos, repeats, codas, etc., should all be clearly marked. Notes should be connected by beams in a logical fashion, not printed individually. You should be able to print text along with the music, as well, for things like song lyrics and tempo changes.

In a sense, people who have a background in computers are better equipped to deal with the complexities of MIDI/computer music than are most musicians. Music can enrich and enhance your enjoyment of computers (not to mention your enjoyment of life itself!), and, using the right software, you just might amaze yourself with your own hidden musical talents. It has become a cliche to talk about computers being an "extension of the human mind." But when music is created on a computer, wondrous things can happen. People who couldn't even keep time in a bucket can suddenly create beautiful pieces of music with perfect meter. Just as the printing press brought literacy within everyone's reach, computers and the MIDI standard are making music much more accessible to the common man. These are very exciting times we live in! //

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 cat. Charles is a SYSOP on the ANALOG Publishing Atari SIG on Delphi; his user name is CFJ.

Database Delphi

ST matters discussed in the Atari Users' Group SIG on Delphi.

by Matthew J.W. Ratcliff

Tom Hudson was on-line in January for an "ST Graphics" conference. The turnout was rather lean, but it was a *beautiful* Saturday afternoon—and everyone was washing their cars. It was still an interesting discussion, overall.

The edited transcript can be found in the Reviews and News database. Delphi seems to be *far busier* on Friday evenings, which is when our next conference will be held: "GDOS and the ST." I'll have a brief report on this March 6th conference next month.

Now, Daniel Moore and Sheldon Leemon have shed some light on programming with Megamax C. They explain what its real limitations are, and give us some good tips on programming techniques.

We'll also take a quick look at the new uploads in the ST programs database, plus a Delphi navigational note or two.

Megamax limits.

From: MOCKO (Mike Duppong)
To: DLM (Daniel Moore)

Could you fill me in on the 32K limit of Megamax C? What is the problem there, and how do you get around it?

From: DRX (Sheldon Leemon)
To: MOCKO

Because of the addressing modes used (and primarily as a holdover from the Mac, on which it was originally implemented), Megamax can only address 32K segments at a time. This means that no object module of your program code can be larger than 32K (you'd probably want to break down your source into a number of files anyway), and you have to use the "overlay" directive to put these code chunks together into one program.

It also means you can't define a single data structure that is larger than 32K. The way you get around this is to use system memory allocation functions like malloc to reserve a chunk of memory, and then use pointers to link that in with your data structure. Although some people are under the misapprehension that these "limitations" make it impossible to write a decent program in Megamax, remember that *all* Macintosh programs work within these restrictions, and there have been at least one or two useful programs written for that machine. Also, **Thunder!** (the standalone, not the desk accessory) was written in Megamax, so it appears that it's possible to do useful work with it on the ST.

From: DLM
To: MOCKO

There are actually several limits in Megamax. (1) A single code block (function) is limited to 32K, since PC-relative code is used. (PC-relative code is smaller and faster than absolute referenced code.) The solution here is to group related functions together and add the statement *overlay "name"* to the start of that file. You can have 10 code segments or 320K of code. (They aren't really overlays on the ST. The name is a carryover from the Mac version, which supports overlays—that's an OS feature on the Mac.)

The second limit is on the size of the global/static variables. They cannot exceed 32K, since they're addressed relative to an address register, (using negative offsets). Arrays/strings are addressed off the same address register, using positive offsets. I really don't consider that much of a limit, since very few programs have 32K of global data. And those that do usually dynamically allocate it using malloc.

The last limit is on the size of a single "object" (e.g., structure). Since all structure elements are addressed relative to an address register, there's a 32K limit on the size of the structure.

But then, who's going to define a structure that's 32K long?

Megamax is working on an absolute referenced version of the compiler, one that will remove those limits in exchange for producing larger/slower code.

C—a programming question.

From: KURTO (Kurt Oestreich)
To: ALL

Does anyone know how to pass a float out of a function with a return? I have everything defined as FP, but when I print out the data, it comes out as an integer.

From: MATRAT (Matthew J.W. Ratcliff)
To: KURTO

I'm a novice here, but I think she goes like this:

```
/send MATRAT ?


main()
{
     /* declare function too! */
     float var1,var2,mult();

     var1 = 5;
     var2 = 3.14159;
     printf("5 times pi is %f\n\n",
          mult(var1,var2));
}

float mult(a,b)
float a,b;
{
     return (a * b);
}
```

Database Delphi *continued*

The problem you're having is that a function is INT by "default." If you don't specify it in the function name, it will return an integer no matter what you do. In the above, I define the function mult as a float. This tells the compiler to set up the function to return a floating-point value.

Megamax and *getchar(), putchar()* troubles.

KURTO ran into a few other problems in learning to use his Megamax compiler, with getchar() and putchar(). Most of the problems were minor (syntax or capitalization). DLM did provide some advice that will help many programmers "come up to speed" in C programming on the ST.

From: DLM
To: KURTO

One of the best things you can do with *any* C on any system is print out all the system header files. There are lots of different ways to implement some things (getchar/putchar, buffered I/O, the CTYPE.H functions), and knowing which one a given C uses can save you a lot of headaches. Also, it lets you figure out the problems that can occur if you're not real careful about writing portable code. I move a lot of code between an IBM (Lattice C), ST (usually Megamax, but sometimes Alcyon or Lattice) and my Amiga (Aztec or Lattice). After looking at what all of them do, I've had no problems with porting code, with *no* changes for the target machine. . .

Also, printing out the headers will teach you a *lot* about how your C works. Another good idea is to look at the startup code (INIT.C for Megamax). It shows how memory is laid out during initialization and how a lot of general work is done. Very interesting. (And useful; reading it showed me a couple of trivial tricks to save 4 or 5K in program size for some programs.)

ST hard drive.

Daniel Moore has been a wellspring of information this month, indeed. If you have a hard drive, or are concerned about some of the "folder limits" you may have heard about regarding GEM, you should pick up the "ST Hard Drive" thread beginning at message number 14060. READ 14060, and then FOLLOW the thread. After that, just keep pressing RETURN until the END OF THREAD is reached.

ST Programs database.

Wake up! Asleep at the keyboard again, eh? I've literally dozed off at the ST keyboard many times, waiting for those slow disk-based compilers to do their job.

CANNONBALL (David Sheilbey) has sent up an Alarm Clock Desk Accessory, courtesy of MICRO-TIME Electronics. You can set up as many as five different alarm reminders (one for compile, the other for link, etc.) to go off from any GEM application. It also gives you a digital clock display at the top right corner of the monitor. A brief reminder message may be attached to each alarm.

Are you a new member of the **Publishing Partner** Addicts Society? Well, if you could possibly tear yourself away from the program, you'll find a PUBLISHING PARTNER FONT EDITOR program in the database, uploaded by CFJ (our own loveable, ever-productive Charles F. Johnson). Deron Kazmaier has placed his font editor in the public domain, so you can now create your own custom fonts for **Publishing Partner**. There's only one small hitch: Deron never got around to writing *any* documentation for it. If you have any problems, questions or enhancement ideas about **Publishing Partner**, just leave EMail or Forum messages to username DERON. He's now on Delphi, and we hope to see him there regularly.

MONOPOLY ST has been sent up by KIDX (Andy Eddy). It has only been up for a few days, but already has *many* addicted users. It was written and then compiled with GFA BASIC, which was reviewed in last month's **ST-Log**. It's already getting *rave* reviews among the users of Delphi; looks as if it may be the standard, to replace that obfuscated package called ST BASIC.

Delphi tips and tricks.

Conferencing is a *lot* of fun. Hidden away are a lot of slash commands (they all begin with a slash character, /) to do some neat tricks. After entering the CO, you can create your own conference by entering *JOIN* and a name, such as: *JOIN WILD PARTY*. You will then create a group called "Wild Party."

Once there, you can see who's in the neighborhood with the /WHO command. If anyone is in the CO area, his username will have parentheses () around it. If you would like to get me in conference, and my name is in the /WHO list, then enter a /PAGE MATRAT. If I'm available, I'll come chat for a while.

If you want to change your name (for the duration of the conference session), use the /NAME MYNEWNAME command. If you would like to rename your conference, use the /GNAME BIGGER WILD PARTY. Use the /HELP command for further conference command details. ///

Matthew J.W. Ratcliff is an Electrical Engineer in St. Louis, Missouri. He's been programming in BASIC and assembly language on the Atari since 1982. He's also active in telecommunications and is a remote SYSOP on the Gateway BBS, (314) 647-3290.

EXCLUSIVE PREVIEW

The Making of *ST-Talk Professional*

A look at version 2.0 in progress.

by D.F. Scott

It seems like ages ago, not just a year and a half, when we were extolling the virtues—those we could find—of ***Chat, Express!*** *(not the Keith Ledbetter product), and the original* ***ST-Talk*** *over the plain white Atari VT-52 terminal emulator accessory. At about the time some of us were convinced that* ***ST-Talk*** *was a solid product which gave ST telecom users everything they'd ever need, the pressure rose —there was a whiff of humidity in the air, and borne in on some very high clouds moving in from Canada came* ***Flash.***

In offering the manipulable, editable capture buffer, as well as the automated "macro" command sequence, ***Flash*** *presented telecommunications in a bold, new form, to which our concepts of function adapted themselves. Suddenly, these new features were no longer luxuries but necessities.*

Flash *also created a market environment in which competitors were either viewed as imitators or substandard entries.* ***ST-Talk****'s inarguably low price became its chief virtue—it was selling in some areas for $7.95. Now, John S. DeMar, President of Quantum Microsystems, Inc., and* ***ST-Talk****'s author, is in the process of "professionalizing" his product line, with a telecom program he hopes will trash* ***Flash.***

ST-Talk Professional *is not* ***ST-Talk.*** *It's a completely new program from byte 1. It is not* ***Flash****; in its execution and planned versatility, one will draw some, but not many, similarities. Yet perhaps most importantly, it is* not finished. *Beta test versions miss many of the features that may sway* ***Flash*** *aficionados' opinions upon* ***ST-Talk Pro****'s release.*

We talked to QMI's John DeMar at length, to gather from him the latest on ***ST-Talk Pro 2.0****, as well as to study the opinions of a programmer-in-progress. Here's the story, in his own words.*

The original **ST-Talk** did what we intended it to do; it was something reliable, easy to use, and at the lowest possible price. At that particular time, everybody was having all kinds of problems with **Chat**, **PC-Intercomm** was too expensive, and nothing really was in the public domain. We were there at the right time with that program, but **[ST-Talk]** was never meant to be anything that sophisticated. That program was written in FORTH. We don't use FORTH anymore; as a developing language, it had some limitations. Right now, we're using Megamax C, and a lot of 68000 assembly language. **ST-Talk Professional** is about 40 percent assembly language. Quoting Alan Page up on CompuServe, he said there isn't any assembly language in **Flash**, at least in the first version that they released.

You do have to have [assembly language] for running under GEM. There are a lot of speed limitations just because of the way GEM is written. For instance, text output to a window, under GEM, is pretty much limited to something less than 1200 baud, if you use the built-in routines. We have completely re-invented the wheel; we do our own complete screen driver, multiple fonts. We have a true 8-bit Atari ATASCII character set; so if you log onto the 8-bit boards, it puts out graphics on the screen which are the funky characters.

Visually, the program looks like a real nice GEM program; you're not used to seeing GEM programs with desktop icons like that until recently. **[ST-Talk Pro]** has a GEM window which is almost a full screen except for a strip down the right side, [where] there are seven cute icons for all the major functions—like dial, send, receive, capture-to-printer, terminal mode, and RS-232 options. You can still get all the functions from the drop-down menus, or you can get them by hitting alternate keystrokes, so it should please everyone. I've gotten a lot of feedback from selling 10,000 copies of **ST-Talk 1**. We gathered all this information from people who suggested they'd like to see this, that, and the other thing.

The text display, within the smaller GEM window, is still an eighty-column-wide display, with twenty-four lines—the typical number of columns and rows. If you normally wanted to put text inside of a GEM window, you could only get something like seventy-two columns, with all the gadgets around the outside. We're using a seven-bit wide font instead of an eight-bit wide font. When you first look at it, you say "Ah, that's nice," because the built-in ST font is a pretty poor font, as fonts go.

The mode that I use, on a monochrome monitor, you have an optional forty-eight-line mode, instead of twenty-four. What it really does is take the color font, which is seven lines tall, and uses that instead of the monochrome font, which is fourteen lines tall.

On the monochrome, you can pretty much read the 140-column text; but color, I wouldn't want to look at it all day. Either way, I wouldn't want to look at it all day. If you want to have true VT-100 emulation, and you call into a computer that switches into 132 columns—even if it's temporarily—if it doesn't work you're *stuck*. You might as well have not even purchased it at all. It's not something that you'd want to have running continuously, but if it's something you have to have and it doesn't work, it's no good.

Writing the font driver for that was not that tough, because all the work was done trying to figure out how to get this all working for the normal eighty-column driver. It's easy if the font is eight bits wide, because that's how memory's built. If you want to get the eighty columns within the GEM window, with all the scroll bars and things—a real GEM window, not a kludge—you have to go to a smaller pixel-wide font, drop down to seven bits. Anyone who knows about font drivers knows an odd number of pixels is a bad choice; but it had to be done, because if we went with six, then it would really look stupid.

The seven-bit wide font is pretty good. I've had people tell me they like it now that they like it now that they're used to it; when they go back to the normal text styles, they say those look really distorted or weird.

Anyway, if you're off an eight-bit boundary, you have to do all the bit shifting and masking, depending on whether the position of the character on the screen is overlapping, one byte into another. It turns out to be a really weird algorithm. Actually, that part of it was written as far back as last June or July, because that's the part that had to be implemented first, before I could say I could do a real GEM terminal program. If you can't put the text out, you might as well not write the terminal program running for GEM. I think that's the conclusion everybody so far has come up with on the ST, for writing terminal software.

People want true DEC VT-100, or better, emulation. Nothing that falls short of that is useful to a lot of people, especially universities and small businesses that have to access DEC computers. Our VT-100 emulator does most of VT-220, and not just VT-100. It does all the character attributes, including the underline; whereas normally with **Flash**, if you get any character attributes at all, it turns into inverse—no matter if it's bold, blinking, underline.

The [built-in Autopilot] language is one big interpreted language, if you want to think about it as that. You write a little program, using our language syntax, and you can run from hitting a keystroke, you can run automatically upon carrier-detect and call a phone number, or you can run automatically upon the phone ring. You could set one up to run automatically when you leave the program, if you wanted to. So this language is a very extensive language, it has FOR loops, and WHILE and UNTIL, IF-ELSE. It has a full language syntax. It looks a lot like BASIC when people glance at it; it has string variables and numeric variables. You have all the disk operating system functions.

The macro language—that in itself is a project in writing an interpretive language. That alone is currently running external to the [beta] code, and is about 20K of code. The program is about 70K right now, so that makes 90K altogether. With some other things, my estimate is about 100-120K of program. It's written in Megamax C, although there's lots of assembly language. Megamax lets you drop right into assembly language any time you feel like it. The screen driver is written in about 99K of assembly language; that particular module is about 33K of source code. It compiles way down to hardly anything, because it's written in assembly language. That allows us to get almost 9600-baud text output inside of a GEM window.

> *"ST-Talk Pro will ARC and un-ARC, and view an archived file... without leaving."*

One thing I'm trying to get somebody to do, as a demo to put on the disk with [**ST-Talk Pro**], is implement a limited bulletin board written under the [Autopilot] language. It's so sophisticated—but easy to use, still—that you can run a full bulletin board using the macro language under control of **ST-Talk**.

The other item people will like is that the autodialing [utility keeps] information on how long you called. You have to supply a cost-per-hour estimate, if it's a long-distance call; [**ST-Talk Pro**] will tell you on-screen how many dollars and cents you're spending. It will also log that when you hang up, keeping track of how much money you spend. You can print that stuff out—that's a lot of fun.

Another thing about the tally: some times I look at it—I have a lot of phone numbers—and I can't remember, "Did I find anything really interesting on this place before? Maybe I shouldn't bother calling this place long-distance if I haven't downloaded anything interesting from it." [**ST-Talk Pro**] keeps a tally of how many times you've downloaded from each place, so you can say, "Last month, I found some nice things there and there, so I can concentrate on calling these places and dropping those." That's for heavy users.

Today everything is ARCed [ARChived]. You have to un-ARC it, and you run the kludge of a public-domain shell (some work, some don't; some work better than the others), and you have to exit the program and go around the shell to un-ARC. **ST-Talk Pro** will ARC and un-ARC, and view an archived file from within **ST-Talk**, without leaving. You don't have to go outside it.

You can run any program from **ST-Talk** without leaving it. I've even run **VIP Professional**, things like that, without leaving **ST-Talk**. So if you're a heavy telecommunications kind of person, you can use **ST-Talk**, and then go run programs from within it.

Memory is definitely a limitation; it's getting better as Atari comes out later in the year with these bigger computers. Still, people have 520s—and will have 520s—for quite a long time, without memory upgrades. We have to take that into consideration. [The computer will] only let you run a program if you have space to run it; that goes without saying. But if you download some little public-domain thing, and you want to see what it looks like and don't want to have to log off, you can un-ARC it, run it, and come right back to where you left off—all the text is still there.

You can do [visual conferencing] with someone on a Mac and an IBM, where you have some previously-stored bit-maps that the person on the other end can call up with certain actions. It's only useful if you can, in real-time, change what that bit-mapped face looks like. The user on the other end should have control, and not have to say, "Here are my faces; put them on your disk ahead of time."

So what I tried to do was make [**ST-Talk Pro**] so that it works under most of these multi-user networks. You can't really use any escape sequences or control stuff. If you were writing your own service, you could put your own emulation for that kind of thing in there. That'd be nice to do in the future for multi-line bulletin board software. Right now, to make it work on any service, it has to be all ASCII text. The first thing that is bad about it is that people who are not running **ST-Talk** in that mode—or running another terminal program which doesn't have this kind of mode—[get all kinds of garbage].

I don't know if you know how the distributor pricing scales go—they expect 60 percent off. If I want to make any money at all, I have to really make sure that I don't spend too much on things like packaging and manuals, if I want to keep my product priced at $29.95.

If you go to a dealer, and he doesn't like you sitting there for an hour playing with the program on the computer at the store, how are you going to really know enough

about a hundred-dollar product? If it's $29.95—with the way that these things are discounted—if you buy it, even if it does not do what you want, you couldn't be too disappointed.

With software pricing, if you see a product that says $69.95, $79.95, you'll step back a couple of seconds before you buy it, compared to if it was $29-, $39.95, $49.95 in some cases, depending upon what you're getting. I think, especially with the way the market for the ST is—you know there's a mass-market there; it's pretty hungry and growing—if you're going to be greedy at the beginning, you're not going to be around.

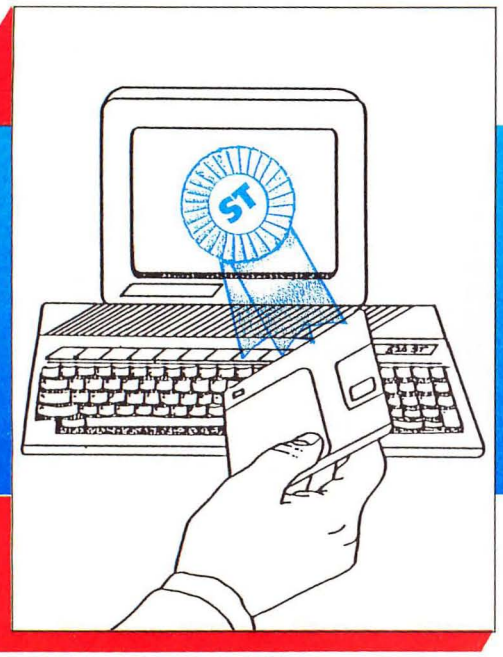
It's okay to come out with something at the beginning that you know is substandard—but if you do it honestly: price it accordingly, and then continually support it and upgrade it, and don't just pretend "this is a finished product worth seventy, eighty dollars." People complain a lot if we spend more time worrying about revising the packaging, to make it look better, than revising the product.

A lot of these [enthusiasts] have been disillusioned, and aren't doing ST or 8-bit stuff anymore, because things are a lot different than they used to be. It used to be, if you had a good idea and you were enthusiastic, you came out with it. To make some money, you have to be at a different level now; you have to risk a lot more in order to bring something into the market nowadays. Everybody expects [it to be] a lot better and a lot cheaper, and they expect it heavily advertised.

People may appreciate the fact that, when we're coming out with this stuff, we don't really advertise heavily—in fact, we never advertised **ST-Talk 1**. We really think it sold itself; we had some free advertisement from nice reviews—those are worth their weight in gold compared to paid advertising.

*Finally, in our discussion, DeMar stated he hoped that, at a price of $29.95 ($10.00 for **ST-Talk 1** owners who send in their old disks), people wouldn't bother to go through the time and trouble of pirating his work. He said, "Pushing a button that would blow up a million people a million miles away would be easier to do than if you took a gun and shot them all. It's the same kind of thing; if it's indirect, it's easier to handle. There's also this borderline kind of thing, do you really stop for a stop sign? People, I think, consider pirating stuff about as illegal or immoral as, 'Well, nobody's watching me, so what difference does it make?' "*

The difference it makes is in the survival of the enthusiast as programmer, the backbone of the Atari software industry. Let's hope this rather simplistic replacement for the copy-protection scheme becomes the one part of this program that isn't worth cracking. ///



The Challenge Met!

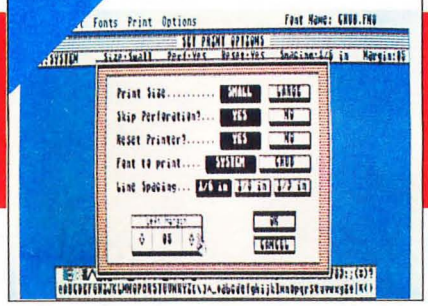
ST-Log **is proud to announce the winners in our ST Programming Contest.**

1ST

1st Prize $5,000

Paul Lay
for
Floyd the Droid on the Run

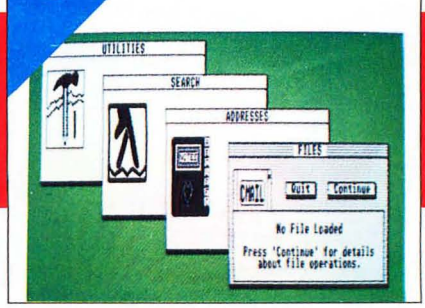
2ND



2nd Prize $2,500

Charles F. Johnson
for
ST Font Printer

3RD



3rd Prize $1,000

Peter Nichols
for
C-mail

Honorable Mention — $500 each

Guy Davis, Colin Faller and Mike Rupertus, for *Picture Puzzle, Khan* and *Flag Trivia*, respectively.

You'll be seeing these award-winning programs in upcoming issues of **ST-Log**. Our congratulations go to these programmers—and we thank all of you who submitted ST programs, whether chosen for publication or not. Keep up the good work.

A different kind of computer game

The acquisition of Batteries Included.

by D.F. Scott

In late 1985, when the 520ST was just gaining prominence, a company called Batteries Included, Ltd., of Richmond Hill, Ontario, Canada—just outside Toronto—decided that, to remain competitive in a metamorphosing computer market, it needed to develop an entirely new product line. To do that, Batteries Included needed money, as well as a reputable manufacturing press, to publish its software. Rather than take out a loan, BI's owners, Alan and Robbie Krofchick, decided to search for willing investors.

This is the story of BI's search for investment capital, and how, as Shel Silverstein once noted, some kinds of help we could all do without.

Among our sources for this story are: Trip Hawkins, President of Electronic Arts and new caretaker of the Batteries Included brand; Marty Herzog, former Vice President for Creative Services of Batteries Included and current Director of Creative and Marketing Services for Atari Canada; Ian Chadwick, former Director of Editorial Services for Batteries Included and current **ST-Log** *Contributing Editor; and Lindsay B. Swartz, President and Chief Executive Officer of Batteries Included.*

The opinions of Mr. Herzog and Mr. Chadwick expressed or interpreted herein are theirs alone, and not those of Atari Corp., Batteries Included, Ltd., or ANALOG Publishing.

The decision to seek outside investors was not conceived solely from the inside. Marty Herzog explains: "I think it was January of '86, or possibly even before that, [when] the original owners of Batteries Included were offered x amount of dollars for the company from Activision and Epyx. The [Krofchick] brothers were really not that interested in selling at that time, and basically walked away from the deal."

Activision, in working to create a multi-divisioned company, would later acquire Infocom, allowing that company to retain its creative integrity, while Activision took care of business matters like publishing and distribution.

As buy-out and merger courting persisted, and the money supply grew tighter, the Krofchicks changed their minds and decided to consider a buy-out, depending on its terms. "At the same time," relates Herzog, "Michael Reichmann, [then] president of Batteries Included, was looking for other investors to put money into the company, because we wanted to do much more in the lines of research and development—things we really couldn't do with the financial situation we had, because we were basically independent, privately owned, and in need of more money to do greater things."

So, apparently, being creative artisans more than financiers, the Krofchicks sought financial advice from their bankers, who suggested that a Canadian company should keep its assets in Canada. Herzog states, "The company found for us from a search of willing investors was ITM."

ITM of Calgary, Alberta, was affiliated with the software industry, and relatively unknown in the U.S.—or anywhere else. According to Herzog, ITM made some very persuasive promises to BI, such as "*great deals* of money would be invested into the company [and] no jobs would be lost, because they had no interest in the company other than as an investment—they knew nothing of computers, they wouldn't take anything out of the company, they would just invest money into the company. Of course, that sounded just great."

By comparison, other offers entailed a full-scale buy-out, or a trimming of officers from the payroll, or a transposition of jobs to other locations under new ranks, or a simple purchase of individual software titles. The brothers turned down these offers in favor of ITM's, which consisted of, as one source says, "$500,000 Canadian and a promise to stay out of BI's business." The half-million would be paid, read the agreement, by September of '86; the agreement was signed in June.

On the day of the buy-out, ITM's privately held stock reportedly increased in value over 75 percent, from around 45 up to 70 cents per share. But, in terms of actual value, the gain would only register on the New York exchanges as about a quarter-point increase—which is quite common.

Anxious to begin its new, aggressive research and development program, BI went back to the bank—this time for a loan. The bank, seeing ITM's promise in writing of a half-million by September, extended BI's line of credit and granted the loan.

Knowing the money would soon be available, BI began pursuing new authors and wrote contracts for the development of new programs, many for the ST and many advertised in print. By the fourth quarter of '86, BI was to have put together and released a complete ST software line, which included the Integral Solutions series of data-interchangeable applications and business management programs.

September finally came, but, by then, ITM had changed its mind. Instead of cash, ITM bargained to give BI executives shares of ITM stock, as what they termed "executive bonuses." ITM stock value had

since settled to near its pre-June figures. Did the brothers accept the switch, knowing they would still be in debt to the bank? "Yes they did—to the best of my knowledge. But it seemed like it was going to work, because ITM stock seemed to be fairly low-priced, and could only go up," says Herzog.

As a result of the delay in payment, followed by the switch in the *method* of payment, Herzog relates, "We had to renege on a number of contracts; so software that was promised, advertised and promoted never was completed. A number of titles were dropped, and a number of programmers were dropped, because we didn't have the money ITM promised that they would deliver.

"Mark Skapinker," Herzog continues, "as the Director of Product Development, was given a very, very nice job—something that a lot of people would like but Mark did not—it was to sit back and do nothing. He was the person who was supposed to make the deals. . .who investigated new software and accepted, rejected, or at least gave the advance money to proceed with it. His job was, at this time, almost negligible, because he had no money to develop anything with. All he could do was develop a lot of promises."

The six-month period between September '86 and February '87 was one of developmental stagnation. Release dates for **The Consultant** database and **B/Graph Elite**, both independently produced ST products, were continually postponed, while inside the company, development of the spreadsheet **BTS** was slowed. The success of BI would depend upon the sales of its recently unaugmented product line, which included **DEGAS Elite** and the ever-popular **PaperClip** for the Commodore 64 and Atari 8-bit line.

"As soon as the peak sale periods—the fall, Christmas periods—were over," explains Herzog, "and all the money had come in, ITM decided they were going to sell the company. They made overtures to Epyx, Electronic Arts (EA) of San Mateo, California, and a few others." EA President Trip Hawkins tells us his company approached BI several months prior to April, with the offer of not only publishing BI programs, but distributing them through its own network, as well, while allowing BI to retain its artistic integrity and internal structure, even as an ITM subsidiary.

Hawkins explains: "Part of our distribution philosophy is direct sales to retailers, as opposed to selling to distributors, who then sell to retailers. It's very important to the industry, because retailers need to have a direct marketing relationship with the publisher—particularly in home software, because [there] the market is driven by *variety* and personal tastes on the part of users.

"So we started up a program called the Affiliated Labels Program, about two years ago. What we do is look for companies that have a very good reputation for quality in their particular market area." Among EA's current affiliated labels are Origin Systems (which makes the **Ultima** series), Software Toolworks, Aegis and Interstel.

"We approached Batteries Included several months ago," Hawkin's continues, "because we admired their products; and of course, Batteries Included had been very strong all along in the Commodore 64 market, with their home productivity products. They also have a very nice line of Atari ST products, and a number of other products that have been very highly regarded, such as **Thunder!** and the **Isgur Portfolio** manager product."

As negotiations with BI and ITM continued, relates Hawkins, the terms of the deal changed: "The people at Batteries indicated to us that they were more interested in an acquisition, for a whole variety of different reasons. We started pursuing things along those lines and, basically, that's the way it turned out—we're acquiring the Batteries Included brand, which means we will be continuing to support it ourselves."

It is Marty Herzog's understanding that negotiations for the sale of BI took place only between EA and ITM. Says Herzog, "Soon as the deal was made last June, all rights or all decisions to the real direction of Batteries Included were in the hands of ITM; and that direction was wielded by saying, 'Yes, you can have money,' or 'No, you can't have money.' There was no development money; there was no 'futures' money. Everything that we did, we did with the money we could generate from *internal sales only*."

According to Herzog, when BI executives were informed that a buy-out deal was being reached, "there was a rather despondent attitude, not just toward Electronic Arts." EA was widely known for having produced software for certain computer brands *besides* Atari, for reasons referred to by some EA spokespersons as overall machine inferiority.

When asked if the BI acquisition represented a change in EA's corporate attitude toward Atari, in favor of a greater Atari emphasis, Trip Hawkins responded, "I would say it does. I think that one of the reasons we were interested in distributing the product in the first place was that Batteries Included has a very strong reputation for quality in the Atari ST market. **DEGAS** and **DEGAS Elite**, and some of their other programs, are very highly regarded, so we're delighted to have the opportunity to market those programs.

"I know there's been some concern," continues Hawkins, "in the ST community that we would want to stop doing that, but if that were the case, why would we have wanted to acquire them in the first place?"

In late February of 1987, it seemed the brand acquisition deal was drawing to a close. Batteries Included would remain in Ontario, and would serve partially as EA's Canadian office, while BI developed the programs for EA to publish and distribute—partly overseas through EA's new European office. Then, just before the pen hit the paper, on February 27, several BI employees received their termination notices. They were to leave their offices as soon as possible. The firings, contrary to employee anticipation, were *not* initiated by EA, for EA had not yet signed the papers.

Marty Herzog explains what happened that day: "The president [Swartz] walked into my office and said, 'Marty, very sorry, but today's your last day.' And I said, 'Okay,'—not that I wasn't expecting it—but I was actually quite shocked."

The firing scheme was systematic, and started near the top with the vice presidential officers, then worked its way down the ranks. Ian Chadwick, who reported to Herzog, was released on March 10. Under Canadian law, for every year of tenure an employee has served, he shall be granted one week's written notice prior to termination. Marty Herzog worked at BI for three and one-half years, yet he was told to leave in that many *days*. Says Chadwick of the whole affair, "Simple human courtesy was not applied. . .the staff at BI got screwed."

Says Herzog, "They gave me no separation papers, no severance, no vacation; they asked me for the company car back ASAP—gave me actually two days to get a new one—blamed it all on the receiver, and [told me I] have to make a claim for [severance pay]."

The receiver in this case was the bank—the same bank which granted BI the loan. One source believes ITM originally bought BI to use that company's assets as leverage to take out a loan of its own from that same bank.

Trip Hawkins says his company negotiated with the founders and part-owners of BI, the Krofchick brothers. Still, as he puts it, "the relationship [the brothers] developed with ITM allowed them to basically liquidate part of their ownership position at Batteries Included. It didn't develop into a situation where they were able to continue to raise the capital to fund it as an independent company.

"We wanted to help Batteries have a better market presence with those products; and as it turned out, the only way to do that has been to acquire the product line and the rights to the brand—which is what *the founders decided* they wanted to do."

We're told that, when the deal was about to be finalized, the bank foreclosed on the loan to BI it couldn't pay back, thus placing the bank (one source names Coopers & Liebrand Canadian Imperial Bank) in the role of "receiver" of BI. When the BI

firings took place, the receiver was credited (or blamed), and laid-off employees were told all claims had to be made against the receiving bank.

The deal, we're told, as it was originally drawn up, fell through. BI shareholders, as well as distributors, were given notices by mail, stating that in thirty days, Batteries Included would be no more—implying that Electronic Arts had backed out of the deal.

Then, when we tried to learn further details from BI on March 23, CEO Lindsay Swartz told us, "It's happening. Everything looks good." The deal with EA was being redrawn; on April 3, it was signed. EA had assumed the rights to the Batteries Included *brand*, the rights to manufacture and distribute BI's current product line, and the rights to negotiate with independent authors then under contract for BI.

So now Trip Hawkins is effectively the chief of Batteries Included. He lays out his plans: "We'll certainly be spending some money on the marketing of the products. In the short term, obviously, we're going to have to spend additional funds to get caught up on customer support. I think there are a lot of customers still waiting for their upgrades, and are still waiting to get their questions answered. Batteries was getting behind in that part of their business. So we're going to play catch-up and really hustle to try to get customer support, because we want people to continue to feel good about Batteries Included as a brand. We will be launching the product line through our own sales force, but, initially, we will be continuing to sell the product in the existing packaging. In fact, that may continue for some time."

EA will also acquire the rights to the Integral Solutions brand, which includes **I*S Talk** and the **Isgur Portfolio System**. Still, as Hawkins points out, "in terms of the full product line, there are quite a few products in it, some of which are quite old and no longer selling to any degree. We will be concentrating our attention on the products based on the demand; if there's sufficient demand for any product, we will continue to market it."

About those programs that were under development for BI but delayed by lack of funds, Hawkins says, "Programs that are being developed by software artists under contract—and making progress on their development—we are absolutely interested in the completion and the marketing of those programs." Concerning BI's project cancellations, Hawkins adds, "Sometimes the company determines that there's not sufficient demand to justify continuing the project, or perhaps they decide that, financially, they can't justify continuing and need to spend their money elsewhere.

"It's pretty common in this industry," continues Hawkins, "for companies to go out and say, 'Our hopes and dreams are to have all these products out," and then reality turns out to be somewhat different from that."

Hawkins is referring to the situation that arises when artists try to become businessmen. A software author generally assumes a role that's part craftsman, part mathematician. He has little time left, generally, to be an accountant, as well. He has no intrinsic concept of "market potential" or of a "targeted audience"; an artist is much more familiar with beauty and structural integrity.

So it is the craftsman's nature to find a business associate, an agent to keep the books and make the sales. Artists cannot be salesmen simultaneously, bringing forth their concepts of beauty while *explaining* them verbally. Artists have a tendency to believe a work of art should act as its own advertisement.

When the craftsman's work consists of tangible *single* objects—like a sculpture, wooden chair or steel fixture—the art of trade is a much easier matter. When processes of mass-production are invoked, every work of art can be duplicated 20,000 times. Justification must be found that it meets the needs or wants of 20,000 people. Therefore, the craftsman becomes less an artist, and more a market researcher or demographer.

Art as an industry has yet to find a feasible, workable format, though we are making progress toward that end. The development, or creativity, division and the marketing division of an enterprise may be separated and categorized, though they will not cooperate without a previously drawn set of guidelines. Several times in its history, Batteries Included was offered the opportunity to resolve difference between creativity and productivity.

At least in this instance, one observant corporation—Electronic Arts—managed, despite turmoil, to pick up the pieces, salvage the still-working components and begin to *recharge*. Still, the *corporation* that was Batteries Included is gone—employees scattered to new jobs, structural integrity destroyed by a loan foreclosure. The marketing division, in a sense, has eaten its own tail; the creativity division, however dismembered, remains. ///

REGULAR FEATURE

Assembly line

A column to teach assembly language for the 68000. This month: The Life of a 68000.

by Douglas Weir

We've now learned how the 68000 "runs" a program by fetching the program's codes, one by one, and performing the instructions interpreted from those codes. And what do these instruction codes tell the 68000 to do? Very simple things. In this respect the 68000 is a lot like you or me before our first cup of coffee in the morning. For the most part, it spends its time "reading" values from memory, performing simple operations on these values, and then "writing" the new values back into memory, at either the same locations or in new ones. For example, one instruction adds two numbers together, another multiplies two numbers, still another forces the 68000 to "jump" to another part of a program by loading the Program Counter with a memory address, and so on. But by far the most-used 68000 instruction is the one used to write and read values to and from memory—move.

So in order to add numbers *x* and *y*, we first need to move copies of them from memory to. . .where?

The 68000 has a group of special internal memory spaces which it uses to hold values it's working on. Unlike the computer memory we've discussed so far, these spaces are inside the 68000 and are thus a part of it. These spaces are called *registers*. Usually, a value must be in a register in order for the 68000 to be able to do anything with it. The Program Counter itself is a special-purpose register which can't be used for anything else, but the others—sixteen of them—are much more flexible.

Eight of the sixteen are "data" registers, meaning that they're intended to hold values being processed in some way. The other eight are called "address" registers, because these usually hold values which refer to memory locations. The two numbers we wanted to add would normally go into two of the data registers. We could then perform an add on them. The data registers are named and numbered *d0-d7*, the address registers *a0-a7*, and that's the way the programmer refers to them.

You may ask—how does the 68000 enforce the restriction of having only "data" in data registers and addresses in address registers? After all, both consist simply of numbers. The answer is—it doesn't really. You can put any numbers you like into these registers; the crucial thing is what you try to do with them after they're there.

Every instruction expects that the values it works with—its "operands"—will be in a specific place. Usually, this will be one of the registers. If you want to add two numbers, you must specify where the numbers are, and they'd better be in "legal" registers. If not, your source code will make no sense to the assembler, it won't be able to complete its translation, and it will give you an error message. Typically, a 68000 instruction will expect a value it uses as data to be in a data register, and a value it uses as an address to be in an address register. Of course, there are instructions that seem to operate directly on values in memory, but we'll ignore those for now.

Last month I said—or at least strongly implied—that the 68000 can handle numbers as large as 4 bytes, or 32 bits. So it's not surprising that its registers are each 32 bits "wide." In "ordinary" memory, each byte can be addressed separately. After all, that's what 68000 memory is—a series of bytes, numbered from *0* up. Registers, however, are somewhat different. Let's start by looking at a simple transfer of a byte value from memory to a register.

The right byte, code and mode.

If you want to refer to a byte in memory, all you have to do is specify its address. Let's say we wanted to read a byte of data into register d0. Assuming the byte was located at address 2304 (this number has absolutely no significance!), the source code would look like this:

```
move.b  2304,d0  get byte
```

meaning, "move the byte at location 2304 into register d0." Notice the .b after move. This suffix tells the 68000 to move only a byte. Since the 68000 can handle data up to 4 bytes—a longword—long, we have to specify the data size we want it to use in a particular operation. There are two other data length suffixes: .w and .l (lowercase L) for word (2 bytes) and longword size, respectively. So:

```
move.w  2304,d0  get word
```

means "move the word—the 2 bytes beginning at location 2304—into register d0." And:

```
move.l  2304,d0  longword
```

means "move the longword—the next 4 bytes beginning at location 2304—into register d0." If you don't include a suffix, the assembler always assumes a .w and generates an instruction to handle word-length data.

As I mentioned, you can normally access any byte in memory, simply by giving its address. If you're working with word- or longword-length data, however, there is a restriction. Remember our notebook model of computer memory from last month? When storing data as groups of 2 or more bytes, you have to keep track of where the groups begin and end, in order to read and write the data correctly. The 68000 has a very simple rule to help make sure this is done: all memory accesses involving data of word or longword length must be done at even memory addresses. If you break this rule and, for example, try to read a word of data from address 2305, the 68000's internal circuitry will stop you dead in your tracks with an "address error" (three bombs).

Even this safeguard doesn't make things foolproof. You can still access longword data at the wrong boundary—namely, in the middle of the longword, which is still an even address. But it's "loopholes" like this that make assembly language such a flexible medium.

Notice that we've now seen our first example of assembly code:

```
move.b  2304,d0  get byte
```

This shows most of the elements of a typical line. First comes the instruction itself (move.b), with its optional data-length suffix. This is followed by at least one space to allow the assembler to separate it from what comes next.

The operands (2304,d0) follow. Most 68000 instructions have at least one operand, and many have two—none have more than two. There's a standard way of referring to them. The one on the left is called the "source," and the one on the right the "destination" operand. That makes sense; a byte is being moved from address 2304 into the register. If we did it the other way around:

```
move.b  d0,2304  save it
```

the register would now be the source, and the memory location would be the destination. Two operands are separated by a comma.

The last part of our line of code (again, after at least one space to let the assembler know that the operand field is complete) is the comment. Comments are, of course, optional (mandatory comments were voted down by a silent majority at the last Assembly Programmers Convention). However, working in assembly language is a little like dreaming: by the time you're finished, you've usually forgotten what you were doing. Comments help you remember.

One important element wasn't needed in this example, so I left it out. Here's what the line would look like if it had a label:

```
loc:  move.b  2304,d0
```

You'll remember from last month that labels are one of the most helpful things about assembly language: they let us refer to locations in memory with a name we make up, leaving the actual arithmetic calculations to the assembler. Here the label refers to the location of the instruction which follows it. If we wanted to jump to this location from elsewhere in the program, we would only have to say:

```
jmp  loc
```

to do so. Labels are also used to refer to locations where data is stored. Instead of using address values like *2304*, etc., as we did here, we could have "labelled" these locations elsewhere in the program and simply used the labels—in fact, that's the preferred method. When labels are defined, they're usually followed by a colon, as above. The colon is omitted in other references, as in the jmp instruction.

Thus, the four fields of a line of assembly code are: label, instruction, operand(s) and comment. Many 68000 instructions do not use all the fields—some require only one!

One more thing, before we take a closer look at registers. The instruction move.b 2304,d0 moves the byte at location 2304 into the register. It does *not* move the number *2304* itself. If you wanted to do this, you would have to write:

```
move.w  #2304,d0
```

Two things have changed. First, we put a # symbol in front of 2304, to tell the assembler that we're referring to the value itself—in 68000 jargon, it's now an "immediate value." Second, we changed the instruction's size suffix from .b to .w. The number *2304* requires more than eight digits (bits) in binary representation, just as it requires more than, say, three decimal digits.

So far we've been pretty strict in our use of the term "address" as a number denoting a location in memory. However, in assembly language "addressing" is also used in a general way to describe any method of referring to an operand. Thus the 68000 manuals will tell you that in:

```
move.w  2304,d0
```

the 2304 indicates Absolute Addressing Mode, and in:

```
move.w  #2304,d0
```

the #2304 indicates Immediate Addressing Mode. There are officially twelve 68000 addressing modes, and we'll cover each one as we need to use it.

Anatomy of a register.

I said that registers are addressed somewhat differently from memory. The instruction:

```
move.b  d0,2304
```

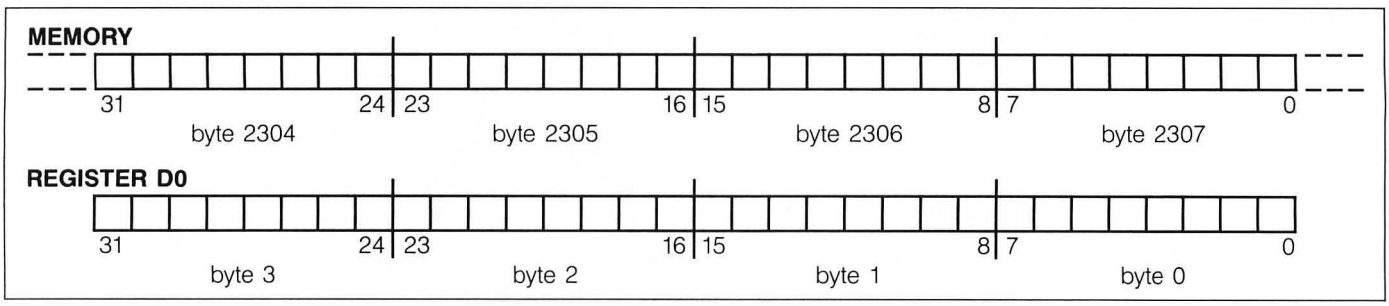


Figure 1.

allows us easily to access a byte from d0 and put it somewhere. But which byte? We need to look inside a register to find out.

Take a look at "Register D0" in Figure 1. First, some terminology. Remembering our binary numbers, we know that if bit 0 were "on" or "set" (i.e., bit 0 equals *1*), and all the other bits were "off," this register would then contain the binary number *00000000 00000000 00000000 00000001*, or *1*. If bit 8 (the "lowest" bit in the next byte) were on, and all the others off, then the number would be binary *00000000 00000000 00000001 00000000*, or decimal *256*. If bit 24 were on, and all the other bits off, our register would contain the binary representation of decimal *16777216*.

In other words, going from right to left, the bits—and thus the bytes—become "more significant" in value, just as decimal places in numbers become more significant going right to left—1000 is more than 100 (I know, because my personal banker told me so). For this reason, the leftmost byte in a 68000 register is often called the "most significant" or "high" byte, while the rightmost becomes the "least significant" or "low" byte. The same terms are also often applied to rightmost and leftmost bits.

Back to our discussion of how to address registers. Move.b d0,2304 moves the low byte (byte 0, nothing else) from d0. A move.w would move the low word (bytes 0 and 1). There are ways of getting at the other bytes in a register separately, but they involve some manipulations we haven't learned yet. The point is this: addressing bytes and words doesn't work the same in memory and with registers. If the above 4 bytes were in memory there would be no problem—each would have its own address.

Many ways to move.

To make things clearer, let's look in detail at our original byte move from memory to register d0 (Figure 1). If the 4 memory bytes beginning with 2304 happened to be holding a longword value, then our operation would take the high byte from the memory longword and put it in the low byte position in d0. Similarly, moving a word beginning at 2304 into d0 would result in the high word (again, assuming that 2304-2307 hold a valid longword value) being stored as the low word of d0. Of course, if we move a longword from 2304 into d0, then nothing is changed—all 4 bytes are moved and they remain in the same order.

If it helps, you can imagine the registers as a set of little upright cylinders, something like the devices bus drivers used to wear on their belts to hold change. When you put data in—from the top—it "drops" to the bottom of the cylinder. When you take data out, you do it the way the bus driver used to—from the bottom.

I should mention one thing. You can't use the .b suffix in operations with address registers. Only .w or .l are allowed. That's because a byte-size value is never a valid address to the 68000, although under certain circumstances a word can be.

Finally, don't let the term "move" confuse you. Moving a value from a register to memory, or vice versa, is very much like executing an assignment statement in BASIC or C: the value is copied from source to destination, but the original remains in the source location until explicitly cleared or overwritten.

We're now almost ready to take a look at our first complete program. But first, I'm so tired of typing out binary numbers that I think something drastic has to be done. And so, as promised last time, we'll take a look at . . .

Hexadecimal notation.

Changing careers isn't easy. Consider the case of Rodney Carp, star assembly programmer-turned-investment banker. Rodney's first task was to improve the accuracy of a seesaw company's balance sheet.

"What's this, Rod?" asked his supervisor, when Carp turned in a folder full of digits intermixed with little *A*s, *D*s, *F*s and who knows what. He winced at the answer.

"This isn't too hot, Rod. I asked for *extra* decimals, not hexadecimals!"

You can't really blame Rodney—hexadecimal numbers are such a mainstay of assembly language programming that it's sometimes hard to believe everybody else doesn't find them as useful. What are they? Essentially, nothing more than an abbreviated way of writing binary numbers.

If you look at, for example, the binary number *1000* and its decimal equivalent, *8*, there isn't any obvious relationship between the two. To convert one to the other, you have to go through the necessary calculations, which get more tedious the bigger the numbers are. That's because the base of decimal numbers, 10, isn't a power of the binary base, 2.

Hexadecimal notation is a way of writing numbers with a base of 16. This solves two problems. First, the base number is big enough that quite large numbers can be expressed with just a few hexadecimal digits. Second, since 16 is a power of 2 (2 to the 4th power, or 2^4, is 16), there is a direct correspondence between hexadecimal and binary numbers, and conversion from one to the other is very easy.

The mechanics of hex numbers are simple. Just keep in mind that we have to be able to represent sixteen different values with each digit. Since we only have ten "official" digits (*0-9*, the ones we use in decimal notation), we have to find six more from elsewhere. By common agreement, the first six letters of the alphabet have been co-opted for this task. Thus, to count from zero to fifteen in hexadecimal, you would say (louder than last time): "0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F." Decimal *16* would be *10* in hex.

For comparison, let's look at a binary number from last month. You'll remember that binary *11001* is the same as decimal *25*. In hexadecimal this would be written as *19*. You can convert that to decimal in the same old way:

```
 1 * (16** 1) =  16
+9 * (16** 0) =   9
                  25
```

But the conversion to binary is easy. Each hex digit expresses exactly four binary digits. To convert to binary, start from the right side of the hex number and convert each digit to four binary digits. Let's try the hex number *a1f0*, which at first looks forbidding:

convert hex *0* to binary *0000* (decimal *0*)
convert hex *f* to binary *1111* (decimal *15*)
convert hex *1* to binary *0001* (decimal *1*)
convert hex *a* to binary *1010* (decimal *10*)

This gives us the binary number *1010000111110000*. To convert binary to hex, just reverse the process: start from the right-hand side of the binary number, and convert each group of 4 bits to a hexadecimal digit. From now on, we'll usually write in hex any numbers whose binary representation is important, to save space and improve readability. Otherwise we'll just use decimal. To distinguish the two types, we'll always write hex numbers with a $, thus: *$4a*, *$ff*, *$10*, etc.

The program.

Well, that's enough talk. Take a look at Listing 1. These fifteen lines are a complete program. When run, it will print the words *Hello world* on your screen, then wait for you to hit a key before terminating. Use your favorite editor to write a text file (in "plain" ASCII mode) containing these lines, just as they appear here. You can substitute your own message for the *Hello world* between the single quotes, but leave the quotes as they are. Name the file TEST.S.

To assemble, link and run it, you need the programs AS68.PRG, AS68INIT, AS68SYMB.DAT, LINK68.PRG and RELMOD.PRG. With these five files and your source file, TEST.S, in the same directory, run COMMAND.TOS (all of these programs are included in the Developer's Kit, so if you have one you should have the others too). You will now be "in" TOS's type-in mode.

Typing the following lines will create a runnable program called TEST.PRG:

```
as68 -p test.s >test.lst
link68 test.68k=test.o
relmod test
```

The first line assembles the program and writes a listing file as well as an object file to the disk. If you don't want a listing, just type *as68 test.s*. Type the line, press RETURN, and wait for the assembler to run. Oddly enough, the assembler prints no sign-on or sign-off message—I couldn't confirm the rumor that this is because **AS68** is really a "generic" assembler originally intended to be marketed on late-night TV for $12.95 a copy. In any case, if there are no errors in your source code, it won't print anything at all, and the only way you'll know it's finished is that the TOS prompt will re-appear. If you request a listing file, errors will be printed there and not to the screen; so the quickest way to make sure everything is going according to plan is not to request one. If there are errors, go back to your source code and check for typing mistakes.

When the assembly is finished, type the next line and press RETURN. This will "link" your object file. Actually, no real linking is done here—you need at least two object files for that. However, this step is still necessary to the process of generating a file that TOS will recognize as a program. The linker does sign on. There should be no errors, unless you typed the command line wrong.

After the linker is through, enter the last line. **Relmod** outputs a *real* program—*finally*. What does it do? Officially, it generates "relocatable" code—a program that can be located anywhere (more or less) in memory. I'll have more to say about the linker and the relocater next time.

If you want to assemble and link programs out of GEM, the best way is to use batch files. There's an excellent explanation of how to go about this in Charles F. Johnson's **AS68 Helper**, in the March **ST-Log**.

You already know enough to understand some of the instructions and addressing modes used in this program. The others we'll discuss next time. ///

Listing 1.
Assembly listing.

```
*** this is a program!
          text                              code segment
          move.l    #test_str,-(a7)         get string address
          move.w    #9,-(a7)                code=display string
          trap      #1                      do it
          addq.l    #6,a7                   fix stack
          move.w    #1,-(a7)                code=wait for keypress
          trap      #1                      do it
          addq.l    #2,a7                   fix stack
          move.w    #0,-(a7)                code=exit program
          trap      #1                      do it

          data                              data segment
          even                              start with even address
test_str  dc.b      'Hello world',10,13,0     a string
*** ...and that's all there is!
```

INDEX TO ADVERTISERS

This index is an additional service. While every effort is made to provide a complete and accurate listing, the publisher cannot be responsible for inadvertent errors.

GREAT SOFTWARE
from across the pond

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

ST Karate★ $29.95

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

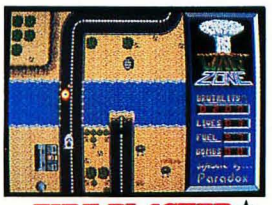
New **METROPOLIS★ $24.95**

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

WANDERER★

3D Stereo $39.95

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



FIRE BLASTER★
WAR ZONE★

midi music 2 Disks

$34.95

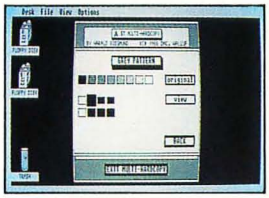
Amazing value! Two disk pack with two full colour ST arcade games! Fireblaster is the ultimate shoot-em-up with whole screen movement, multiple ships and over 35 levels. War Zone is a futuristic tank simulation with multiple features. An amazing pack for shoot-em-up fans.

ST PROTECTOR★
SPACE STATION★

midi music 2 Disks

$34.95

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



FLASH CACHE
FLASH BAK

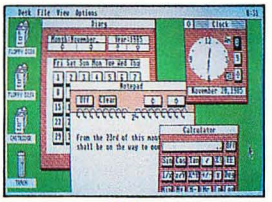
$79.95 GEM

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

PRO SOUND DESIGNER $99

Includes 8 bit sound digitizer mono version GEM

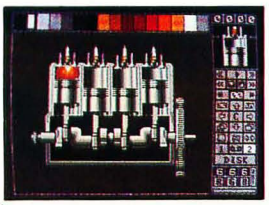
Quite simply the ultimate in sound control on your ST! A colourful VCR style interface ensures that anyone can use this program and be creating fantastic sounds within minutes. Full control over the AY-3-8910 sound chip is provided (with support for your own programs) as well as a superb 8 bit digitizer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



BACK Pack

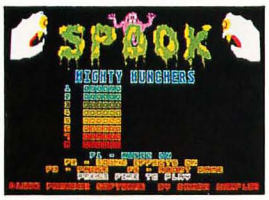
$79.95 GEM ROM

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



PRO SPRITE DESIGNER $59.95

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



THE ELDERSOFT EXTRAVAGANZA

$19.95

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

mini comm

$39.95 GEM

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

Tempus

$49.95

The Professional Editor GEM

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

To order or request product information please telephone

1-412-947-3739

VISA MasterCard

or write to the address below.

Eidersoft USA Inc
PO Box 288
Burgettstown PA 15021

PA residents add 6% sales tax.

EIDERSOFT
T·R·I·A·N·G·L·E
SOFT INNOVATION
Paradox

Shipping Info: Continental US free: Add $3 for COD
Paying Info: Credit Card (no surcharge), cashier check or money order.

★ colour only. GEM is the trademark of Digital Research Inc. ST and Atari are the trademarks of the Atari Corporation.

DEALERS/DISTRIBUTORS WE NEED YOU
Please telephone for dealer pack and full information.

Note we only sell high quality ST software that is supplied on single sided diskettes and is compatible with all 520/1040 ST computers. Mega ST compatibility please telephone.

CIRCLE #135 ON READER SERVICE CARD

Kill ~~Put on~~ a Happy Face!

New from Xanth F/X! The game that lets up to 16 people play at the same time on 16 different STs. All networked together using MIDI.

Design your own mazes using First Word™.
Select up to 15 happy face opponents in single player mode.
Requires 1 MIDI cable for each computer in network.

Only $39.95 suggested retail.

For a free product catalog and newsletter call or write us at:

Hybrid Arts, Inc.™

11920 W. Olympic Blvd., Los Angeles, CA. 90064
Computer BBS: (213) 826-4288 **Main Office**: (213) 826-3777 **Telex**: 4931231 (HYBRDUI)

Hybrid Arts offers a full range of MIDI products for the Atari ST including:
EZ-Track™: A 20 track polyphonic MIDI recorder.
EZ-Score™: A sheet music printing program.
CZ-Android™: A patch librarian-editing system for the Casio CZ keyboard series.
DX-Android™: A patch librarian-editing system for the Yamaha DX-7 and TX series.
GenPatch ST™: A generic patch librarian system for all MIDI devices.
MIDITrack ST™: A professional MIDI recorder availlable with and without SMPTE sync.
ADAP SoundRack™: A 16 bit digital audio workstation for music and broadcast professionals.

CIRCLE #106 ON READER SERVICE CARD