The #1 Magazine For Atari Computer Owners

A.N.A.L.O.G. COMPUTING

T.M.

FDC 50075

DECEMBER 1989
ISSUE 79

USA $3.95
CANADA $4.95

ST COVERAGE:

TURTLE GROOVES

DEGAS IMAGE GENERATOR

THE COMPUKID CONNECTION

8-BIT COVERAGE:

SKYRISER

ASSEMBLER/ EDITOR REFERENCE

TRUCHET TILES

REVIEWS:

HISOFT BASIC

BATTLE CHESS

THUNDER BLADE

SUMMER GAMES

DOUBLE THE STORAGE CAPACITY OF YOUR MEGAFILE HARD DRIVE!

THEY'RE BACK!

MONOPOLY
Parker Brothers Real Estate Trading Game
GO
COLLECT $200.00 SALARY AS YOU PASS
Clue MASTER DETECTIVE
More Suspects, More Rooms, More Weapons
RISK
Parker Brothers World Conquest Game
SCRABBLE
crossword game

ON A COMPUTER NEAR YOU

	IBM	APPLE II	MAC	C64	AMIGA	ATARI ST
SCRABBLE®	$39.99	$39.99	$39.99	$29.99	$39.99	$39.99
MONOPOLY®	$39.99	DEC		$29.99	$39.99	$39.99
RISK™	$39.99	$39.99	$39.99	$29.99	$39.99	$39.99
CLUE® MASTER DETECTIVE™	$39.99	$39.99		$29.99	$39.99	$39.99

Virgin
Leisure
Genius

Distributed by:- VIRGIN MASTERTRONIC INTERNATIONAL Inc.
18001 Cowan, Irvine, California 92714 U.S.A.
Telephone (714) 631-1001 Telefax (714) 631-1004

Virgin MASTERTRONIC

CIRCLE #101 ON READER SERVICE CARD.

EDITORIAL

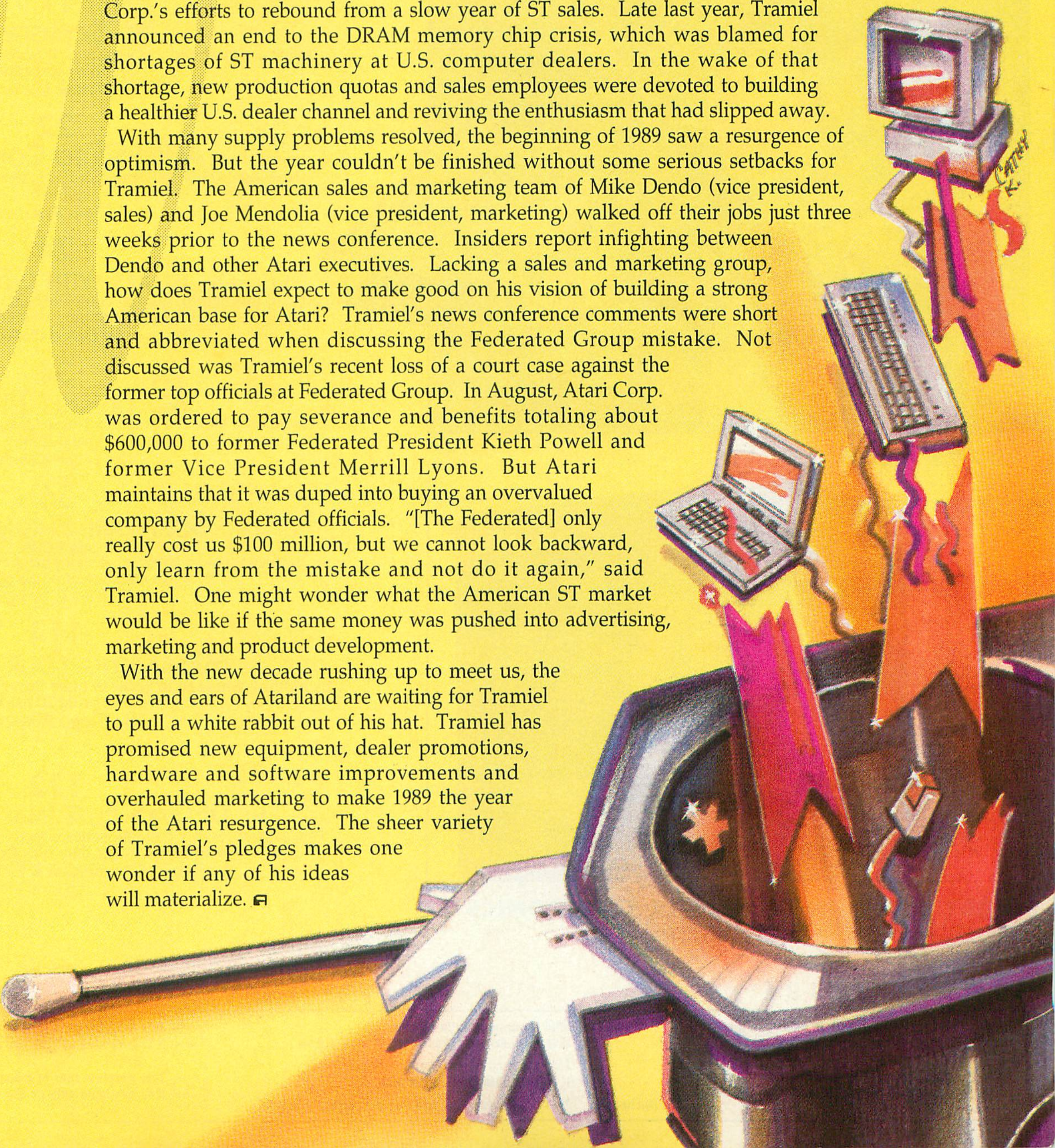
BY FRANK COHEN

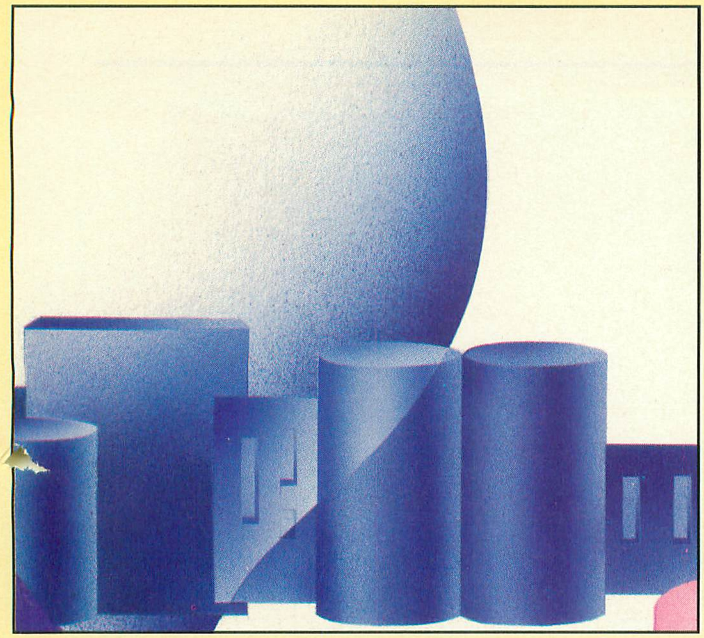
"We made a very, very costly mistake," said Sam Tramiel, president of Atari Corp., when asked about Atari's purchase and eventual dumping of the Federated Group retail chain. "We are computer/consumer electronics people, not retailers. In the future, we will stay in our field only."

Tramiel's words fell on a receptive audience of Atari enthusiasts at a recent news conference on GEnie, the modem information service. Questions ranged from TOS 1.4 operating system availability to technical specifications for new Atari equipment. Missing from the news conference, however, was mention of recent events confounding Atari Corp.'s efforts to rebound from a slow year of ST sales. Late last year, Tramiel announced an end to the DRAM memory chip crisis, which was blamed for shortages of ST machinery at U.S. computer dealers. In the wake of that shortage, new production quotas and sales employees were devoted to building a healthier U.S. dealer channel and reviving the enthusiasm that had slipped away.

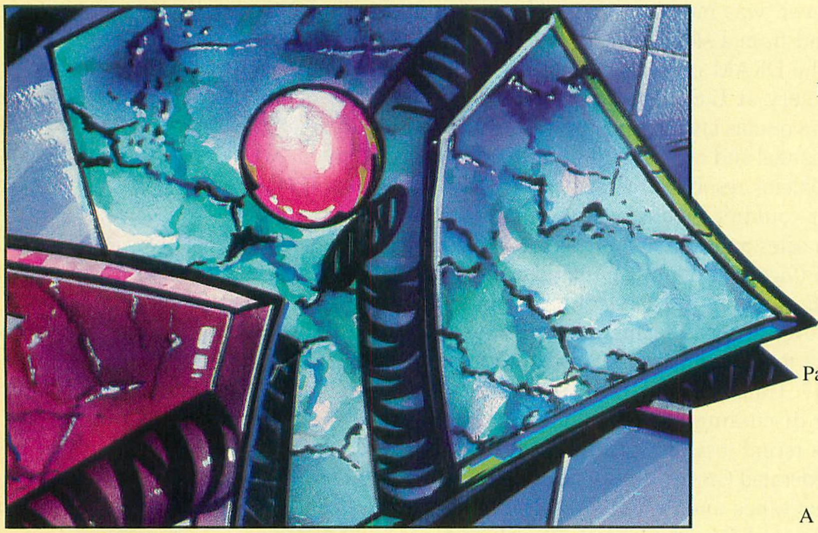
With many supply problems resolved, the beginning of 1989 saw a resurgence of optimism. But the year couldn't be finished without some serious setbacks for Tramiel. The American sales and marketing team of Mike Dendo (vice president, sales) and Joe Mendolia (vice president, marketing) walked off their jobs just three weeks prior to the news conference. Insiders report infighting between Dendo and other Atari executives. Lacking a sales and marketing group, how does Tramiel expect to make good on his vision of building a strong American base for Atari? Tramiel's news conference comments were short and abbreviated when discussing the Federated Group mistake. Not discussed was Tramiel's recent loss of a court case against the former top officials at Federated Group. In August, Atari Corp. was ordered to pay severance and benefits totaling about $600,000 to former Federated President Kieth Powell and former Vice President Merrill Lyons. But Atari maintains that it was duped into buying an overvalued company by Federated officials. "[The Federated] only really cost us $100 million, but we cannot look backward, only learn from the mistake and not do it again," said Tramiel. One might wonder what the American ST market would be like if the same money was pushed into advertising, marketing and product development.

With the new decade rushing up to meet us, the eyes and ears of Atariland are waiting for Tramiel to pull a white rabbit out of his hat. Tramiel has promised new equipment, dealer promotions, hardware and software improvements and overhauled marketing to make 1989 the year of the Atari resurgence. The sheer variety of Tramiel's pledges makes one wonder if any of his ideas will materialize.





MASSAGING YOUR MEGAFILE 114



TURTLE GROOVES 10

WHAT'S NEW IN CONSUMER ELECTRONICS 58

CONT

FEATURES

SYMBOL·GUIDE

C/M This program runs in both color and mono. It is available in type-in form & on the disk.

C This program runs in color only. It is available in type-in form & on the disk.

M This program runs in mono only. It is available in type-in form & on the disk.

C/M This program runs in both color and mono. It is available only on the disk.

(No image) This article does not have an associated program.

C This program runs in color only. It is available only on the disk.

M This program runs in mono only. It is available only on the disk.

8 This program runs on 8-bit Ataris

ENTS

REVIEWS

COLUMNS

DEPARTMENTS

ANALOG COMPUTING STAFF

Publisher
LEE H. PAPPAS

Executive Editor
CLAYTON WALNUM

Art Director
LEW BRYANT

Associate Editor
ANDY EDDY

Managing Editor
DEAN BRIERLY

East Coast Editor
ARTHUR LEYENBERGER

West Coast Editor
CHARLES F. JOHNSON

Contributing Editors
MICHAEL BANKS, FRANK COHEN,
MATTHEW J. W. RATCLIFF

D.A.BRUMLEVE, IAN CHADWICK,
FRANK COHEN, MAURICE MOLYNEAUX,
STEVE PANAK, MARSHAL M. ROSENTHAL,
GEORGE L. SMYTH

Cover Photography
GARRY BROD

Illustrations
DALE CORY
STEPHEN STERLING
CATHY KOSTREBA

Cover Illustration
MAURICE MOLYNEAUX

Copy Chief
SARAH WEINBERG

Copy Editor
ALYSON GOULD

Editorial Assistants
PATRICIA KOURY
CATRINA MASON
ROBIN THOMPSON

Chief Typographer
DAVID BUCHANAN

Typographers
B. MIRO JR.
QUITA SAXON
LIGAYA RAFAEL
LARRY GANNON

Vice President, Production
DONNA HAHNER

Production Assistant
STEVE HOPKINS

Advertising Production Director
JANICE ROSENBLUM

Advertising Production Coordinator
MAGGIE CHUN

National Advertising Director
JAY EISENBERG
(213) 467-2266
(For regional numbers, see right)

Subscriptions Director
IRENE GRADSTEIN

Analog Computing Published By L.F.P., Inc.

President
JIM KOHLS

Vice President, Sales
JAMES GUSTAFSON

Vice President, Client Relations
VINCE DELMONTE

Corporate Director of Advertising
PAULA THORNTON

Corporate Editorial
TIM CONAWAY

DECEMBER 1989
ISSUE 79

Where to Write

All submissions should be sent to: **ANALOG Computing,** P.O. Box 1413-M.O., Manchester, CT 06040-1413. All other editorial material (letters, press release, etc.) should be sent to: Editor, **ANALOG Computing,** 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210.

Correspondence regarding subscriptions, including problems and changes of address, should be sent to: **ANALOG Computing,** P.O. Box 16927, North Hollywood, CA 91615, or call (818) 760-8983.

Correspondence concerning a regular column should be sent to our editorial address, with the name of the column included in the address.

We cannot reply to all letters in these pages, so if you would like an answer, please enclose a self-addressed, stamped envelope.

An incorrectly addressed letter can be delayed as long as two weeks before reaching the proper destination.

Advertising Sales

Address all advertising materials to:
Paula Thornton — Advertising Production
ANALOG Computing
9171 Wilshire Blvd., Suite 300
Beverly Hills, CA 90210.

Permissions

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

Due, however, to many requests from Atari club libraries and bulletin-board systems, our new policy allows club libraries or individually run BBSs to make certain programs from **ANALOG Computing** available during the month printed on that issue's cover. For example, software from the July issue can be made available July 1.

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

In addition, any programs used must state that they are taken from **ANALOG Computing** Magazine. For more information, contact **ANALOG Computing** at (213) 858-7100, ext. 163.

Subscriptions

ANALOG Computing, P.O. Box 16927, North Hollywood, CA 91615; (818) 760-8983. Payable in U.S. funds only. U.S.: $28-one year, $54-two years, $76-three years. Foreign: Add $10 per year. 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.

For further information, write to **ANALOG Computing,** P.O. Box 1413-MO, Manchester, CT 06040-1413.

JE Publishers Representative

6855 Santa Monica Blvd., Suite 200
Los Angeles, CA 90038

Los Angeles	— (213) 467-2266
San Francisco	— (415) 864-3252
Chicago	— (312) 445-2489
Denver	— (303) 595-4331
New York City	— (212) 724-7767

ANALOG Computing (ISSN 0744-9917) is published monthly by L.F.P., Inc., 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210. © 1989 L.F.P., Inc. Return postage must accompany all manuscripts, drawings, photos, disks, etc., if they are to be returned, and no responsibility can be assumed for unsolicited materials. All rights reserved on entire contents; nothing may be reproduced in whole or in part without written permission from the publisher. U.S. subscription: $28 for one year (12 issues), $52 for two years (24 issues), $76 for three years (36 issues). Foreign subscription: Add $10 per year. Single copy $3.50 (add $1 for postage). Change of address: Six weeks advance notice, and both old and new addresses are needed. POSTMASTER: Send change of address to **ANALOG Computing** Magazine, P.O. Box 16927, North Hollywood, CA 91615. Second-class postage paid at Beverly Hills, CA, and additional mailing offices.

READER COMMENT

A Thank-You

Every so often there are comments in print about the poor support many software publishers provide for their products. I would like to take this opportunity to single out two companies for their excellent product support. They are ISD Marketing for *Calamus* and VersaSoft for *dBMAN V*.

I had been having problems with different functions of both products. After contacting both companies about the problems, both supplied me with updated versions of their programs. The only charge was postage, which is more than reasonable.

I live in a remote area of Alaska that is accessible only by air travel for most of the year. The nearest Atari dealer, Far North Atari, is 180 miles southeast in Fairbanks and is an excellent source of help. But they cannot be expected to provide software support. Having companies like ISD and VersaSoft makes it more enjoyable owning a computer.

—Ken Springer
Bettles, AK

Users' Groups, Where Art Thou?

Would it be possible for ANALOG Computing to periodically (even monthly?) publish a listing of Atari users' groups and bulletin board systems. Such a listing would serve as a directory to help put Atari "loners" in touch with other users in their areas while also giving users' groups and BBSs the exposure they need to attract new members.

—Edward Brown
Indianapolis, IN

ANALOG Computing *used to carry such a listing, and we think it might be a good idea to start it again. However, such a listing is possible only for users' groups; BBSs are started and discontinued so often that any listing we could print would be woefully out of date by the time it got to you. If you have an Atari users' group, send the name of the group along with the address to which people should write for information to:* ANALOG Computing, *Users' Group Listing, P.O. Box 1413-M.O., Manchester, CT 06040-1413. Make sure you tell us whether you're an 8-bit or ST group.*

DVT vs. ICD

This letter is in reference to Frank Cohen's article on the World of Atari Show at Disneyland from the September '89 issue of ST-LOG. First, I would like to point out that the ICD FA-ST Tape Backup does not use "audiotapes." While our data cassette uses the same type of "Phillips compact cassette" shell used by audiotapes, the media and hardware inside the shell are very different.

Frank's comparison of our ICD FA-ST Tape Backup system to the Seymor/Radix DVT also incorrectly leads people to believe that the DVT is faster than the ICD product. In spite of the claims made in their flyer, DVT is now said to make backups at 300K per minute, about one-third the speed originally advertised. That makes the ICD FA-ST Tape system about 2,000% faster.

—Tom D. Harker, President
ICD, Inc.

Convertor Problems

The *Ultimate Graphics File Convertor* (ANALOG Computing, May '89) listing was printed incorrectly. When I unscrambled the listing and typed it in, the program got confused when trying to load *Newsroom* files and also when going from Graphics 8 to *Newsroom*. Please talk to Dr. Brilliant and have this program corrected, since it seems to be a very useful utility.

—Dave Faxon
Kingston, NH

The Doc Replies

Yes, indeed, there are problems with UGFC. *The problem arose because I had a disk failure just before I finished the program. I thought I had reconstructed the program, but in fact I had used an earlier version, which was written to be used only with a specific* Newsroom *picture file. The lines below correct the problems. Load the original* UGFC *program and delete lines 5050 to 5130 and line 6015. Then enter the lines below and resave the program.*

—Lee Brilliant, M.D.
Granada Hills, CA

```
631 IF T0=4 AND COLR-COLL>247 THEN COL
R=COLR-1
632 RETURN
681 IF T0=4 AND COLR-COLL>247 THEN COL
L=COLL+1
682 RETURN
5050 COUNT=(N-1)/32:GOSUB 100:GOSUB 13
020:? #6;CHR$(125)
5060 DIR$(1)="♥":DIR$(7680)="♥":DIR$(2
)=DIR$:GOSUB 10000
5070 GOSUB 180:IF X=255 THEN 5200
5080 LO=X:GOSUB 180:HI=X:GOSUB 180:OFF
=X
5090 SECT2=SECT:BYTE2=BYTE:SECT=LO+256
*HI:BYTE=OFF+1:IF BYTE>128 THEN BYTE=1
:SECT=SECT+1
5100 GOSUB 10000:GOSUB 180:ROWT=X:GOSU
B 180:ROWB=X:GOSUB 180:COLL=X:GOSUB 18
0:COLR=X
5110 C=INT((COLR-COLL)/8)+1:R=ROWB-ROW
T+1:AD=ADR(DIR$):SIZE=C*R:COUNT=0:RCOU
NT=0:OFF=40*(ROWT-1)+INT(COLL/8)
5115 OFF1=OFF
5120 START=AD+OFF:TERMINATE=0:GOSUB 30
0:SHIFT=COLL-INT(COLL/8)*8:IF SHIFT=0
THEN 5125
5122 FOR N=1 TO SHIFT:A=USR(ADR(ROR$),
ADR(DIR$),7680):NEXT N
5125 A=USR(ADR(ORA$),ADR(DIR$),PEEK(88
)+256*PEEK(89),7680)
5130 SECT=SECT2:BYTE=BYTE2:GOTO 5060
5200 IF BUF$(BYTE,BYTE)<>CHR$(255) THE
N 5080
9005 A=USR(ADR(MOVE$),ADR(CURSOR$),153
6,LEN(CURSOR$)):A=USR(1560,0,8,191,247
)
9010 D=4:FLAG=0:COLL=8:ROWT=0:COLR=247
:ROWB=191:POKE 764,33:GOSUB 500
9142 D=COLL-8:IF D<0 THEN COLR=COLR-D:
COLL=8
```

Magniwriter Appreciation

Wow! This is great! Thanks so much. *ST Writer* was already my favorite word processor. Now you give me large print, too!

I am partially sighted, with about 4% of normal vision, if such a thing can be quantified. My optic nerve did not develop, or was partially destroyed, due to a high fever I had as an infant. To a normally sighted person, I imagine that my vision would seem to be like looking through the wrong end of a powerful telescope. Everything is small.

The Atari ST and your *Magniwriter* offer visually impaired people a low-cost alternative to other large-print computer-writing approaches, and may even help many to become more self-sufficient employmentwise. Thanks very much.

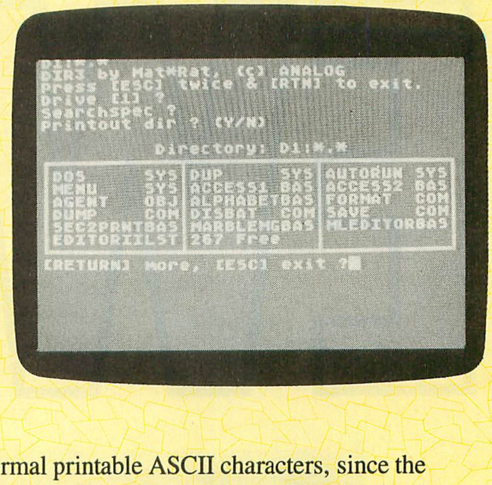
—Sam Wells
Glendale, CA

Sam, we are absolutely delighted that we've been able to provide you and other visually impaired Atari users with a program to make your word-processing tasks easier. Thanks for writing.

ALL LETTERS TO BE CONSIDERED FOR PUBLICATION SHOULD BE ADDRESSED TO:

ANALOG, READER COMMENT
P.O. BOX 1413-M.O.
MANCHESTER, CT 06040-1413

DIR3



```
D1:*.*
DIR3 by MatMRat, (c) ANALOG
Press [ESC] twice & [RTN] to exit.
Drive [1] ?
Searchspec ?
Printout dir ? (Y/N)

          Directory: D1:*.*
 DOS        SYS DUP        SYS AUTORUN    SYS
 MENU       SYS ACCESS1    BAS ACCESS2    BAS
 AGENT      OBJ ALPHABETBAS FORMAT     COM
 DUMP       COM DISBAT     COM SAVE       COM
 SEC2PRNTBAS MARBLEMGBAS MLEDITORBAS
 EDITORIILST 267 Free
[RETURN] more, [ESC] exit ?
```

by Matthew J.W. Ratcliff

DIR3 is a three-across directory listing utility for all Atari-compatible disk operating systems, including SpartaDOS X—the super DOS in a cartridge from ICD.

It is frustrating to list the directory of a disk, only to have the most important files scroll off the top of the display. For this reason, by using DIR3, three filenames may be listed across the Atari 38-column display, with the standard left margin set at the second column. There isn't room, however, for the file size or for the indicator (an asterisk) for a write-protected or locked file. If a file is protected, the first character of the name will be listed in reverse video. File size will remain a mystery, however. If this data is necessary, use your DOS's standard directory listing command.

When first loaded from DOS, DIR3 will display the default directory search specifier. You will be prompted for the drive number with the current default shown in square brackets. Type the desired drive number and press Return, or type Escape twice and Return to exit the program. Press only Return to accept the current drive number. You will then be prompted for a "search specification." Enter the template specification for the files you are interested in seeing (such as " * .BAS" for all BASIC programs). Do not type a drive specifier at this prompt. The drive number has already been set and will be merged with the specification. Press only Return to keep the default, displayed at the top of the screen.

If you would like a hard copy of the directory on the printer, answer yes to the next prompt by typing an uppercase or lowercase "Y" and pressing Return. Any other input is assumed to be "no." The directory will be listed to the display and to the printer if this option is enabled.

The filenames are formatted into three fields, neatly outlined by Atari's graphic characters for the display. Printer output uses normal printable ASCII characters, since the graphic characters would garble the hard copy.

DIR3 keeps track of how many lines it has output. If any filenames are about to scroll off the top of the display, you are prompted to press Return to continue or Escape to exit. The escape key will send control back to DOS without clearing the screen. (Some DOSs always clear the screen after a machine-language program has executed, however.) The return key will allow the DIR3 list to continue until another scroll prompt is necessary or the end of the directory is reached. This prompt is displayed after the complete disk directory has been listed. At this point, pressing Return will restart DIR3 with the previous settings as defaults. Press Escape to exit to DOS. Note that the scroll prompt is not presented if the printer has been enabled.

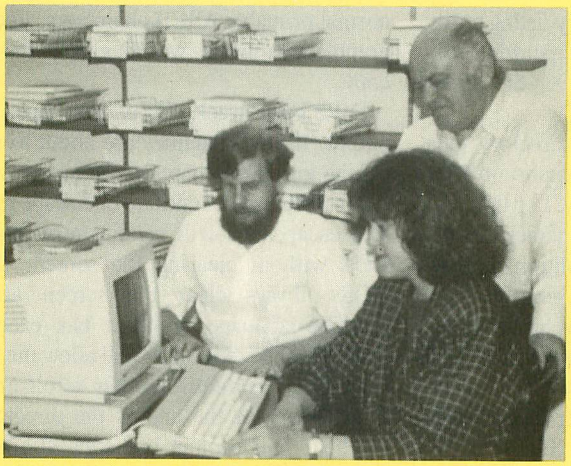
DIR3 is especially well suited to SpartaDOS X users, since this DOS can handle up to 1,400 files in a single directory. Because DIR3 will put 66 files on the screen before scrolling, Atari DOS users may list an entire disk to a single display. Most Atari-compatible DOSs have a limitation of 64 files per disk. ▪

DIR3 is especially well suited to SpartaDOS X users, since this DOS can handle up to 1,400 files in a single directory. Because DIR3 will put 66 files on the screen before scrolling, Atari DOS users may list an entire disk to a single display.

Matthew J.W. Ratcliff is an electrical engineer at McDonnell Aircraft in St. Louis, Missouri. An experienced assembly language, C and Ada programmer on IBM and main-frame computers, he still enjoys developing new programs and articles for the 8-bit Atari home computer. He has been an Atari enthusiast since 1982.

(PROGRAM LISTINGS CONTINUED ON PAGE 84)

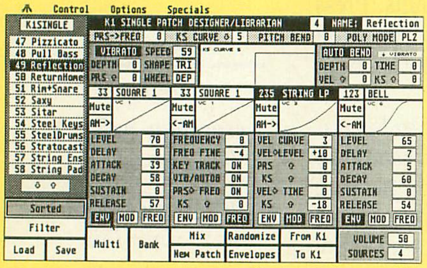
NEWS CLIPS



Atari Donation

Atari Corp. recently donated a complete Mega ST4 desktop-publishing system to Recording for the Blind (RFB), an organization that records books to tape for listening by the visually impaired. The system is used to aid RFB in keeping track of their many volunteer activities. The photo shows Jack Tramiel explaining the new system to Joanne Sweet, the RFB studio director. The shelves in the background are filled with book-recording projects in progress.

CIRCLE #131 ON READER SERVICE CARD.



K1 SINGLE PATCH DESIGNER/LIBRARIAN 4 NAME: Reflection

PRS->FREQ 0 KS CURVE ⇩ 5 PITCH BEND 0 POLY MODE PL2

VIBRATO SPEED 59 DEPTH 0 SHAPE TRI PRS ⇧ 0 WHEEL DEP

AUTO BEND ↓ VIBRATO DEPTH 0 TIME 0 VEL ⇧ 0 KS ⇧ 0

33 SQUARE 1 | 33 SQUARE 1 | 235 STRING LP | 123 BELL

LEVEL	70	FREQUENCY	0	VEL CURVE	3	LEVEL	65
DELAY	0	FREQ FINE	-4	VEL◇LEVEL	+10	DELAY	7
ATTACK	39	KEY TRACK	ON	PRS ⇧	0	ATTACK	5
DECAY	58	VIB/AUTOB	ON	KS ⇧	0	DECAY	60
SUSTAIN	0	PRS◇ FREQ	ON	VEL◇ TIME	0	SUSTAIN	0
RELEASE	57	KS ⇧	0	KS ⇧	-18	RELEASE	54

VOLUME 50
SOURCES 4

VOICE DEVELOPMENT SYSTEM TM

Music Software

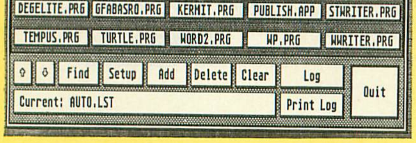
Musicode has announced the release of its K1-VDS voice-development system, a combination editor, librarian and sequencer designed for use with the Kawai K1 instruments. Features of this integrated package include quick displays, click-and-drag graphic envelope editing and organization of your patches into libraries and banks with sorting and filtering.

The sequencer is available from any point in the program, and new patches may be edited while the sequencer is running. The sequencer supports standard MIDI files for exchange of data between different MIDI systems.

Using eight edit buffers, the user may edit, randomize and mix patches. Patches may also be swapped between libraries and banks. All the program's functions may be accessed with the mouse or from the keyboard. The K1-VDS sells for $89.

Musicode
5575 Baltimore Drive, Suite 105-127
La Mesa, CA 92042
(619) 469-7194

CIRCLE #130 ON READER SERVICE CARD.



Thursday, September 14, 1989
7:28 AM
Current: AUTO.LST

GO - GO ST

Atari Go-Go

Maxwell CPU has started shipping their new ST program, *Go-Go ST,* which allows you to run any program with a single click of the mouse. Programs in a program list are displayed in the *Go-Go* window as buttons. Several program lists may be created and saved with one of them auto-loading at bootup. Also available when using *Go-Go* is a special "log" file that lets you keep track of how much time you've spent with each program, convenient for client-billing purposes. *Go-Go ST* is priced at $34.95.

Also available from Maxwell CPU is the MegSTender, a keyboard extension cord for the Mega ST line that comes in 6-, 8-, 12- and 25-foot lengths. Prices range from $16.95 to $24.95.

Maxwell CPU
507 W. Baseline
Lafayette, CO 80026
(303) 665-4849

CIRCLE #133 ON READER SERVICE CARD.

Tractor Tape

If you're getting tired of single-item feeding on your printer when you want to print such things as checks, business cards and envelopes, you might want to take a look at a new product from ProExcel, Inc. Their Tractor Tape can be attached to the sides of the items you wish to print, allowing them easy transport through your printer. According to ProExcel, the tape is safe and will not damage your print material. Better still, the tape is reusable. Using the same adhesive found on 3M's Post-It notes, Tractor Tape comes in 25-foot, ¾-inch-wide rolls and fits most regular tape dispensers. The price is only $4.95 per roll.

ProExcel, Inc.
P.O. Box 15236
Lakewood, CO 80215
(800) 748-3935

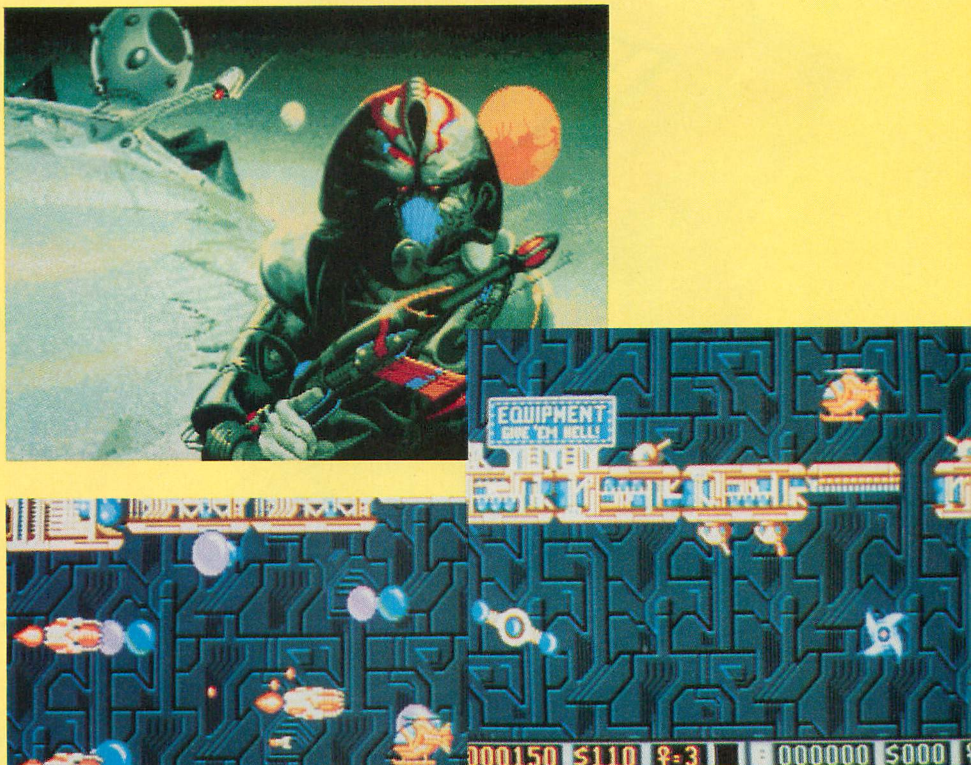
CIRCLE #132 ON READER SERVICE CARD.

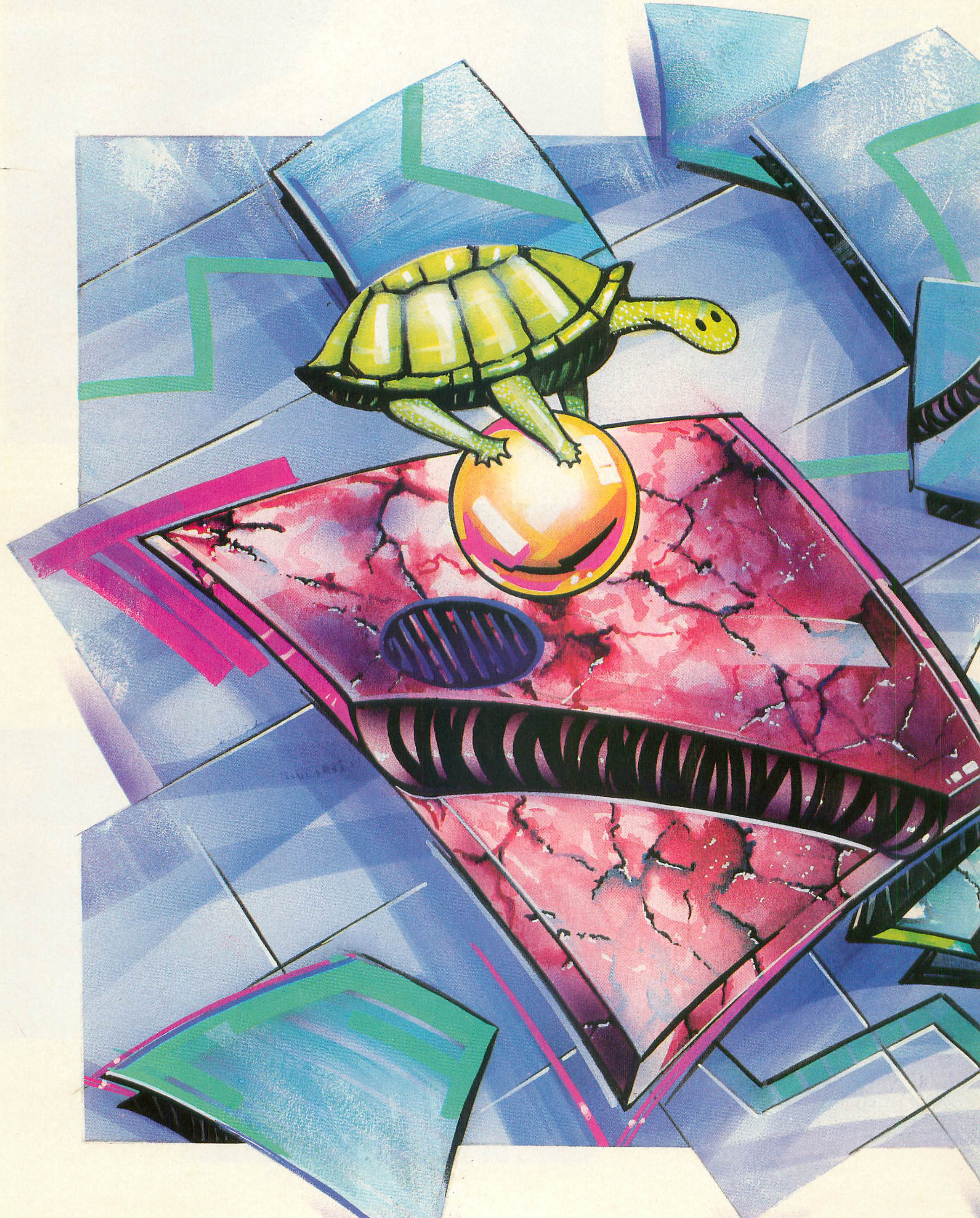
New Release From Psygnosis

From the guys who brought you *Menace*, a previous Psygnosis release, comes *Blood Money*, which, according to the manufacturer, "plunges you into a maelstrom of destruction that burns out your monitor with graphics in stunning colors, assaults your ears with the most exciting sampled soundtrack you've ever heard and challenges all your game-playing skills in ways you've never even dreamt possible." *Blood Money* has a two-player cooperative mode and involves using strategy as well as arcade action as you try to get through four alien-safari missions. Each planet features its own graphics. You can embark on this safari for about $45.

Psygnosis Limited
Century Buildings
Tower Street
Liverpool L3 4BJ
England

CIRCLE #129 ON READER SERVICE CARD.





TURTLE GROOVES

A GAME OF STEADY PANIC

BY GREG KNAUSS

Turtle Grooves is a simple game where the object is to keep colored marbles rolling along notches in tiles by rotating the tiles with the mouse—at least it sounds simple. But *Turtle Grooves* can become nerve-wracking as you skitter across the screen trying to keep three moving balls on wildly misshapen tracks.

Playing the Game

When the title screen comes up, press Escape to exit to the desktop, or the space bar to start the game. When you begin, a box will appear, asking which screen (set) you want to start with. You may choose any set you have already reached, so if this is your first game of the day, you must begin on set one. Pressing the mouse button from the title page will always start you on set one.

At the beginning of each new set, you are given a grace period in which you can rotate any tiles on the board, even the ones the marbles are on. Once the balls begin rolling (when the three-color counter at the lower right of the screen reaches zero), you will not be able to spin the tiles that have marbles on them.

To spin a tile, position the cursor over it with the mouse and click either mouse button, the right one rotating a tile right, the left one left. To finish a set, a marble must have rolled over every tile on the board, pushing it down. Once you complete a set, the next one will be put on the screen and your grace time will be shorter.

After a few sets, a second marble will join the first, and a few sets after that, a third will join the first two.

The Editor

From the title screen you can press Return to enter the *Turtle Grooves* editor, where you can alter existing tracks or add your own.

While in the editor, position the cursor over a tile and press either mouse button. You can cycle through all the different tile types this way and build your own tracks.

Press F1 to load an existing set. Use the arrow keys to select the set number you want and press Return to load it. Escape aborts this operation.

Press F2 to save the set you are working on. Like loading, use the arrow keys to select the set number you want to save the current screen as. You will be allowed to select a number that is one past the last set on the disk to add this track onto the end of the existing ones.

F10 will clear the screen of the current set, and Return will take you to the title screen of the game.

(CONTINUED ON PAGE 105)

BY MATTHEW J. W. RATCLIFF

ASSEMBLER /EDITOR REFERENCE, PART II

Debugging

Your assembly language programs are bound to have some bugs. Luckily, Asm/Ed provides a method for testing assembled object code. When at the Asm/Ed EDIT prompt, type BUG and press Return. You'll be presented with the DEBUG prompt.

The debugger uses short one- or two-letter commands, some followed by an optional hexadecimal address. Below is a complete list of the commands.

X—Exit from the debugger.

DR—Display the contents of the 6502 registers:

```
DR
     A=BB X=10 Y=20 P=B0 S=DF
```

A is the accumulator, X and Y are the index registers, P is the processor status register (which includes the carry flag, zero flag, etc.) and S is the stack pointer.

CR—Change the contents of any of the 6502 registers.

```
CR<,1,2,,DE
```

The specified values are stored in the registers in the same order the registers are displayed by the DR command. In the above example, the accumulator is unchanged, the X register receives a 1, the Y register a 2, the status register remains unchanged and the stack pointer is adjusted to DE.

D—Display memory.

"D3000,0" displays memory location $3000. When the second parameter is less than or equal to the first, only one location is shown. "D3000,3010" displays memory from locations $3000 through $3010. Enter D by itself, and the next eight locations (in this case, $3011 through $3018) will be displayed. If only the second parameter is omitted, a default of eight memory locations is displayed:

```
D3000
3000  10 40 20 22 34 11 12 FE
```

Note that the output of the debugger is always in hexadecimal. All input addresses and register values must be specified in hex as well.

C—Change memory.

```
C3034<21,23,,2E
```

The command itself is immediately followed by the starting hexadecimal address to change. The values to be placed in memory, starting at the first location, are separated by commas. Two consecutive commas

IF YOU DO NOT KNOW THE ABSOLUTE ADDRESS OF A NEEDED LABEL, THEN YOU SHOULD GO BACK TO THE SOURCE CODE, MAKE THE CHANGES THERE AND REASSEMBLE.

tell the debugger to skip over that memory location, leaving it unchanged. In the above example, memory location $3034 receives $21, $3035 receives $23 and $3037 gets $2E.

M—Move memory.

```
M0600<0700,0800
```

The above tells the debugger to move memory from locations $0700 through $0800 to memory beginning at location $0600. The destination address ($0600 in this case) must be less than the first source address ($0700) or greater than the ending source address ($0800). If the source and destination areas of memory overlap, you may get unexpected results.

V—Compare two blocks of memory (verify).

You might, for example, use this to compare two slightly different versions of the same program to see where something has changed.

```
V7000<7100,7123
```

The above command tells the debugger to compare memory from $7100 through $7123 to memory at $7000. Any memory locations that do not match are shown side by side:

```
V7000<7100,7123
7101 00 7001 21
```

In the above comparison, all memory from $7100 through $7123 matched memory from $7000 through $7023, except at one location. Memory location $7101 contained a $0, while $7001 contained a $21.

L—List memory with disassembly.

The "L" command is one of Asm/Ed's most powerful. It can be used to disassemble your operating system ROM (beginning at $C000) to see what some of the routines look like. It can be used to disassemble object files loaded into memory to see how they work. Here are some examples:

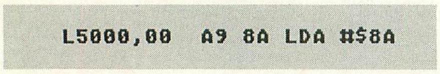
L7000—List memory with disassembly (as many lines that will fit on the screen) beginning at $7000.

L—List memory with disassembly starting at the next location (picking up where the previous L command left off).

L7000,0 or L7000,7000—List and disassemble $7000 only.

L2300,2400—List and disassemble memory from $2300 through $2400.

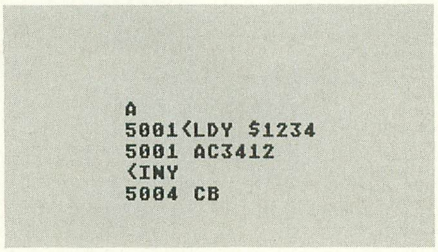
When the debugger comes across data that cannot be disassembled (such as data tables or strings, for example), it will print a series of question marks. Otherwise, the data is shown in hexadecimal, as well as in its equivalent assembly mnemonic form:



```
L5000,00  A9 8A LDA #$8A
```

A—Assemble a single instruction.

This comes in handy when you want to test a small patch to a program. Simply type A and press return to get into the single-line assembly mode. You must first specify an address, followed by a less-than character (<) and the assembly instruction. To assemble to successive memory locations, subsequent entries require only the less-than character followed by the assembly instructions. For example:



```
A
5001<LDY $1234
5001 AC3412
<INY
5004 CB
```

In the above example, we have assembled LDY $1234 and INY into consecutive memory, starting at $5001. Note that here your assembly instructions must use the dollar sign to indicate hexadecimal. Press Return on an empty line to exit the mini-assembler. You cannot refer to labels in the program, since the debugger doesn't keep track of them. If you do not know the absolute address of a needed label, then you should go back to the source code, make the changes there and reassemble.

SOONER OR LATER, YOU'LL GET TIRED OF USR ROUTINES (MOSTLY BECAUSE THEY ARE SO DIFFICULT TO DEBUG).

G—Execute instructions beginning at a particular address.

Type the letter "G" followed by the first execution address. The program will continue to run until the system crashes, you press the break key, or a BRK (break) instruction is executed.

T—TRACE.

Type the letter "T" followed by the address at which to begin execution. The instruction will be executed, immediately followed by a dump of the instruction (list a single line with disassembly) and the CPU registers. This continues until a BRK instruction is executed or you press the break key.

S—Step singly through instructions.

Sometimes you need to test a single instruction at a time. The debugger's step command is used for this task. Enter "S" followed by the address to begin execution. The effects are the same as the TRACE command, except the debugger stops execution after each assembly instruction. Type "S" and Return repeatedly to continue single-stepping through the program.

X—Exit the debugger and return to Asm/Ed's editor.

Error Codes

The error codes between 128 and 255 are the same as those in your Atari BASIC reference manual. These are generally input/output errors associated with CIO (central input/output) utility operations, the heart of your Atari's operating system. There are 19 other error codes that you may encounter while assembling or debugging your programs:

1—The memory available is insufficient for the program to be assembled.

2—For the command "DEL xx,yy," the line number xx cannot be found.

3—There is an error in specifying an address (mini-assembler).

4—The file named cannot be loaded (wrong file format).

5—Undefined label reference (you probably misspelled a label in your program).

6—Error in syntax of a statement (missing operand or misspelled assembly mnemonic).

7—Label defined more than once.

8—Buffer overflow. (I'm not certain what this means.)

9—There is no label or "*" before "=". (An equals sign was found in the first field of a line of code. All equals signs must be preceded by either a valid label or an asterisk.)

10—The value of an expression is greater than 255 where only one byte was required. (e.g., LDA #LABEL, where label is an address of some memory location greater than 255).

11—A null string has been used where invalid.

12—The address or address type specified is incorrect (e.g., LDA (PGZRO),Y would result in this assembly error if the label PGZRO was not an address of a memory location less than 256).

13—Phase error. An inconsistent result has been found from pass 1 to pass 2 (e.g., two bytes were reserved for some label on the first pass, but on the second pass only one byte was needed. This is avoided by minimizing forward references and by defining all known labels at the top of the file before any assembly code. You will get this error a lot as you learn the language).

14—Undefined forward reference (e.g., misspelled label or reference to a label not defined).

15—Line is too large.

16—Assembler does not recognize the source statement.

17—The line number is too large (32767 is maximum).

18—LOMEM command was attempted after other command(s) or instruction(s). LOMEM, if used, must be the first command after entering the Asm/Ed editor.

19—There is no starting address (e.g., you forgot the "*" directive at the top of your program).

IF YOU DIDN'T PAY MUCH ATTENTION TO ANALOG'S "MASTER MEMORY MAP" SERIES, I STRONGLY RECOMMEND THAT YOU GO BACK AND READ IT.

Expressions

The assembler can perform many useful computations for you. The operators recognized and the operations they perform are as follows:

```
+   Addition
-   Subtraction
*   Multiplication
/   Division
&   Logical AND
```

Expressions may not contain parentheses, and they are always evaluated left to right. (There is no precedence placed on operators.) Some examples follow:

```
100 STORAGE = $4000
110  *= STORAGE + $10
200  JMP START+20
300  LDA #STORAGE&$0FF
310  LDX #STORAGE/$100
320  LDA #3*15
```

USR Routines

The USR command of Atari BASIC allows you to call assembly language routines. These routines can perform special functions to vastly improve the performance of BASIC. For example, assembly USR routines may be implemented for Player/Missile graphics movement, sort algorithms or high-speed disk I/O functions.

Assembly code won't normally be loaded as part of your BASIC program. It must be loaded using a routine in BASIC, placing the data values into strings or POKEing it into safe RAM, for example. You may place up to 256 bytes of assembly code into Page 6 (beginning at memory location 1536). If you do not use the cassette (C:), up to 128 bytes of code can go into Page 4 (beginning at memory location 1024), the cassette buffer. If your code is "position independent" (relocatable), it may be loaded into a BASIC string.

What is position-independent assembly code? Such a program may have no JMP or JSR instructions (with the exception of JSR's to ROM addresses that are guaranteed not to move). So how do you implement loops? Use branch instructions. If your code gets much larger than 256 bytes, writing position-independent code can be difficult. The largest routine I've ever written of this type was 410 bytes long. You may also "relocate" your code. This requires a foreknowledge of all the JMP and JSR instructions in your code. You may then load the object code into a string, determine its starting address, and then POKE adjusted address values in for all the JMP and JSR instructions. This is no small task and is seldom used. Generally, your USR routines will be fairly small and can be written in a position-independent manner.

The format of a BASIC USR command is:

```
A=USR(ADR,PARAM1,PARAM2,PARAM3)
```

The first parameter, ADR, is the starting address of the assembly code you wish to execute. The values following are parameters that are passed to the assembly code on the system stack after being converted to integers. The variable A takes on an integer from memory locations $D4 and $D5 (low byte, high byte). This is how you return a value to BASIC.

Let's write a USR routine to add two integers and return the result. Our BASIC program might look like this:

```
10 TRAP 1000
20 OPEN #1,4,0,"D:MYUSR.OBJ":
REM Our USR code in a file
30 TRAP 70
40 FOR I=1 TO 6:GET #1,A:NEXT
 I:REM Ignore 6-byte load hea
der of file
50 I=1536:REM USR routine was
 assembled for Page 6
60 GET #1,A:POKE I,A:I=I+1:GO
TO 60:REM End of file error w
ill terminate our entry of th
e program
70 CLOSE #1
80 PRINT"INPUT NUMBER 1 ";:IN
PUT N1
90 IF N1<0 OR N1>65535 THEN ?
 "OUT OFRANGE":GOTO 80
100 PRINT "INPUT NUMBER 2 ";:
INPUT N2
110 IF N2<0OR N2>65535-N1 THE
N ? "OUT OF RANGE":GOTO 100
120 SUM = USR(1536, N1, N2 )
130 PRINT "NUMBER ";N1;" PLUS
 ";N2;" EQUALS ";SUM
140 END
1000 PRINT "COULD NOT FIND US
R ROUTINE FILE"
1010 PRINT "MYUSR.OBJ"
1020 END
```

Now we need to write an assembly language program with Asm/Ed that implements this USR routine. It will accept parameters N1 and N2 off the stack (two two-byte integers), add them, and return the result to SUM through memory locations $D4 and $D5. Our code might appear as shown in Listing 1.

Enter this program with Asm/Ed and execute the instructions in the first two comment lines. When you get an assembly with no errors, your file D:MYUSR.OBJ should be ready to test with the BASIC program.

Work at this until it performs as expected. As you become more adept at writing USR routines, you may wish to develop utilities for converting .OBJ files into a series of BASIC DATA statements, so you can simply READ and POKE them without using messy file I/O to initialize the USR routine. It takes a relatively long time to install USR routines by poking them into memory or strings, but once in place, they execute amazingly fast.

You will find that USR routines are difficult to debug since you need to initialize and call them from BASIC. If you mess up the stack or some other operation, the computer usually crashes inexplicably. It isn't easy to debug USR routines from DEBUG, because you will have to write sophisticated test routines to stuff all sorts of test values on the stack.

Stand-Alone Assembly

Sooner or later, you'll get tired of USR routines (mostly because they are so difficult to debug). When you do, it is time to take the plunge into writing a stand-alone assembly language program. Then you will get into the complexities of keyboard input, screen output, disk I/O and printer output from the Asm/Ed environment. Complete libraries of routines, such as a "graphics package" that performs the equivalent of BASIC's GRAPHICS, COLOR, PLOT and DRAWTO, will become a necessity. This is where ANALOG's *Boot Camp* series will help the most. In the months to come you will learn everything from keyboard input to floating-point processing, all from the assembly language level.

As an example of a stand-alone assembly language program, and an illustration of its raw speed, we present the following demonstration. First, type this BASIC program and run it. While it executes *(it will* take about 12 minutes), read the remainder of this article to see how the same functions can be performed in assembly language:

```
10 DINDEX=88:REM Screen RAM p
ointer
20 SCREEN=PEEK(DINDEX)+256*PE
EK(DINDEX+1)
30 FOR X=0 TO 255
40 A=X
50 FOR Y=0 TO 255
60 POKE SCREEN+Y,A
70 NEXT Y
80 NEXT X
```

At location DINDEX is a two-byte "pointer." Memory locations 88 and 89 hold the address of the beginning of screen RAM. The equation in line 20 calculates the variable SCREEN, which we use as a direct pointer for the POKE in line 60. In our assembly language equivalent of the above program, this problem is even easier to solve. (This is seldom the case, however; most things are harder to do in assembly language. This demonstration is designed specifically to show the strengths and speed of assembly language.)

Next, two loops are set up. The inner Y loop is used to POKE the current value of X into the first 256 screen RAM locations. You will see these characters fill the top portion of your display. All ATASCII values from zero through 255 are POKEd, with the help of the X loop. The variable A was used simply for a more symmetrical comparison with the assembly code to follow.

Let this BASIC program run to completion. Time it carefully. When you finally get the READY prompt, reboot your computer with Asm/Ed and enter this equivalent assembly language program:

```
0 ;LIST#D:SCREEN.ASM
1 ;ASM,,#D:SCREEN.OBJ
2  *=$3400
3 RUNAD=$2E0
10 DINDEX = 88  ; Screen RAM
pointer
20 ; We don't have to compute
 SCREEN; we use post indexed
addressing
30 START LDX #0 ; Initialize
variables for loops
40  LDY #0
50  STORE TXA    ; Place scre
en character into A register
60 PUTIT STA (DINDEX),Y ; Pla
ce character on screen
70  INY          ; Next scree
n location
80  BNE PUTIT    ; Y register
 "wraps around" to zero after
 255
90  INX
100  BNE STORE   ; NEXT X
110  RTS         ; Return con
trol to DOS
120  *= RUNAD
130 .WORDSTART  ; So we can l
oad and run from DOS
```

Now execute the two commands in the first two comment lines at the top of the listing. If you get no assembly errors, you will have a file SCREEN.OBJ that is ready to load and run. Go to DOS and execute a binary load of the file SCREEN.OBJ. It will run immediately after loading and return control back to DOS after performing all 65,536 "POKES" of characters to screen memory. Did you catch it? You probably didn't if you blinked. This version of the program takes barely a second to run! If you want to watch the show for a while and exit to DOS when a key is pressed, for example, modify your program as follows:

```
15 CH = 764     ; Keyboard buf
fer
101  LDA #255
102  CMP CH     ; keypressed?
103  BEQ START ; Nope, loop
104  STA CH     ; Yes, clear o
ut key buffer and exit to DOS
```

List this version to disk and reassemble it. When loaded from DOS, it will POKE all those ATASCII patterns to the screen continuously until you press a key on the keyboard. To randomize the show, make these changes:

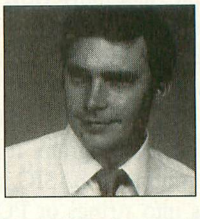
```
16 RANDOM = 53770    ; Always
a random number here
50 STORE
60 PUTIT LDA RANDOM ; Get a r
andom fill character
61  STA (DINDEX),Y  ; Place c
haracter on screen/PX
```

Notice how I always added a meaningful label for each important memory location. Avoid the use of code, such as LDA 53770. The proper use of labels makes it much easier to see exactly what your program does and how it does it.

If you didn't pay much attention to ANALOG's "Master Memory Map" series, I strongly recommend that you go back and read it. Even if you do not understand it all, you will learn a lot. A good memory map is the key to unleashing all the power of your computer. As a 6502 assembly language reference manual, I use *6502 Assembly Language Programming* by Leventhal. This is a general reference for the 6502 microprocessor and does not have any specifics on the Atari computer. It does detail all the 6502 assembly mnemonics and provides examples of multiply, divide and other useful routines.

When you find that Asm/Ed is too slow to suit your tastes as you build larger and more sophisticated programs, consider upgrading to MAC/65 (sold by ICD). This macro assembler supports the use of INCLUDE files, allowing you to easily import "canned" routines that have already been debugged. Its MACRO capabilities allow you to define high-level constructs that vastly simplify the development of assembly programs. With a good MACRO library (such as the *MAC/65 Toolkit* from ICD or *QuickCode* from Stardust Software), your assembly source code will resemble BASIC or some other high-level language while retaining all the power and speed of pure assembly language. MAC/65 is the fastest native 6502 assembler I have ever used, bar none. (*Mad Mac* for the Atari ST will assemble 6502 code at a speed that blows the doors off MAC/65; but that's a whole new ball game.)

Welcome to the fast and complicated world of assembly language programming. I hope this guide will inspire you to put that inexpensive Asm/Ed cartridge to work on all those fantastic ideas that the old faithful Atari BASIC could never handle. ⊓

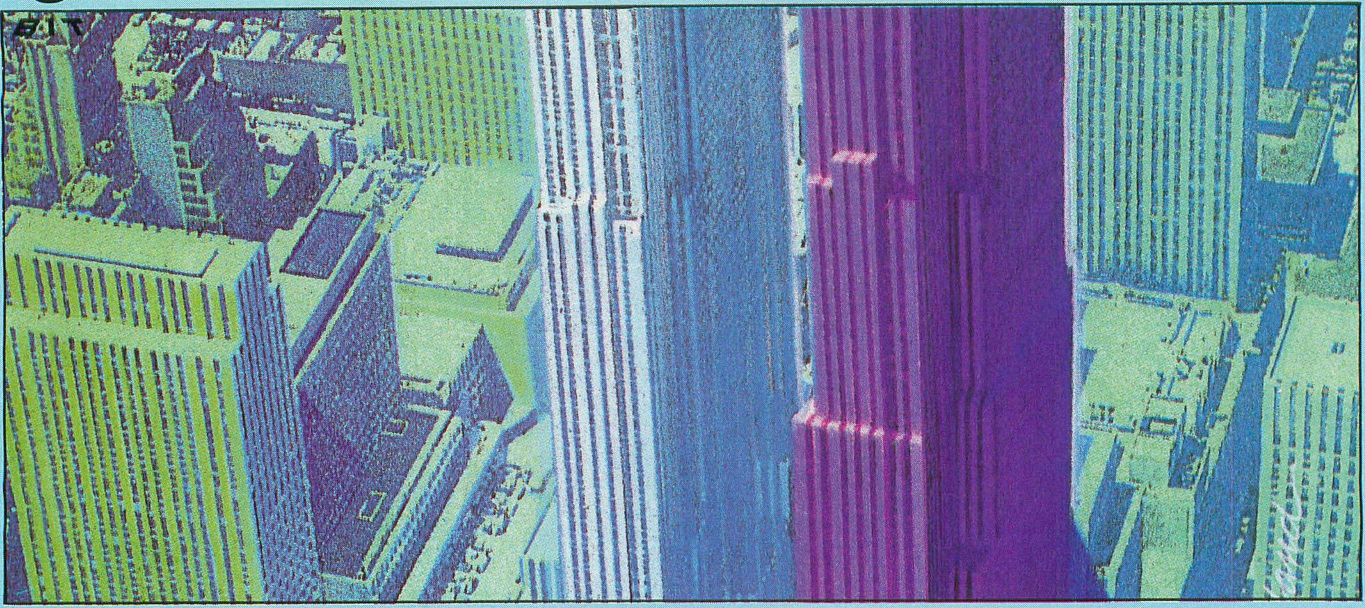


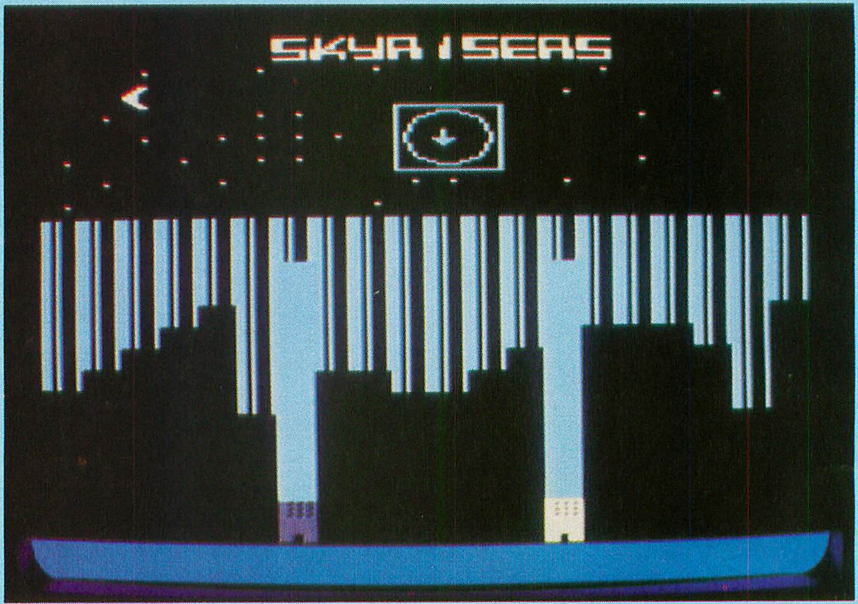
Matthew J.W. Ratcliff is an electrical engineer at McDonnell Aircraft in St. Louis, Missouri. An experienced assembly language, C and Ada programmer on IBM and main frame computers, he still enjoys developing new programs and articles for the 8 bit Atari home computer. He has been an Atari enthusiast since 1982.

SKYRISER

THE OBJECT IS TO BE THE FIRST PERSON TO SUCCESSFULLY CONSTRUCT A SKYSCRAPER TO THE TOP OF THE SCREEN.

BY FRANK MARTONE





Welcome to the mad building game of *Skyriser*, a two-player game that requires two joysticks and fast reflexes. The object is to be the first person to successfully construct a skyscraper to the top of the screen, and although that may not sound too challenging, you and your competitor must watch out for dangerous flying planes and explosions.

At the beginning of the game, each player is given a building, a purple one for player 1 and a white one for player 2. Inside a box located at the center top of the screen, several symbols flash on and off in a random pattern. The actions these symbols represent may be good or bad. The symbols are:

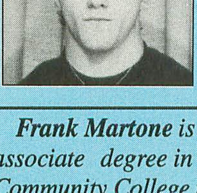
* An up arrow
* A down arrow
* A bomb
* A question mark

To activate a symbol, press the joystick button. If you push your button when the up arrow is displayed, your building will get higher. If you push the button when the arrow is facing down, your building will shrink. If you happen to push the button when the bomb symbol is shown, you will lose your entire building. Pushing your button on a question mark can cause two different actions. If the question mark just flashes, you've avoided any action. But most of the time, it will cause an airplane to zip across the screen. If your building is too high, the plane will hit it, and you'll lose your entire building. There are times when you get lucky, of course, and the plane passes overhead, just missing your building.

The plane can affect only the building of the person who made the mistake of pushing on a question mark. Your opponent's building is not affected.

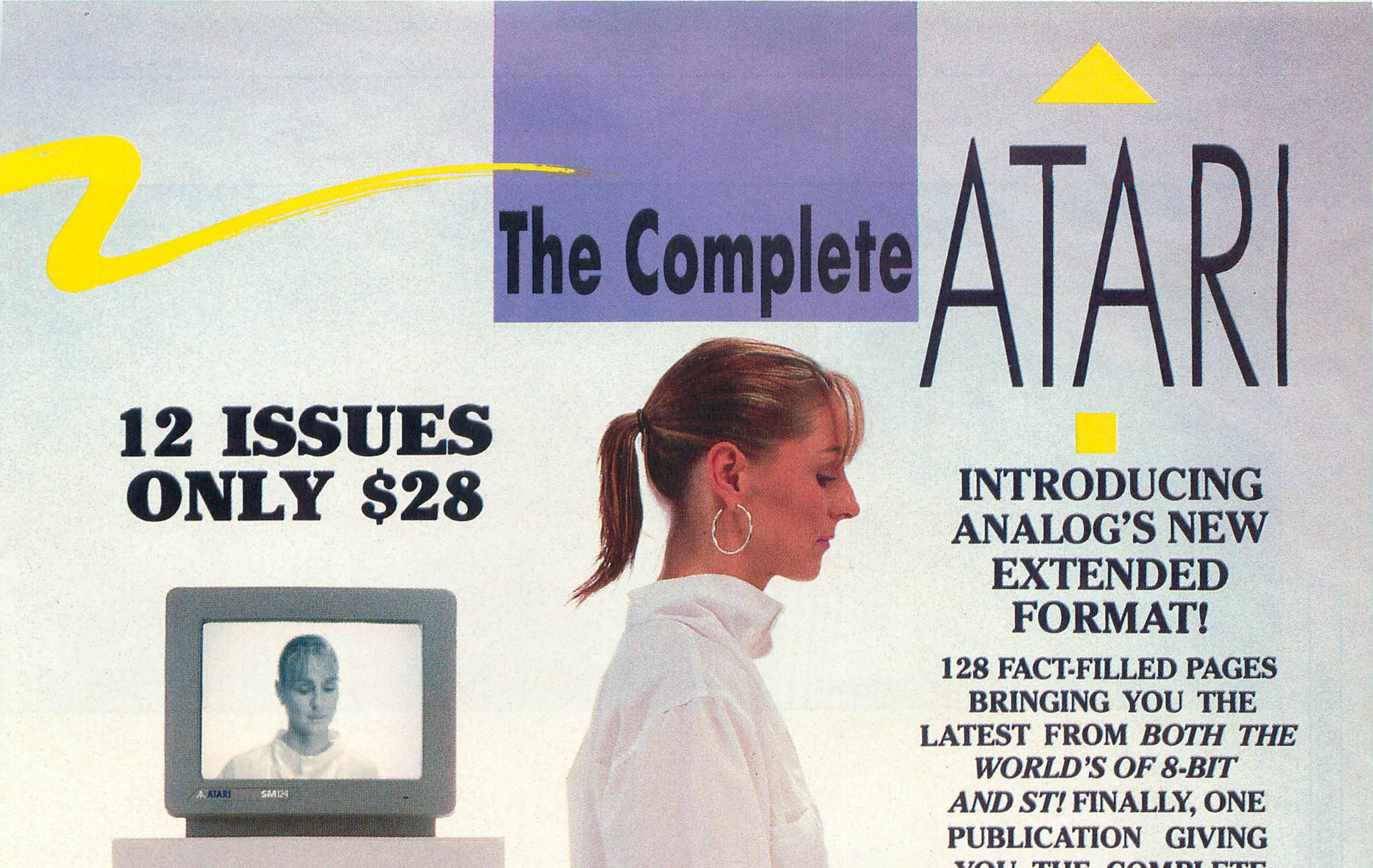
The first building to reach the top of the screen wins. (It is possible to tie.)

Skyriser includes a difficulty feature. The harder levels offer more rapidly changing symbols. There are three levels to choose from: easy, intermediate and hard. Use your joystick to move the flashing cursor to your selection and push the joystick button. I recommend easy for children, intermediate for the average adult and hard for people with a real competitive spirit.



***Frank Martone** is 20 years old and has an associate degree in liberal arts from Suffolk Community College in Selden, New York. He plans to obtain a B.A. in advertising.*

(CONTINUED ON PAGE 20)



The Complete ATARI

12 ISSUES ONLY $28

INTRODUCING ANALOG'S NEW EXTENDED FORMAT!

128 FACT-FILLED PAGES BRINGING YOU THE LATEST FROM *BOTH THE WORLD'S OF 8-BIT AND ST!* FINALLY, ONE PUBLICATION GIVING YOU THE COMPLETE ATARI COVERAGE YOU NEED!

ANALOG COMPUTING

P.O. Box 16927
North Hollywood
CA 91615

□ **12 Issues $28** CDKYY

GET THE NEW ANALOG COMPUTING ON DISK!

□ **12 ISSUES OF ANALOG COMPUTING WITH 8-BIT DISK—ONLY $79!** CDKYY

□ **12 ISSUES OF ANALOG COMPUTING WITH ST DISK—ONLY $79!** CDKYW

NAME ______________________________

ADDRESS ______________________________

CITY ______________ *STATE* _____ *ZIP* __________

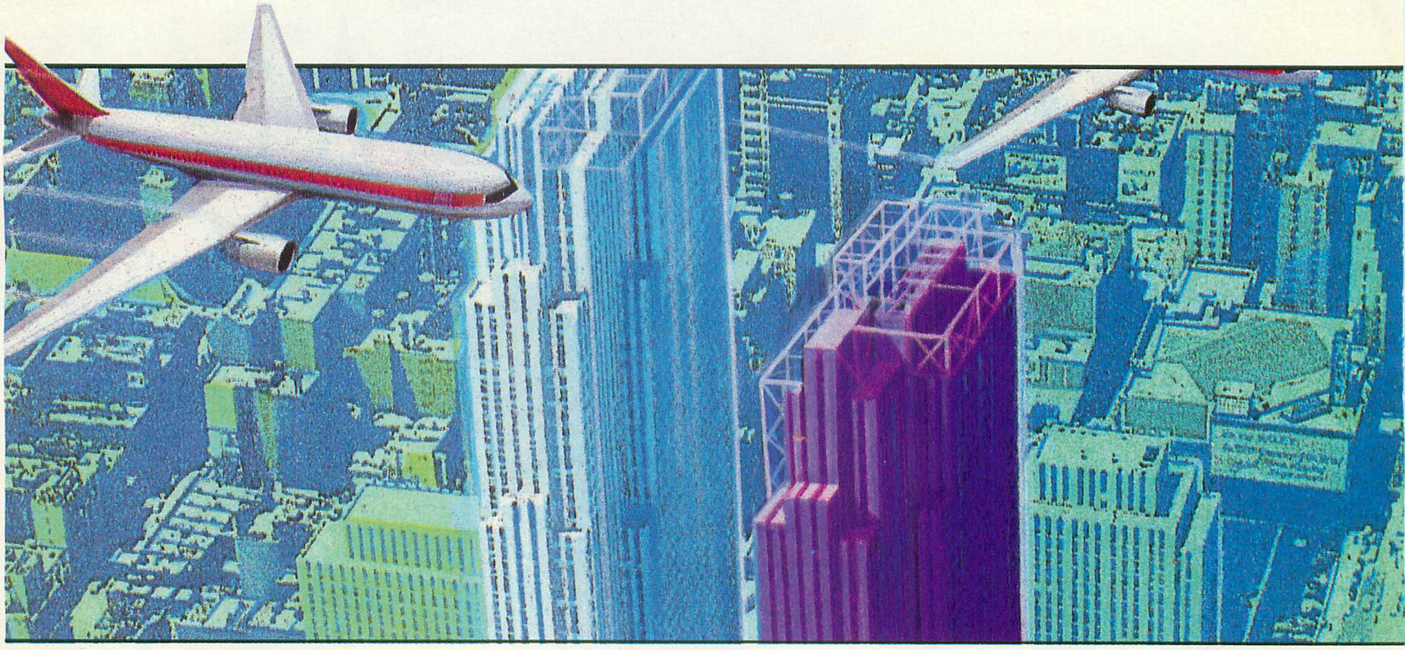
□ *PAYMENT ENCLOSED* □ *BILL ME*

CHARGE MY □ *VISA* □ *MC*

______________________ *EXP* __________

SIGNATURE ______________________________

MONEY BACK ON UNUSED PORTIONS OF SUBSCRIPTIONS IF NOT SATISFIED! FOREIGN—ADD $10. MAKE CHECKS PAYABLE TO LFP, INC. YOUR FIRST ISSUE WILL ARRIVE IN 6 TO 8 WEEKS. WATCH FOR IT! OFFER EXPIRES FEBRUARY 28, 1990



LISTING 1: BASIC

```
AE 10 REM ******************************
PE 20 REM *         SKYRISERS          *
YL 30 REM *            by              *
AW 40 REM *       Frank Martone        *
FW 50 REM *                            *
BS 60 REM *       COPYRIGHT 1989       *
YQ 70 REM *    BY ANALOG COMPUTING     *
AL 80 REM ******************************
BG 90 REM
QF 100 GOTO 490
DJ 110 XX=4:TI=0:P1H=21:P2H=21:P1W=0:P2W=
   0:TIE=0:BOOM=0:POKE 710,101:POKE 711,1
   4
ZF 120 SOUND 1,0,0,0:TI=TI+1:IF TI>TD THE
   N GOSUB 430
XG 130 POSITION 6,P1H:? #6;"Z":POSITION 1
   3,P2H:? #6;"Z":POKE 77,0:SOUND 0,0,0,0
RI 140 IF XX=3 AND STRIG(0)=0 AND STRIG(1
   )=0 THEN BOOM=1:GOSUB 300
BX 150 IF XX=3 AND STRIG(1)=0 THEN GOSUB
   370
QI 160 IF XX=3 AND STRIG(0)=0 THEN GOSUB
   300
ZG 170 IF STRIG(0)=0 AND XX=1 THEN P1H=P1
   H-1:FOR D=15 TO 0 STEP -2:SOUND 0,P1H+
   50,10,D:NEXT D
DG 180 IF STRIG(0)=0 AND XX=2 THEN POSITI
   ON 6,P1H:? #6;" ":P1H=P1H+1:FOR D=15 T
   O 0 STEP -1:SOUND 0,D+90,10,D:NEXT D
EB 190 IF P1H>21 THEN POSITION 6,P1H:? #6
   ;"J":P1H=P1H-1
MS 200 IF P2H>21 THEN POSITION 13,P2H:? #
   6;"F":P2H=P2H-1
EZ 210 IF STRIG(1)=0 AND XX=1 THEN P2H=P2
   H-1:FOR D=15 TO 0 STEP -2:SOUND 0,P2H+
   50,10,D:NEXT D
QJ 220 IF STRIG(1)<>0 OR XX<>2 THEN 230
YD 225 POSITION 13,P2H:? #6;" ":P2H=P2H+1
   :FOR D=15 TO 0 STEP -1:SOUND 0,D+90,10
   ,D:NEXT D:GOSUB 430
LS 230 IF P1H=3 AND P2H=3 THEN TIE=1:GOTO
    1940
LV 240 IF P1H=3 THEN P1W=1:GOTO 1940
NA 250 IF P2H=3 THEN P2W=1:GOTO 1940
EM 260 IF XX=4 AND STRIG(0)=0 THEN GOSUB
   750
IM 270 IF XX=4 AND STRIG(1)=0 THEN GOSUB
   860
MT 280 GOTO 120
IY 290 REM ****PLAYER 1 BUILDING DESTROYE
   D****
UB 300 FOR D=15 TO 0 STEP -1:POKE 708,D+4
   0:SOUND 0,D*444,8,D:SOUND 1,100,0,D:NE
   XT D:SOUND 1,0,0,0
VH 310 FOR R=P1H TO 21:POSITION 6,R:? #6;
   "1":POSITION 6,R:? #6;"6":FOR D=1 TO 5
   :SOUND 0,D,0,D:NEXT D
BS 320 POSITION 6,R:? #6;" ":POKE 708,RND
   (0)*10+50:IF R<8 THEN POSITION 6,R:? #
   6;"!"
HA 330 NEXT R:SOUND 0,0,0,0
MH 340 P1H=21:POKE 708,137:POSITION 6,22:
   ? #6;"J":IF BOOM=1 THEN GOTO 370
IP 350 BOOM=0:RETURN
JN 360 REM ****PLAYER 2 BUILDING DESTROYE
   D****
UP 370 FOR D=15 TO 0 STEP -1:POKE 708,D+4
   0:SOUND 0,D*444,8,D:SOUND 1,100,0,D:NE
   XT D:SOUND 1,0,0,0
UE 380 FOR R=P2H TO 21:POSITION 13,R:? #6
   ;"1":POSITION 13,R:? #6;"6":FOR D=1 TO
    5:SOUND 0,D,8,D:NEXT D
LE 390 POSITION 13,R:? #6;" ":POKE 708,RN
   D(0)*10+50:IF R<8 THEN POSITION 13,R:?
    #6;"!"
GV 400 NEXT R:SOUND 0,0,0,0
VR 410 P2H=21:POKE 708,137
VR 420 POSITION 13,22:? #6;"F":BOOM=0:RET
   URN
QD 430 XX=INT(RND(0)*4)+1:POSITION 10,4:?
    #6;" "
BJ 440 IF XX=1 THEN POSITION 10,4:? #6;"%
   "
DI 450 IF XX=2 THEN POSITION 10,4:? #6;"&
   "
```

(CONTINUED ON PAGE 91)

Increase your publishing power.

Migraph Hand Scanner.

Add scanned graphics to all your projects quickly and easily with the Migraph Hand Scanner.

The Hand Scanner provides a scanning window over 4″ wide with four scanning resolutions: 100, 200, and true 300 and 400 dots per inch. It also offers adjustable contrast, three dither settings for scanning halftones, and a special setting for line art.

Build your own library of images. Logos, photographs, books, illustrations . . . provide you with an unlimited supply of art.

Touch-Up,™ Migraph's high resolution image editor, lets you scan *and* edit your images. Powerful editing tools include crop, invert, resize, flip, cut and paste, multiple zoom levels, a full array of paint tools, and much more.

When your images are "pixel perfect" import them into your favorite ST publishing programs like Calamus, Fleet Street, PageStream, and Publisher ST. In addition, several save formats let you use your images on the PC and Mac.

The Migraph Hand Scanner and Touch-Up. Powerful tools for professional publishing.

See your dealer today for more details or call us direct. Dealer inquiries invited.

For all Atari ST and Mega systems with monochrome or color monitors, 1MB memory and 3.5″ DS disk drive.

MIGRAPH™ INC

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

CIRCLE #105 ON READER SERVICE CARD.

ACCE

I often look down at my Atari and wonder where I would be today if I had had access to a computer at a younger age. What if, instead of faithfully watching *Leave It To Beaver* every day after school, I could have explored the countless adventures contained in a personal computer?

It disturbs me to think of the time I wasted during my younger years sitting on the livingroom floor, hunchbacked and stupefied while sitcom after sitcom blinked in front of my face.

Today I think of my computer as an extension of my personality. Where my computer is located, how it is set up, how it is maintained, the software I prefer and, of course, the seemingly endless amount of stored information all say something about who I am.

There is a computer club in my area that shared some of these ideas with a group of underprivileged children during a computer symposium the group held at a local meeting hall.

Giving up time with their families, not to mention time they could have devoted to themselves, the members of ACCESS (Atari Computer Club East Side Scene) ventured into the rain one early November morning with a mission in mind: Shielding their computers from the rain, they trekked to the Fairview Heights, Ill., city hall to conduct the second annual Big Brothers/Big Sisters Computer Symposium, a program the youth organization's officials applauded.

ACCESS President Tom Guelker told me the purpose of the symposium was to give the children participating in the BB/BS program in his club's area a chance to learn something about the machines that will influence them so much in the future.

Interviewing him in the basement of his home in Collinsville, Ill., I started to identify with what he thought his Atari computer could do for an underprivileged child.

"I want to steer these kids toward the good way of life, get them on the right track. A computer can get their brains working a little bit as opposed to a television set that they'll just sit and stare at all day."

I asked Guelker what made him get involved with the BB/BS organization and found that he had a genuine interest in the development of our youngsters, especially those who are disadvantaged.

"I've always had a soft spot in my heart for disadvantaged children," Guelker said. "They really have a challenge ahead of them. It makes it even worse when they're missing a parent, as with these kids. With the financial difficulties experienced by many single parents, many of these kids won't have the chance to actually own a computer. And if they don't have some exposure to computers by the time they get to high school, they will be at more of a disadvantage."

Expanding on the use of computers in today's high schools was John Sullivan, an 18-year-old member of ACCESS. In telling me about himself, he offered some interesting thoughts on the importance of computer knowledge in high school.

"I use my 800 a lot for writing reports," Sullivan said initially. He said having his own computer gave him the extra time at home to complete his classwork, much of which involved computers.

"It's important to know how to use them because you can do so many things on them if you put your mind to it. It's just mathematics."

Sullivan, who plans to attend pilot training school after graduation, said the kids who are not computer literate have a rough time in high school.

"When I took computer math, which is just basic programming, there were a lot of kids who took it because they needed a computer credit and they really had a rough time. The same went for an English class I took that required basic computer skills; the same kids were having problems."

Sullivan said his school uses computers in accounting, drafting, math, English and, of course, data processing.

It's obvious basic computer skills are something every high school student can benefit from. But according to Barbara Cempura, executive director of the Southern Illinois Big Brothers and Big Sisters organization, her members, who range in age from seven to 14, can benefit from computer training right now.

"It's a very positive experience when a young person can sit down to learn a new kind of skill and is able to achieve some success with it. I think that builds self-confidence and gives the youngster a certain sense of control. Being able to do something, and seeing that there is a positive reaction from someone, gives them the encouragement to continue," Cempura said. "This was a very special type of activity for the kids. Generally, our activities include baseball games, art museums, etc. We do try to do things that are educational as well as recreational, but they are always designed for them to have fun."

Cempura told me that this particular outing, and others like it, is geared specifically for the children without Big Brother or Big Sister companions. She explained that there are many children who fall into this category.

"Right now we have about 140 kids in our area without big brothers or sisters," Cempura said. "I really appreciate the fact that [ACCESS] came to us first and asked if they could do something like this. I was really impressed that they brought all their computers to the meeting area, set them up and then took the time to do some training with the youngsters. The time, the giving of themselves and the sharing of their talents was greatly appreciated."

Like Cempura said, many of the kids involved in the Big Brothers/Big Sisters program, although at an obvious disadvantage with only one parent, at least have the advantage of sharing a few hours a week with their Big Brother or Sister; however, the children in attendance at the seminar were even more unfortunate because they had yet to be matched with a companion.

Research has shown that although not all children from single-parent homes need a big brother or sister, they are more likely to have problems with their self image and confidence. They tend to experience trouble in school, loneliness and low aspirations for the future. The interest and care shown by companionship have been proven to raise self-esteem and confidence in the child's earlier years.

What can a computer do for an underprivileged child? I'm neither a computer wizard nor a psychologist, but it seems to me, after listening to the children and the members of ACCESS describe the symposium, the

SS TO COMPUTERS

BY TOM ARTERBURN

computers came to reflect the children's own personality. They made the computers do what they wanted them to do. Whether it be acting out aggression in a harmless yet mayhem-ridden video program or expressing their feelings with the help of a speech synthesizer program, the children, whether they knew it or not, exhibited self-expression, one of the most important aspects of child development.

According to Paula Cobb, coordinator for the Best St. Louis BB/BS office, who accompanied the children to the symposium, it was very helpful to them.

"I thought they loved it. They were very enthusiastic. The one thing that impressed me was their attention span. The children were very interested in the computers and kept their attention on them most of the time we were there," Cobb said.

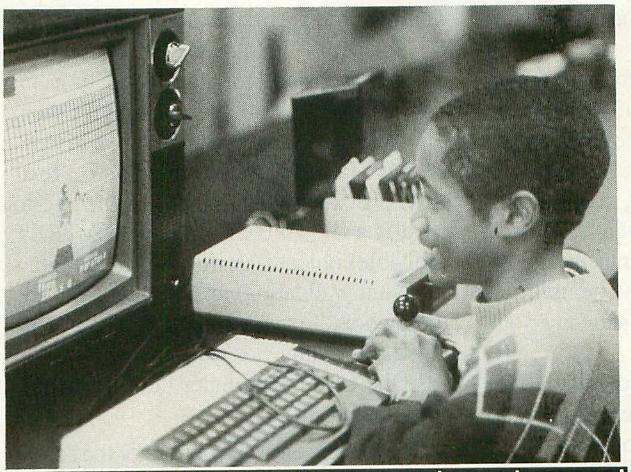
Admitting that she didn't know a great deal about computers, Cobb said she was certain the hand-eye coordination skills necessary in playing some of the computer games were helpful to the children.

"One game the kids loved that really seemed challenging was the karate (*Karateka*) program. I remember the guys in particular liked this game. For a while it seemed to be the only thing being played," Cobb said. "There were also some educational programs the kids worked with, which seemed beneficial. The word games and some of the artwork and graphics they worked with really seemed to help them express themselves."

Keeping in mind the symposium was being conducted by a computer users' group, not trained child-development instructors, Cobb's next remarks were complimentary.

"I thought the ACCESS group showed great patience and kindness in demonstrating to each of the children how to operate the computers. I think the kids will remember that."

When asked about his knowledge of child development and his objectives in organizing the seminar, Tom Guelker, the automobile mechanic from Collinsville, Ill., and president of ACCESS, came clean and described some of his hidden motives for holding the training session.



"Kendrick" clenches a joystick insistently as the second annual Big Brother/Big Sister Computer Symposium draws to an end.

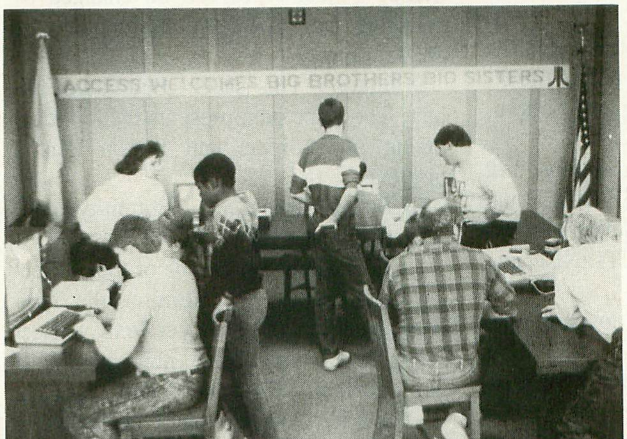


PHOTO BY TOM GUELKER

Atari Computer East Side Scene (A.C.C.E.S.S.) paves the way for underprivileged children to learn more about Atari computers

"Since I became president of ACCESS in 1986, I've always thought community involvement and community relations something I wanted our group to get involved in," Guelker said. "It allows people to learn more about our group and computer enthusiasts in gener-

(CONTINUED ON PAGE 113)

Using Dialog Boxes with GFA BASIC 3.0, Part 2

by David Plotkin

The GFA RCS

GFA BASIC 3.0 comes with an RCS, but unless I missed something, it doesn't come with any instructions on how to use it. Most of it is fairly intuitive, but I want to cover the most important aspects of its use so that you will understand how I built the sample dialog box.

Figure 1 shows the RCS. You will notice two main areas: the icons on the left side of the working window and the icons across the bottom. These have been numbered. I will discuss each one briefly, starting with the icons on the left side, which are used to modify the objects' appearances and attributes. To select one of these items, click on it and a box will open, showing your options:

1. Modifies the background color. If the color is set to white, no pattern will show, regardless of what is chosen in 2.

2. Modifies the background pattern.

3. Modifies the text color.

4. Modifies the text size (two sizes are available) and text writing mode (replace, transparent, etc.).

5. Sets the alignment for text (within a box-type object) and objects within the dialog box (centered, left aligned, bottom, etc.).

6. Sets flags and states. As mentioned last month, the flags are EXIT, EDITABLE and SELECTABLE. The states select such attributes as SHADOWED, OUTLINED and CHECKED.

7. Sets the border color for box-type objects.

8. Sets the thickness of borders for box-type objects.

The Dialog Box Objects

Across the bottom of the screen are the dialog box objects (numbered nine through 19). To move an object into the dialog box, click and drag the appropriate item from the bottom of the screen into its position in the dialog box. To modify the object with the icons at the left side of the screen, you must click on the object to select it, then select the icon you want to use to modify the object. Objects can have their size modified by clicking on the object to select it, then clicking and dragging to the new size the dark box in the lower-left corner of the object. An object can be moved within the dialog box by clicking on it to select it, then dragging the object to its new position.

Objects with strings or single characters in them can have their text modified by double-clicking on them to open a dialog box for input. Editable text fields can have all their fields set (template, validation and initial string) by double-clicking on them to open a dialog box. One bug I've discovered is that if you've modified a G_Text- or G_Boxtext-type object and then go to modify a G_FTEXT or G_FBOXTEXT (editable) object, the first two fields (template and validation) are missing from the dialog box. To work around this, you must close the dialog box you are working on (use the Close item under the File menu), then reopen it.

The 11 objects at the bottom of the screen are somewhat bewildering, but many of the objects are identical except for the default setting of the Type item under the Options menu. Object types can be changed by selecting the Type item; the choices are shown below:

9. Button— This is a G_Button-type object. The Type item presents two options: String (no border) and Button (has a border).

10. String—This is a G_Button-type object, identical to 9.

11. Edit—This can be one of four different object types, depending on the Type item selection. If you select Text, you get a G_TEXT-type object (noneditable). If you select Boxtext, you get a G_BOXTEXT-type object (noneditable with a border). If you select Ftext, you get a G_FTEXT (editable) object, and if you select Fboxtext, you get a G_FBOXTEXT-type object (editable with a border).

12. Identical to 11.

13. This object can take on one of three object types, depending on the Type item selection. If you select I-Box, you get an object of type G_IBOX (box with invisible interior). If you select Box, you get an object of type G_BOX. Finally, if you select Boxchar, you get an object with type G_BOXCHAR (box with a single character inside).

14. This object is identical to 13.

15. Text—This object is identical to 11.

16. This object is identical to 13.

17. Boxtext—This object is identical to 11.

18. This object is an icon. No icon editor is included with the RCS; however, you can import icons from other sources by using the Load item under the Options menu.

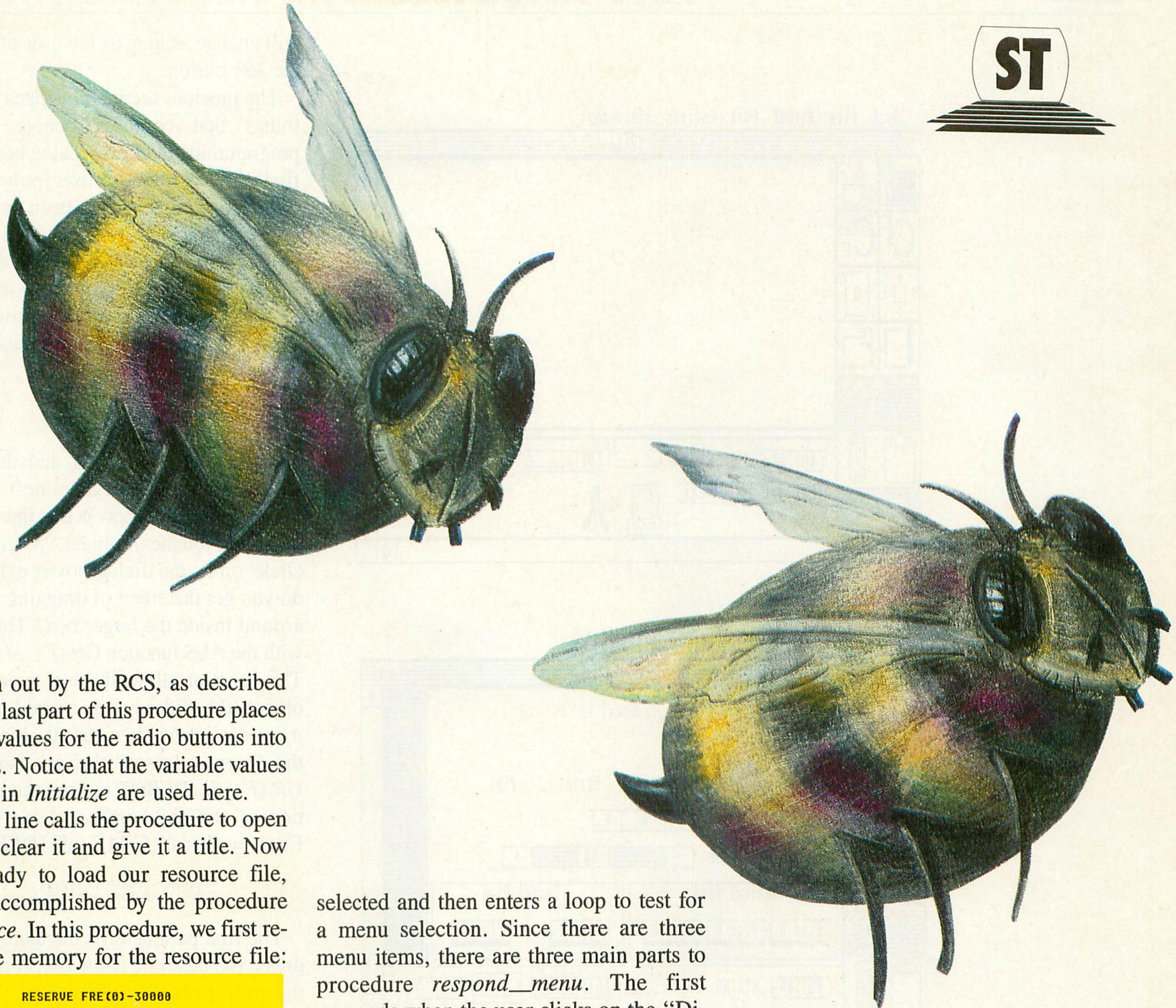
19. This object is a bit image. No bit-image editor is included with the RCS; however, you can import bit images from other sources by using the Load item under the Options menu.

You should notice that one of the things you can set for each object is the object's name. While this is optional, you should do it for an important reason. Under the Global menu there is an item called Output. Clicking on this item brings up a dialog box where you can choose to have the RCS write out a GFA .LST file, which can be merged into your program. This file equates variables to the numbers assigned to each object by the RCS. It is obviously much easier to refer to the object in your program by name, rather than by number. However, the .LST file uses the names you gave the objects as the variable name, so you should name each object (and don't use spaces in the names). To see an example of this, check out the procedure *Initialize* in the sample listing. The comments were modified to note what each field is used for (the RCS doesn't provide very informative comments as part of the .LST file).

Using the Dialog Box

Now that we've gotten all that out of the way, it's time to see exactly how to retrieve information from editable fields, exit buttons, radio buttons, up/down buttons and sliders. We'll look at each section of the accompanying listing.

The first line of the program dimensions two arrays for the two sets of radio buttons. As we'll see later, it is much simpler to use radio buttons if the object values of the radio buttons are contained in arrays. The next line calls the procedure that sets up the simple menu at the top of the screen and is followed by a line that calls the procedure to initialize the variables. The bulk of the procedure *Initialize* is the modified listing



file written out by the RCS, as described above. The last part of this procedure places the object values for the radio buttons into their arrays. Notice that the variable values set earlier in *Initialize* are used here.

The next line calls the procedure to open a window, clear it and give it a title. Now we are ready to load our resource file, which is accomplished by the procedure *ld_resource*. In this procedure, we first reserve some memory for the resource file:

```
RESERVE FRE(0)-30000
```

Then we use *RSRC_LOAD* to attempt to load the resource file called BENEFITS.RSC. If we are successful, we proceed, but if not (the value returned is zero), we give back the memory and end the program:

```
IF RSRC_LOAD("BENEFITS.RSC")=0
  ALERT 3,"Resource file not found",1," STOP ",a%
  RESERVE FRE(0)+30000
  EDIT
ENDIF
```

The next task is to determine the address of the resource and place that address into the variable *adr%*:

```
~RSRC_GADDR(0,0,adr%)
```

Finally, we need to find out the width of the box that contains the "importance" slider, less the width of the slider itself. This number will be used in calculating the new position of the slider when it is dragged by the user:

```
width%=OB_W(adr%,imparent&)-
OB_W(adr%,IMPSLIDE&)
```

Returning to the first portion of the program, the last lines of the main program simply tell GFA to jump to the procedure *respond_menu* when a menu item is selected and then enters a loop to test for a menu selection. Since there are three menu items, there are three main parts to procedure *respond_menu*. The first responds when the user clicks on the "Dialog Tutorial" item under the Desk menu and simply puts up an alert box. The last one responds when the user clicks on "Quit" by jumping to procedure *pgm_end*. This procedure gets rid of the resource, recovers memory, closes the window and returns to the editor.

It is the second menu item, "Add Info," which puts the dialog box into motion. In *show_resource*, the centered coordinates of the dialog itself are calculated by a call to *FORM_CENTER*. The GET command is used to store a copy of what is on the screen where the dialog box will be drawn. This will be used later to restore the screen after we are done with the dialog box. The next five lines use the CHAR command to place an empty string in each editable field. You could just as easily place a non-empty string in each field to initialize it, but again, be careful not to try to put in a string that is too long. Remember that since these are editable fields, you must use the double braces with CHAR.

The next two lines set the number of dependents, *depnum%*, equal to zero and then place that number in the boxed text field designed for that purpose. The number *depnum%* is first converted to a string with *STR$*, then placed in the text field using CHAR with only a single set of braces. Next we stuff the number 50 into the slider and set its starting position. The exact steps used here will be discussed in more detail in the section on sliders. The *FORM_DIAL* command with a parameter of one draws an expanding box on the screen, and *OBJC_DRAW* puts the dialog box on the screen.

The Repeat/Until loop drives the whole process. The first line of the loop uses *FORM_DO* to tell AES to handle the dialog box. The *FORM_DO* command returns the number of the exit object on which the user clicked, and depending on that value, we want to take different actions.

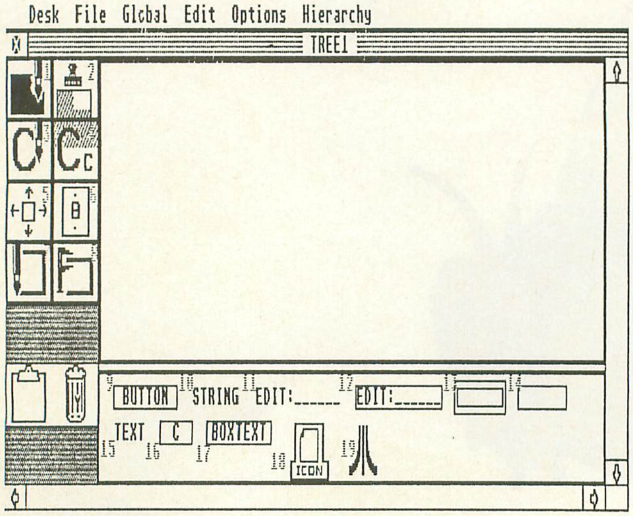


FIGURE 1

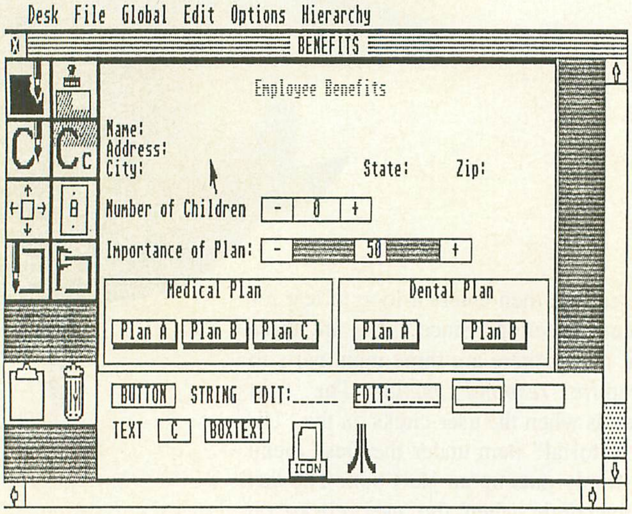


FIGURE 2

The Up and Down Buttons

When we click on the "+" or "−" buttons for the number of dependents, we want the number displayed in the center box to change. The dialog box can't handle this on its own; we must exit the dialog box, adjust the number and reenter the dialog box. For this reason, these two buttons are "touchexit" buttons. As discussed previously, when you click on a touchexit object, the dialog box is exited immediately, without waiting for the button to be released.

In our sample program, the dialog box is entered (*FORM__DO*), and if the "+" button is selected, the *depnum%* variable is incremented. A test is made to ensure that the variable is within legal limits, and it is adjusted if it isn't. The new value of *depnum%* is placed back into the box:

```
CHAR(OB_SPEC(adr%,depnbrsl&))
=STR$(depnum%)
```

And the box is redrawn to show the new number:

```
~OBJC_DRAW(adr%,depnbrsl&,
6,x%,y%,w%,h%)
```

Note that although we exited the dialog box as soon as the touchexit button was clicked, the dialog box was still visible on the screen because we never erased it. Once we are done processing the button, we reenter the dialog box by reexecuting the *FORM__DO* command within the repeat loop. One of the side effects of using a touchexit button here is that the numbers will change as long as the user holds down the left button.

The previous section illustrates one of the things that confuses people who are programming their own dialog boxes for the first time. From the user point of view (which is what they have been up till now), everything appears to be handled by the dialog box, when in fact, a great deal is being handled by the program itself.

The next eight lines of the program handle the case where the user selected the "-" button.

Sliders

Sliders are interesting and useful constructs, so we'll cover them next. The first thing you need to know is that the slider box itself is a touchexit object. When the user clicks on it, the dialog box is exited. How do you get the effect of dragging the slider around inside the larger box? This is done with the AES function *GRAF__SLIDEBOX*. This function allows the user to drag a child object (the slider box is a child of the box it resides in) inside a parent object. When the user releases the mouse button, *GRAF__SLIDEBOX* returns the new position of the child object. Let's illustrate. First, we'll call *GRAF__SLIDEBOX*:

```
sliderpos%=GRAF_SLIDEBOX
(adr%,imparent&,impslide&,0)
```

The first parameter is the dialog box address; the next one is the object number of the parent object, followed by the object number of the child object and then a flag. If the flag is zero, the user can drag the child around inside the parent on a horizontal axis (left and right). If the flag is one, then the user can drag the child around inside the parent on a vertical axis (up and down). The function returns the position of the child object (slider) inside the parent object and is always a number from 0 (left or up) to 1,000 (right or down).

The next step is to convert the returned number (0-1,000) to the new X coordinate of the slider:

```
newpos%=width%/1000*sliderpos%
```

Thus, we get a ratio of the actual width of the parent object, less the width of the child object (defined as *width%*) to the maximum range:

```
width%/1000
```

This number is then multiplied by the actual slider position (*sliderpos%*). For example, if the width is 200 and the returned position was 500 (halfway to 1,000), then the new position of the slider relative to the parent object would be 100 or exactly halfway across, just as you'd expect.

Then, we use *OB_X* to set the new coordinates of the slider:

```
OB_X(adr%,impslide&)=newpos%
```

Now we need to convert the new slider position, as given by *sliderpos%* (which can vary from 0-1,000), to a number in our range (0-100). This is simple:

```
import%=sliderpos%/10
```

We then put this new number back into the slider:

```
CHAR(OB_SPEC(adr%,impslide&))
=STR$(import%)
```

The "+" and "−" buttons associated with the "importance" slider work somewhat similarly to the slider, except that *GRAF_SLIDEBOX* is not required. These buttons are again touchexit, as were the buttons for the number of dependents. Once the user clicks on one of these buttons, the program takes over. In the case of clicking on "−", a certain amount is subtracted from the slider position:

```
SUB sliderpos%,10
```

The number subtracted should be no less than the total range (1,000) divided by your total range (100 in this example). If the number is smaller than this ratio (ten, in case you can't find your calculator) then the slider number won't change every time you click on the "+" or "-" buttons. This is because, as noted above, we divide the new slider position by ten before we put the number back into the slider. We again test to make sure that *sliderpos%* hasn't moved outside the legal range, correct it if it has, then calculate the new slider position:

```
newpos%=width%/1000*sliderpos%
OB_X(adr%,impslide&)=newpos%
```

Does this look familiar? It should; it's the same equation used for the slider. Finally, we use exactly the same steps as with the slider to position the slider, calculate the new value and place it in the slider box. It is then redrawn to show the new value with *OBJC_DRAW.*

The next 13 lines of the listing handle the case of the user clicking on the "+" button.

It's Okay to Quit

When the user clicks on the OK button, the procedure *hndl_ok* takes over. The first line calls *FORM_DIAL* again, this time with the first parameter of two, to create the shrinking rectangle. The screen is then restored with PUT. The next order of business is to return the OK button back to the nonselected state:

```
~OBJC_CHANGE(adr%,exit_obj%,
0,x%,y%,w%,h%,0,0)
```

Note that we are not bothering to redraw the button in its new state (last parameter is zero) because the dialog box is no longer on the screen. We now recover the data that was entered into the edit fields using statements such as:

```
nm$=CHAR((OB_SPEC(adr%,name&)))
```

Remember that since we are using editable fields, we must use the form of the *CHAR* command with two sets of braces. Once we have recovered the information into strings, we print them on the screen.

Don't Touch That Dial!

Now we are ready to handle the two sets of radio buttons. We'll need a variable to hold which radio button was selected, so we zero that variable out first. We'll do the three "medical" radio buttons first, so we need to step through them and test each one to see if it was selected:

```
FOR cnt&=1 to 3
```

Remember that in *Initialize,* we placed the values of the radio button objects into the array *medrbtn&.* Thus, *medrbtn&(1)* contains the value of the "Plan A" radio button (which happens to be 9). In order to find the state of the "Plan A" radio button, we need to use the *OB_STATE* function:

```
OB_STATE(adr%,medrbtn&(cnt&))
```

Here *medrbtn&(cnt&)* contains the successive values of the medical radio buttons as the *FOR* loop is executed successive times. To find out if the selected state of a radio button has been set (and thus that radio button is the one selected; remember, only one radio button in a set can be selected at a time), we use the function *BTST,* which checks to see if a particular bit is set. In our case, we want to see if bit 0 is set (because bit 0 is the selected state, as noted last month in the section on states):

```
IF BTST(OB_STATE(adr%,medrbtn&(ctn&)),0)
   select&=medrbtn&(ctn&)
ENDIF
```

Next, we want to see which radio button was selected and handle the results accordingly:

```
IF select&<>0 ! a radio button was selected
  'so retrieve the text of the radio button
  med$=CHAR(OB_SPEC((adr%,select&))
  PRINT "Medical plan selected: ";med$
  ' change the radio button back to nonselected
  ~OBJC_CHANGE(adr%,select&,0,x%,y%,w%,h%,32,0)
ELSE
  PRINT "No medical plan was selected"
ENDIF
```

This section of code should look familiar, except for one thing. Note how the second-to-last parameter in the *OBJC_CHANGE* function is 32. Previously, we had used zero. We are changing the object state to 32 because the radio buttons in this dialog box are shadowed, which is state= 32. Thus, 32 is what we must change them back to in order to turn off the selected state. If we had changed them back to 0, we would lose the shadowing.

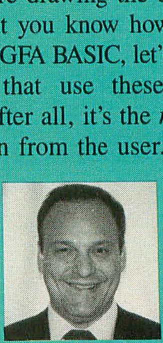
The next 16 lines of code in the sample listing perform the same task for the dental plan buttons, except that there are only two of them. Finally, the last four lines of the program recover the number of dependents from the box and the importance of the plans from the slider and print them on the screen.

Some Closing Thoughts

The dialog box used in this program was quite complex in order to illustrate the many things you can do with a dialog box. Most of your dialog boxes will be simpler, since you won't need everything (editable fields, text, buttons, radio button, sliders and up/down buttons) in one box. And, although this article has been long, the actual explanation of how the dialog box works was quite short.

Finally, you will notice that the sample dialog box doesn't actually do anything useful. This was deliberate. The program is already long enough without adding the necessary code to make it useful too. Obvious extensions would be to store the numbers and editable text fields in an array, so that the data would not simply be thrown away when you clicked on OK. Additional buttons such as "Next" or "Previous" could let you move through the database, recovering the contents of each field from the arrays and placing them into the dialog box fields before drawing the dialog box.

Now that you know how to use dialog boxes with GFA BASIC, let's see some GFA programs that use these handy GEM devices. After all, it's the *right* way to get information from the user.



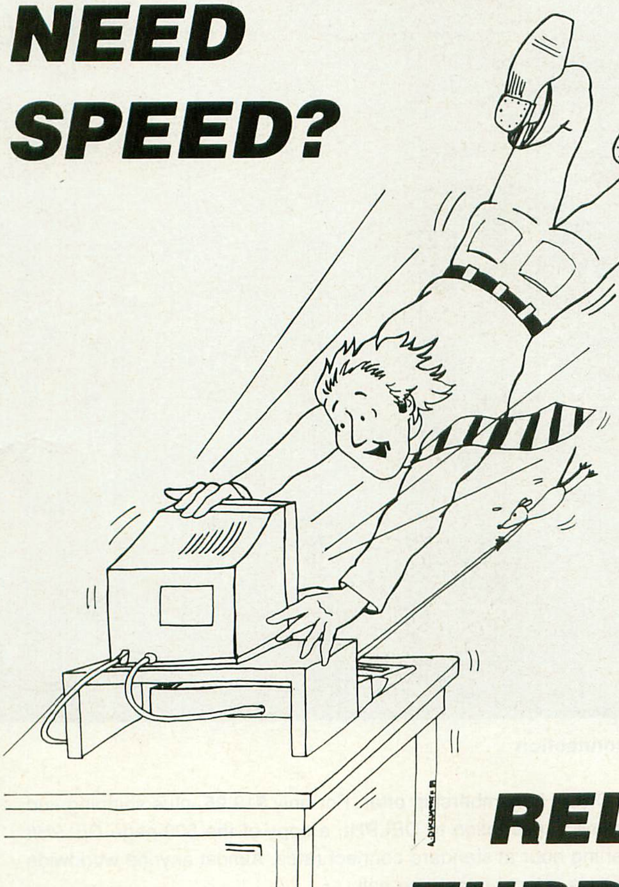
David Plotkin is a chemical engineer currently working as a data analyst for the Human Resources department of the Chevron Corporation. He has had an Atari since 1980.

ATTENTION PROGRAMMERS

**SEND YOUR PROGRAMS
AND ARTICLES TO:
ANALOG COMPUTING**

**P.O. BOX 1413-M.O.,
MANCHESTER, CT 06040-1413**

NEED SPEED?



RELY ON TURBO ST!

Turbo ST turns your Atari ST into a speed demon. Check out **Turbo ST** at your local dealer or download our free demo from your favorite bulletin board.

Upgrades to version 1.8 are only $5 with your original disk. For more information, call (407) 657-4611, or write to SofTrek, P.O. Box 5257, Winter Park, FL 32793.

CIRCLE #106 ON READER SERVICE CARD.

MOVING?

DON'T MISS A SINGLE ISSUE

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

DO YOU HAVE A QUESTION ABOUT YOUR SUBSCRIPTION?

Check the appropriate boxes below:

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

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

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

□ Payment enclosed. □ Bill me.

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

Name ____________

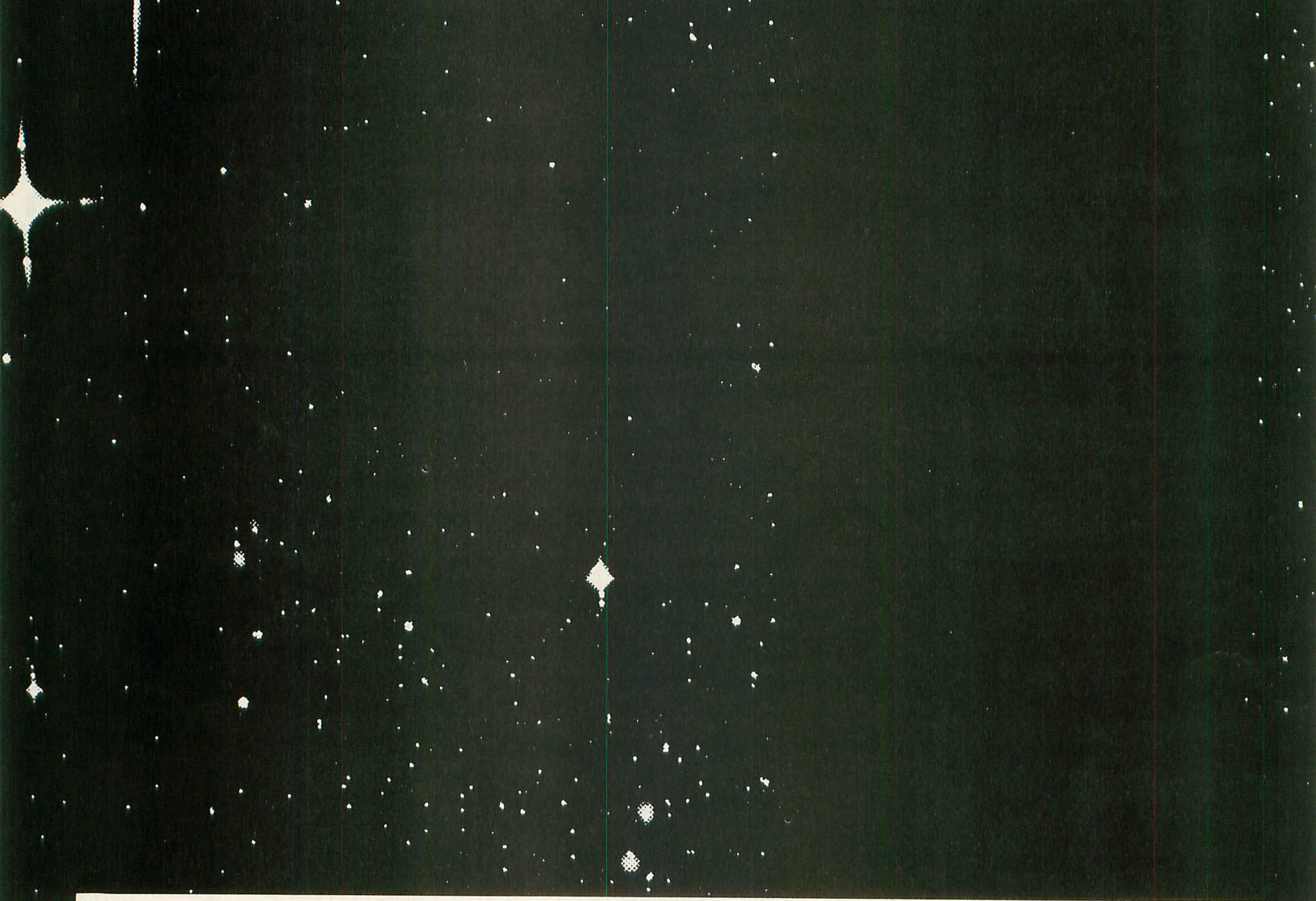
Street Address ____________

City ________ State ______ Zip ______

ATTACH LABEL HERE

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

Database



Make the DELPHI connection

As a reader of ST-LOG, you are entitled to take advantage of a special DELPHI membership offer. For only $19.95, plus shipping and handling ($30 off the standard membership price!), you will receive a lifetime supscription to DELPHI, a copy of the 500-page *DELPHI: The Official Guide* by Michael A. Banks, and a credit equal to one free evening hour at standard connect rates. Almost anyone worldwide can access DELPHI (using Tymnet, Telenet or other networking services) via a local telephone call.

To Join DELPHI

1. Dial 617-576-0862 with any terminal or PC and modem (at 2400 bps, dial 576-2981).
2. At the Username prompt, type JOINDELPHI.
3. At the Password prompt, enter STLOG.

For more information, call DELPHI Member Services at 1-800-544-4005, or at 617-491-3393 from within Massachusetts or from outside the U.S.

DELPHI is a service of General Videotex Corporation of Cambridge, Massachusetts.

DELPHI

Speaking the Lingo: A Newcomer's Guide to "Conference Code"

by Michael Banks

If you've participated in a real-time conference in the Atari ST SIG or Atari Users' Group (or anywhere else on DELPHI), you may well have found yourself a bit confused by some odd words scrolling across your screen. "Btw," for instance, or "ga." While they may look like typographical errors, they really do have meaning in the special context of conference.

Hopefully, you didn't give up and leave because you didn't understand these cryptic words; but if you did, take heart—they're not that difficult to comprehend. In fact, most are merely acronyms ("btw" stands for "by the way," and "ga" stands for "go ahead"). The odd words (and symbols) have evolved to accommodate faster communication. (Some have been adapted from old telegraph or teletype communications conventions; "ga" is one.)

Here's a glossary of the "conference codes" you're likely to encounter:

BRB—"Be right back." Used when someone leaves his or her computer for a moment or enters Mail from Conference, so that others in the conference will not expect responses from that person. Also used when one leaves a conference temporarily.

BTW—"By the way." Used as you would use it in conversation.

GA—"Go ahead." This is usually used after someone has typed several lines that continue a thought (using "...") to indicate that he or she has finished the thought. Also used by one or more conference participants when several people type at once, to defer to another person's comments.

IMHO (or *IMO*)—"In my humble opinion." (Watch out. Someone's dragging out a soapbox when IMHO prefaces a sentence!)

LOL—"Lots of laughter" or "Laughing out loud."

OTF—"On the floor." The ultimate indicator of laughter.

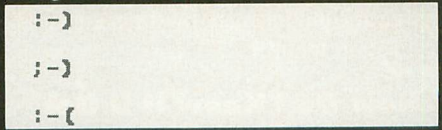
. . . (three periods)—Typed at the end of a line to indicate that the speaker will continue his or her thought with the next line.

<GRIN>—Just what it says, but note that it is enclosed by greater-than/less-than symbols as sort of a cue that you should visualize what the symbols enclose.

<g>—Short for <GRIN>.

<SMILE>—(Same comments as for <GRIN>.)

In addition to the abbreviations, you'll also see some cryptic "symbols" that aren't acronyms. These are "ASCII graphics," used to represent facial expressions and thoughts. Here are the basic ones:



```
:-)

;-)

:-(
```

The first symbol represents a smile (turn the magazine one quarter-turn clockwise if you don't see it). The second is a smile with a wink. The third is a frown or a sad or disapproving face.

Finally, people in a conference with two or more others will often use two dashes and a greater-than symbol (-->), or two greater-than symbols (> >) followed by a name to indicate that the comment that follows is directed to a specific person. For example, if I were online and wanted to direct a comment to someone named RESNICK, I would type:

--> RESNICK

followed by my comment.

And now you're an expert on the verbal shorthand used to make online real-time communication easier. Be aware, however, that new codes are popping up every month. Some catch on, and some don't—but don't be afraid to ask if you don't understand.

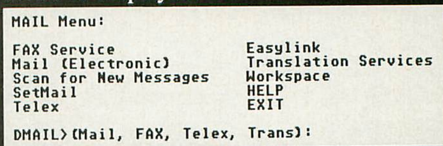
Just the FAX

There, I did it! I promised myself that I'd never use that cliché, but what the heck. (For those of you under age 30 or who don't watch reruns of old TV shows, "Just the FAX" is a takeoff on a favorite phrase used by Sgt. Joe Friday [Jack Webb] on the old *Dragnet* TV series; when investigating a case, Friday was frequently forced to shut down overly excited witnesses with the phrase, "Just the facts, ma'am.")

Now that I've satisfied the irresistible urge to use that cliché, let's talk about FAX. Have you been embarrassed by friends, clients or business associates asking you to "FAX it to me?" Have you thought about investing in a good FAX machine but can't justify it? Do you really want to clutter your desk with another piece of hardware?

If you have a computer and modem (and you do if you're on DELPHI), you don't have to worry about the answers to those questions, thanks to DELPHI's FAX service. You can now send text-only documents to any Group 3 FAX machine in the world (which is to say, virtually any FAX machine) as easily as you send E-mail.

FAX service is available on the DELPHI Mail menu (which is different from the Mail menu that you see if you type MAIL at the menu in the Atari Users' Group or Atari ST SIG. To reach the DELPHI Mail menu, type DELPHI at the SIG menu (or type MAIL at DELPHI's main menu). This menu is displayed:



```
MAIL Menu:

FAX Service                   Easylink
Mail (Electronic)             Translation Services
Scan for New Messages         Workspace
SetMail                       HELP
Telex                         EXIT

DMAIL>(Mail, FAX, Telex, Trans):
```

As you can see, the menu also offers gateways to regular E-mail (the "Mail [Electronic]" selection) and other services.

What Kinds of FAX Messages Can I Send?

DELPHI's FAX service is very accommodating. You can send FAX messages that you compose online or documents that you've previously uploaded to your personal Workspace. (Being able to send a Workspace file from Workspace means that you can also send the document as E-mail and/or a Telex—or resend it to other FAX machines later.)

FAX messages should be no wider than 80 columns. There is no limit on the size of a message (the number of lines it contains). Uploaded messages must be in 7-bit ASCII format to be sent. (Use your word processor's "print to disk" or ASCII-save function or a file-conversion utility to produce 7-bit ASCII files.) You cannot send graphics or binary files of any kind.

If you want to insert page breaks in a message, type /PAGE on a line by itself at the beginning of the message and type /PAGE again wherever you want a page break. (Page breaks can be included in a message you are typing online or in a document you are creating with your word processor.)

How It's Done

All you have to do to send a FAX message is have the destination number(s) ready and upload the message to be sent to your Workspace (or know what you're going to type online). Then type FAX at the DELPHI Mail menu.

DELPHI guides you through the entire process with easy-to-follow prompts. You can send the same message to multiple FAX machines by entering multiple phone numbers at the appropriate prompts. (FAX destination numbers must include the area code and phone number. In the case of messages outside the U.S., Canada and the Caribbean, you must include the country code, city code and telephone number.) You can abort the process at any time by entering ^C.

What Does it Cost?

The rates for sending FAX messages are generally a lot less than you'll pay at a walk-in FAX service center (such as those found in some office supply and computer stores).

FAX messages are measured in pages and half-pages. A FAX page contains 2,500 characters; a half-page contains 1,250 characters. Messages are also billed by pages and half-pages, as shown in Table 1.

For FAX messages sent to:	Cost for the first page:	Cost for each additional half-page:
United States	$1.25	$0.50
Canada	$2.00	$1.00
International	$7.00	$2.00

Table 1

If you send a FAX to more than one number, multiply the total page/half-page charge for the message by the number of destination numbers.

How Do I Know if It Got There?

If you send a FAX message and it is not delivered, you will get an E-mail message automatically at no charge. If you want to be notified that a message did arrive, you can receive such notification for an extra charge of 40¢ per message (the notification will arrive in the form of an E-mail message, advising you of the date and time the FAX message was delivered).

Why Should I Use DELPHI for FAX Delivery?

If you send only text FAX messages, DELPHI's FAX service is the way to go. The cost is reasonable—even attractive—considering the fact that you don't have to invest hundreds (or thousands) of dollars in a dedicated FAX machine. DELPHI FAX also offers these advantages:

* You don't have to print out and scan a message to send it; you can send it directly from your computer.

* You can send the same message to one or more FAX, Telex and/or E-mail destinations.

* You don't have to learn how to use a FAX machine.

DELPHI FAX is equally attractive to those who need to send only occasional FAX messages as well as high-volume FAX senders.

Games, Games, Games

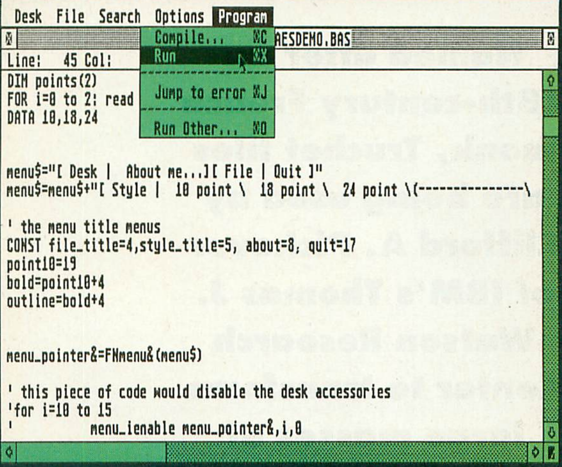
DELPHI's real-time online games continue to draw big crowds. Monday night is *FlipIt!* night. Each game played on Monday night builds a jackpot that is divided among the top finishers in each of three categories: best on 8-by-8 board, best on 10-by-10 board and best overall. Type ENT FLIP to check it out and try your hand.

Thursday nights are poker nights. From 9 p.m. to 12 midnight, poker players take part in tournaments in which they can earn free time. All players start with $1,000 in poker "chips" and can use the money to play five-card and seven-card stud. The player with the biggest pot at the end of the last game wins $30 in free time on DELPHI. Other prizes are awarded as well. You can practice and hone your poker skills at any time by typing ENT POKER at the DELPHI Main Menu.

That's it for now. See you in conference! (Tuesday evening, 10:00 p.m., Eastern time. Be there, or be an obtuse rectangle!)

In addition to science-fiction novels and books on model rocketry and other topics, ***Michael A. Banks*** *is the author of* DELPHI: The Official Guide *and* The Modem Reference, *both from Brady Books. You can write to him via E-mail on DELPHI to membername KZIN.*

HISOFT BASIC



```
DIM points(2)
FOR i=0 to 2: read
DATA 10,18,24


menu$="[ Desk |  About me...][ File | Quit ]"
menu$=menu$+"[ Style |  10 point \  18 point \  24 point \(-------------\

' the menu title menus
CONST file_title=4,style_title=5, about=8, quit=17
point10=19
bold=point10+4
outline=bold+4



menu_pointer&=FNmenu&(menu$)

' this piece of code would disable the desk accessories
'for i=10 to 15
'          menu_ienable menu_pointer&,i,0
```

HiSoft BASIC ($79.95)
HiSoft BASIC Professional ($159.95)
MichTron Inc., 576 South Telegraph
Pontiac, MI 48053
(313) 334-5700

Reviewed by Ian Chadwick.

Depending on how you look at it, this is either a great time for ST BASIC users or a terrible time. It's great because there are so many options from which to choose, including ST BASIC, Fast BASIC, GFA BASIC 2.0 and 3.0, STOS and now HiSoft BASIC. You can pick the language style you prefer: Microsoft-like, structured, game-oriented, compiled, GEM or TOS based and so on. However, because of the proliferation of BASICs, it's unlikely one will become the de facto ST standard, and portability among the various products is limited. Translation is all too often very difficult.

The latest entry into the arena is HiSoft BASIC (with its enhanced version, HiSoft BASIC Professional). MichTron obviously hopes to capture the hearts and minds of the ST users with what is their replacement for the lost GFA BASIC. With the latter emerging as the most popular of the lot, it's natural to compare the two.

Since HiSoft has made a considerable impact in England, where it originated, MichTron must assume that it should find a good following on this continent. But first they must undo the good job they did in selling GFA BASIC over the past two years. And it won't be easy.

HiSoft BASIC (referred to as HSB from here on) is a structured language based on the popular Microsoft QuickBASIC design, offering a high degree of code compatibility with QuickBASIC 3.0. Note that QuickBASIC is already at Version 4.5 on the PC, a couple of generations beyond 3.0. By now, ST users should be accustomed to the lag between product levels.

While there's nothing unique in the commands and functions to set HSB apart from other BASICs, it does stand out from the rest in that it combines a text editor and a compiler into one package. The highlight of HSB is the "seamless" interface between the two: When you write your code, you actually compile it to memory, then run it and return to the editor. This is a far cry from the cranky process of writing, exiting, compiling, testing, rewriting and so on that I complained about in a recent *Ian's Quest* column. Unlike GFA, you don't need to quit the program to go to the compiler.

The other side of the coin is that HSB lacks GFA's interactive parsing and syntax-checking. With HSB, you discover errors only at runtime or compilation. For the novice (or, like me, anyone who has trouble typing on the abysmal ST keyboard), this may not be a worthwhile . tradeoff. However, the error messages produced by HSB are clear enough and more informative than GFA's. If you choose not to continue your program when an error is found, the program halts at that line with the cursor on the problem.

The second major tradeoff is in simplicity. HSB is not as easy to use as GFA. The two (HSB and GFA 3.0) compare favorably when measuring functionality, but GFA comes out on the plus side with more features. GFA has some nice commands and functions, like PI, INP?(), TRUE, SGET and SPUT, BREAK and ON BREAK, array-swapping and sorting, plus full VT-52 terminal support that HSB lacks. On the plus side, HSB shows a considerable improvement in speed over GFA.

To its credit, HSB performs very well; the interface is fast and smooth. I haven't been able to crash it, something I cannot say for any GFA BASIC release. Compilation is equally fast, with the unfortunate caveat that you cannot change drive and path when compiling from the program, except through the inclusion of an $OPTION command within the code itself.

HSB supports the full range of GEM AES, DOS and VDI calls, as well as BIOS and XBIOS calls, in a manner similar to the standard C conventions—including the need to include the proper libraries for proper compilation and execution. This is where the simplicity prob-lem arises.There are no simplified com-mands available in GFA for creating alert boxes, handling menus or most other GEM functions. HSB does have some more sophisticated window commands than GFA, but they require a little more effort to program.

HSB has one advantage for the user converting from

(CONTINUED ON PAGE 78)

Atari Truchet

Named after an 18th-century French monk, Truchet tiles are being used by Clifford A. Pickover of IBM's Thomas J. Watson Research Center to transform large masses of binary data into visual form. His object is to determine the relative randomness of his data using a quick, visual tool.

by Frank Kweder

Truchet tiles were a topic of discussion in the "Computer Recreations" column of *Scientific American* (July 1989). Named after an 18th-century French monk, Truchet tiles are being used by Clifford A. Pickover of IBM's Thomas J. Watson Research Center to transform large masses of binary data into visual form. His object is to determine the relative randomness of his data using a quick, visual tool.

There are two types of tiles (see Figure 1), each composed of two quarter-circle arcs whose centers are located at opposite corners of a square. The radius of each circle is equal to half the length of the side of the square, thus, when tiled, the pattern is connected continuously. Tile #1 is the mirror image of Tile #0. Alternatively, you might think of Tile #1 as being rotated 90 degrees in either direction.

The program presented here has no scientific significance. It merely demonstrates some of the graphical design possibilities. When the program is first run, a preset pattern (Figure 2) is drawn using data statements. The pattern contains three main figures: a large flower-like circular figure, a large "X" shape and a smaller circle.

Press the space bar and a new pattern will be drawn by random placement of the tiles (Figure 3). After running this many times, I have noticed that the flower and "X" patterns almost never appear. The small circles, however, are common.

The larger patterns are created by a sequence of tiles in a 4x4 grid. The "X" pattern requires a repeated sequence involving 16 tiles and the "flower" requires 12 tiles. The small circle uses only four tiles. Therefore, it's logical that the "X" is least likely to appear.

Another press of the space bar will plot another random pattern. This time the figure is simplified by removing the lines defining the tile edges. The next press of the space bar will plot a full-screen pattern (Figure 4).

When the plotting is finished, you may replot it or print the plot using an Epson-compatible printer and the *G: Device* from ANALOG's October '85 issue (and on this month's disk). The file G.OBJ must be loaded before running *Truchet Tiles*. If G.OBJ is not loaded, an attempt to print will abort and the program will remind you to load it. Alternatively, you may want to add your own screen-dump or screen-save routine.

Try printing out some screens and filling in the maze-like channels the pattern forms with different colored markers, highlighters or crayons. This is a great way to keep the kids occupied and not bad therapy for anyone who has been at the keyboard for too long!

Programming Notes

A display list interrupt (DLI) is used to show the title and the type of plot being drawn. Within the DLI, I also used fine scrolling to present a caption or instructions, and to draw your attention to the next operation.

The DLI sets Line 0 to Graphics 2 and Line 1 to Graphics 1. Line 1 also has its vertical-scroll bit set (+32). All you have

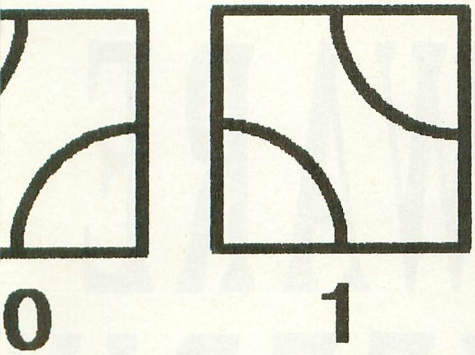
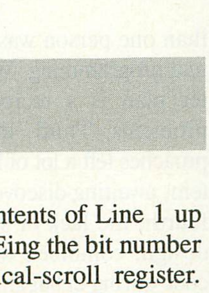


FIGURE 1

Tiles

to do is scroll the contents of Line 1 up eight bits while POKEing the bit number into 54277, the vertical-scroll register. Next, rewrite the contents of Line 1 and then scroll down eight bits, again POKEing the bit number into 54277. Check out the "Master Memory Map, Part IX" in the April '89 ANALOG Computing, where there is an excellent account of this interesting register.

I hope you enjoy *Truchet Tiles*. As for me, I must return to my copy of *Scientific American* and try to figure out how to keep the Biomorphs from eating all the fractal popcorn!



Frank Kweder, an owner of both a 130XE and an ST, reads his Scientific Americans *in Ft. Myers, Florida.*

(PROGRAM LISTINGS CONTINUED ON PAGE 96)

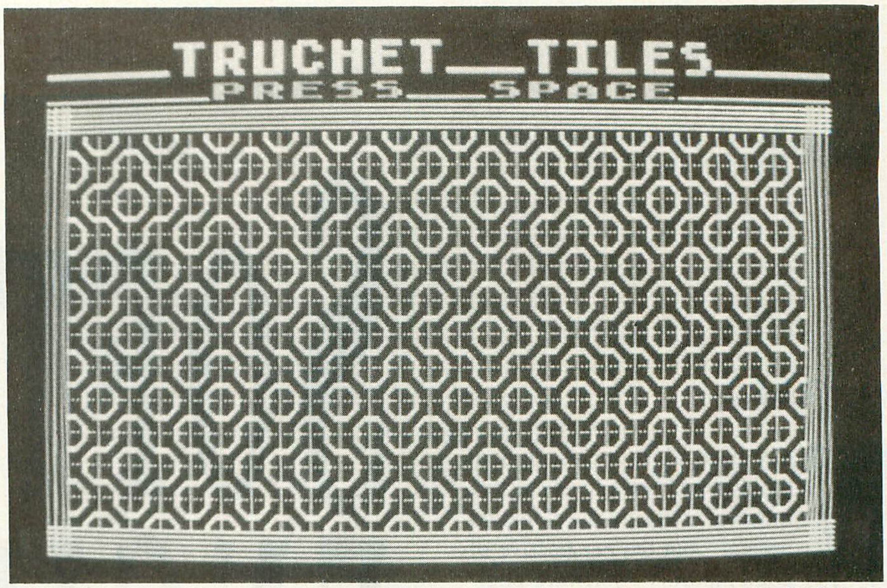


FIGURE 2

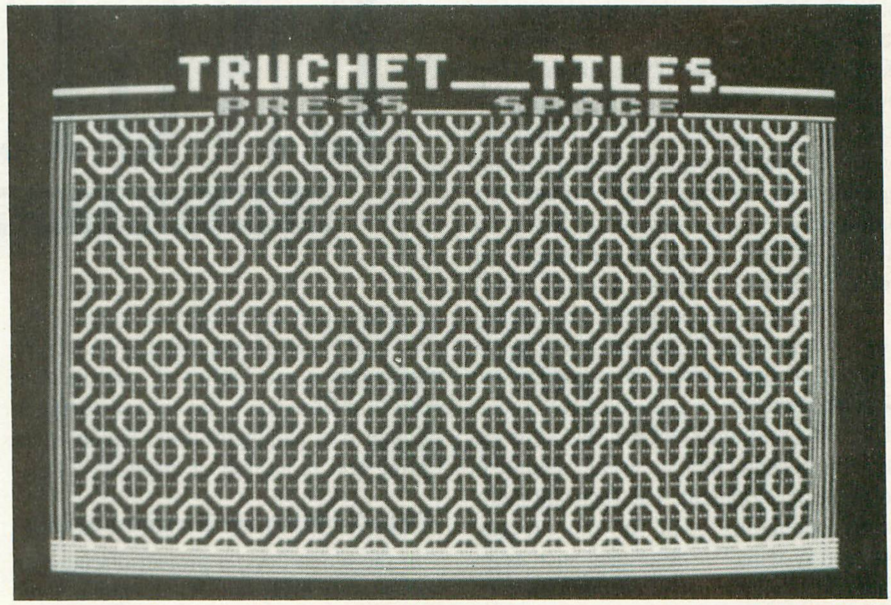


FIGURE 3

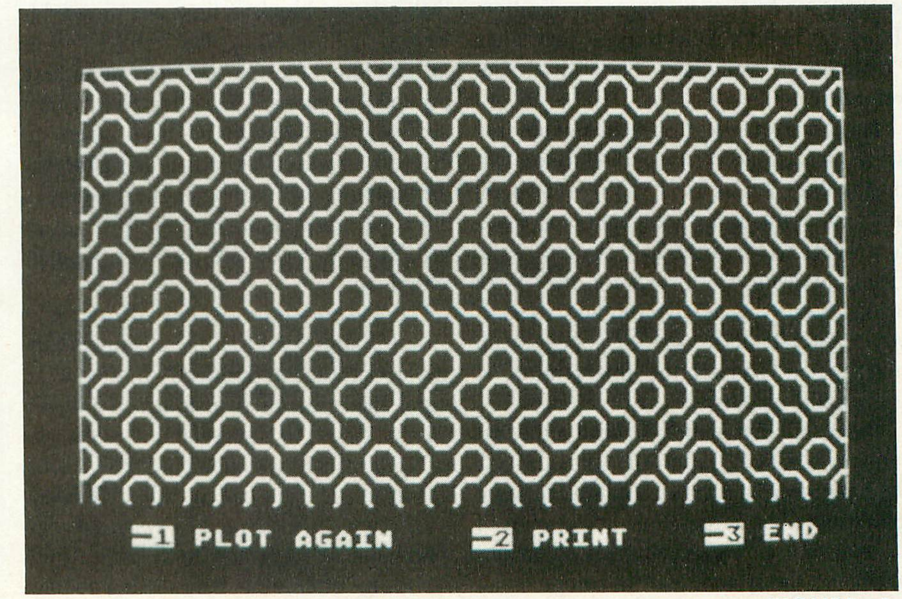


FIGURE 4

SOFTWARE ENGINEERIN

DEVELOPMENT LIFE CYCLES

by Karl E. Wiegers

We've been taking a close look at the different tasks involved in a modern software-development project from the perspective of contemporary software-engineering methodology (some may say "theology"). My thrust has been that a systematic, structured approach to your software development efforts will pay off in a more durable, robust and maintainable product that more closely matches the customer's needs, compared to a traditional program-development approach. The ultimate definition of successful software is that it is delivered on time, comes in on budget and correctly satisfies the user's actual needs. I'm convinced that software engineering holds the promise of meeting these goals.

So far, though, we've only considered one possible sequence of software-development tasks. The method I've been describing is sometimes called the "waterfall" method of software development. The waterfall method is commonly used for the process-oriented systems I've described, such as my chemistry game, *Reaction Time*. However, it is by no means the only methodology available for creating a software application. Today I'd like to explore some alternative "models" or "paradigms" for software development.

These various development models are sometimes called "software life cycles," although these cycles aren't the same as plant and animal life cycles, in which birth leads to growth, to maturity, to birth again and finally to death. With software, the "birth again" step is generally omitted. And if you're too late delivering the goods, death may appear without the intervention of growth and maturity!

These different models do have some features in common. After all, software-systems engineering must always involve some gathering of user requirements, design of the system to be constructed, construction, testing of the system and its components, and ongoing project management, documentation and quality-assurance activities. And it's a safe bet that any system, no matter how it was built, will require some level of maintenance and change during its lifetime. The paradigms differ mainly in the timing and repetition of these activities.

Pre-Software Engineering

In the olden days, a project often began with a rather fuzzy notion of the goal. Some time was spent on planning or design, and then a larger block of time was devoted to writing code. Coding was followed by an infinite loop of testing and debugging, testing and debugging. The testing phase generally was declared to be over when the delivery deadline had come and gone and the customer was beating on the developer's door. We might summarize this historic approach as a "code-and-fix" development technique.

There were some obvious deficiencies with this strategy. First, the customer's involvement was too frequently limited to an initial interview or two, followed (after a sizeable delay) by a presentation of the "final" product, which often bore little resemblance to what he had in mind. Second, insufficient time devoted to system design at both the overview and detail levels meant that things often went awry during coding. The problems were amplified when more than one person was involved with the actual programming. Working without a master plan is a recipe for communication problems. Third, the ad hoc testing approaches left a lot of bugs lurking in the system, awaiting discovery by irate users. And fourth, the lack of structure in the system design, combined with patches placed on code during debugging and the generally inadequate documentation, made system maintenance a nightmare.

Figure 1 graphically illustrates one consequence of this classic life cycle. As time goes on, the developer's impression of what the user wants and the customer's actual expectations tend to diverge. The bigger the gap, the lower the perceived quality of the final system by the user. More modern development methods are intended to reduce this expectation gap. The goals are to have faster and more accurate development of the correct software system.

The Waterfall Model

The waterfall model for software development is essentially a more disciplined variant of the classic life cycle. This is the development method I've been describing over the past several months. Basically, the waterfall model consists of several discrete phases, each of which has specific deliverables (products) that must be completed and

G

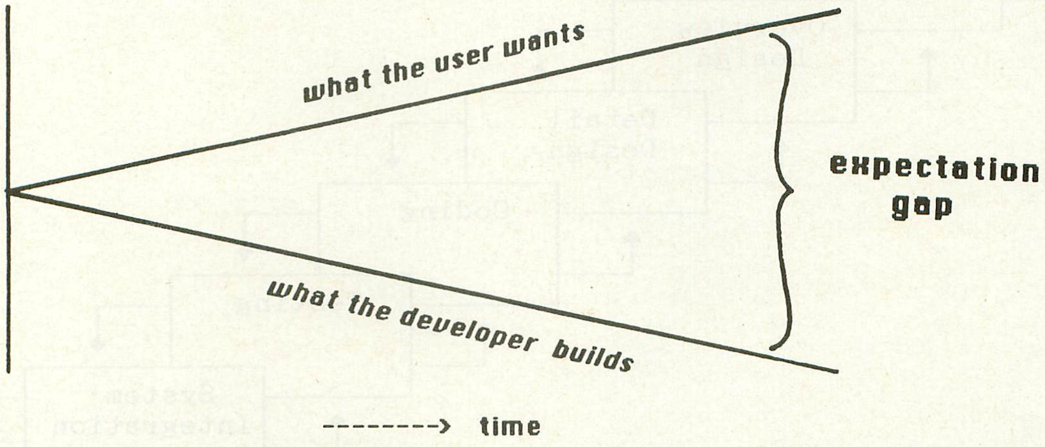


FIGURE 1. EXPECTATION GAP FOR CLASSIC LIFE CYCLE DEVELOPMENT.

approved before proceeding to the next step. These phases go by various names; one set of labels is shown in Figure 2 (I've seen at least four published variants of this diagram, so I made up a fifth). Sometimes you'll hear of "system specification" instead of "requirements analysis," or "implementation" instead of "coding," but the ideas are the same.

If you've been reading this software-engineering series all along, you'll recognize the pieces of the diagram in Figure 2. Notice that each phase involves some feedback to the previous phase (the upward-flowing arrows). This reflects the fact that defects become detected as work goes on, and the defects usually are attributable to errors made in an earlier phase. The best way to handle these problems is to backtrack to the source of the error and correct it there, rather than simply slapping a patch on it at the later stage when the problem showed up. For example, a single error at the requirements phase may lead to several problems during design, so your best bet is to go back and fix the underlying requirements flaw before continuing with design.

The waterfall method also differs from the classic life cycle by using a variety of techniques to improve quality and productivity in each phase. Requirements-analysis benefits from the data-flow (DeMarco) methods used for structured specification. Data-flow techniques also are valuable in high-level design (Yourdon, and Gane and Sarson methodologies). Module-level design is facilitated using structured English (pseudocode) or action-diagram techniques, among other diagramming methods. Structured programming methods lead to code that is readable and maintainable. Structured testing procedures (both white-box and black-box) prevent many errors from seeping through. Quality-assurance activities, such as structured walkthroughs, should be performed before proceeding to the subsequent phase. And many computer-aided software engineering (CASE) tools are available to automate some of these tasks. We've discussed all of these ideas in earlier articles in this series. The key word throughout is "structure."

From the quality perspective, managing a project being developed under the waterfall method poses particular challenges. Because of the sequential flow of tasks, it's important to build in high quality from the very start. It's difficult to go back and retrofit quality after a program has been completed. The quality of the code depends upon that of the system design, which in turn depends upon the quality of the specifications, and so on. One way to manage this is to break each phase of the project into separate substeps, with a checkpoint quality review scheduled at the end of each substep. You don't continue with the project until each subphase meets its goals. This subdivision is the basis for some software project management methodologies, which we'll discuss in a future installment.

One problem with the waterfall approach is that the customer doesn't see any usable products until near the end of the development effort. I've found it valuable to review the specifications and perhaps even the system design with the customers, but this isn't the same as letting the user get his grubby paws on a keyboard. A long time lag between inspiration and delivery can reduce the customer's interest in, or need for, the system, as well as lead to a bigger gap between what he wants and what he gets, as we saw in Figure 1.

Another drawback to the waterfall life cycle is the need to lock the system requirements once the analysis phase is complete. Of course, in real life, it's rare that requirements are truly frozen. However, due to the long sequence of steps between concept and completion, any changes requested by the customer, once design is under way, have repercussions throughout the system-development effort. It's difficult enough dealing with errors that are found at later stages of development, let alone coping with

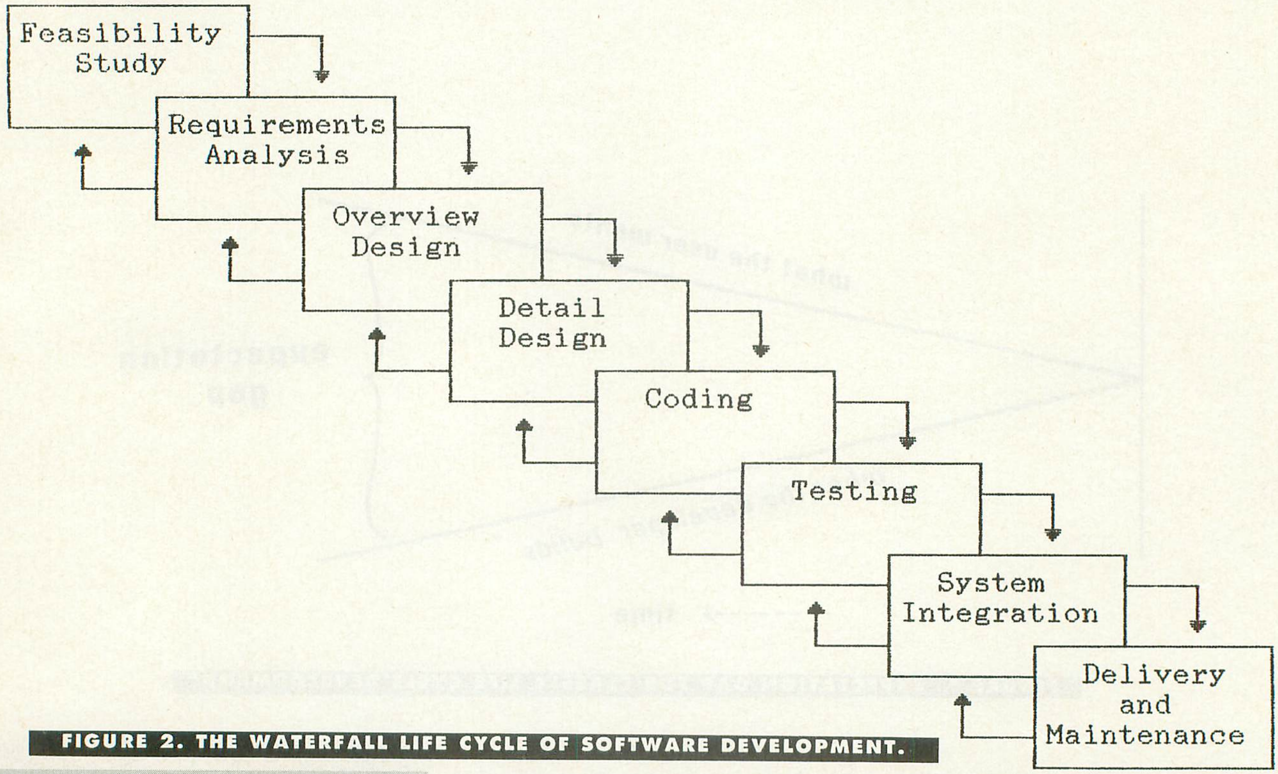


FIGURE 2. THE WATERFALL LIFE CYCLE OF SOFTWARE DEVELOPMENT.

volatile requirements. This is a particular problem in dynamic technical environments, such as the scientific research organization I work in. Since system requirements often do change as time goes on, the software developer needs a better way to cope with this.

Iterative Prototyping

Iterative prototyping is an alternative software-development approach that aims to avoid some of the deficiencies of the waterfall method. This model addresses the problem of having incomplete or uncertain specifications at the beginning of the project, as well as coping with anticipated changes in the specs during the time needed for development. Prototyping tends to be a user-driven development method, rather than the prespecified approach characterized by the waterfall paradigm. Prototyping is less formal and more dynamic, involving continuous incremental change as the user's needs become more precisely defined.

Human beings find it easier to critique than to create. The idea behind software prototyping is to get something into the customer's hands quickly as a first cut, so the user can tell you whether or not this is what he has in mind. Revising the prototype leads to an interim product that more closely meets the user's needs. By repeating this prototype-review-modify cycle as many times as necessary, the divergence problem illustrated in Figure 1 is reduced to something like that shown in Figure 3. I've found that frequent iteration with the customer is vital to success, even when I'm not following a true prototyping life cycle.

When the developer delivers something that closely matches what the user has in mind, the user's perception of system quality is greatly increased. The prototype aids the effort by serving as a vehicle to help gather and refine the system requirements, as well as being a preview of things to come.

The main idea behind prototyping is speed. Users always change their perceptions of what they want after using a new system for a while. If the new system can be constructed and modified quickly, we developers can do a better job of uncovering the ambiguities, inconsistencies, misunderstandings and omissions that plague conventional system-specification efforts.

A prototype can be either a mock-up or a working model of the ultimate system being constructed. A mock-up consists of sample screens and reports, with limited functionality built in. You may write enough code to let the user move from screen to screen in a logical way and enter representative data, but the brains behind the screens are either absent or faked. Such a system has been called a "vacuous prototype." It's valuable for simulating a user interface, but it doesn't do any useful work. For a system that is user-intensive, this can be a valuable way to help the user evaluate whether the "look-and-feel" of the system you propose to build will suit his wishes.

A working-model prototype might consist of a full implementation of just part of the system. There may be just a few screens and one sample report, but with the code to convert inputs into outputs fully functional for that limited subset of the system. This kind of prototype will let the user see if he is comfortable with the approach you propose, and since it actually does useful work, you may be able to incorporate the working model into the final system.

Notice that this iterative method doesn't draw sharp lines between the development phases, as the waterfall method did. You can do a little requirements gathering, a little design, a little implementation, a little testing and a little delivery. Then, based on feedback from the customer, you do a little more requirements gathering, a little more design and so on. You continue with the iteration process until either: (a) you've reached a point where you can now write the actual, deliverable system quickly and accurately, or (b) the final prototype *is* the deliverable system.

Most computer systems don't have specific tools to facilitate prototyping, although more are appearing all the time. A software-prototyping environment might include a screen painter, a report generator and some kind of interpreted language to let you hook the pieces together into a running system. The screen painter lets you quickly design screen displays and write enough code to let the user display them and move from one to another. The report generator can produce printed reports or screen displays from specifications without writing huge amounts of line-by-line print-formatting statements. Often you can use default-printout formats to get some output quickly and let the user then fine-tune the layout for complete implementation at some later time.

The interpreted language lets you display screens, do some actual processing and produce output. Interpreted languages often lack some of the rigor of their compiled brethren (like insistence on declaring all variables before using them), and they don't need the compile and link steps before execution. Hence, it's faster to write and try working code with an interpreter. Since computational efficiency is not paramount for a prototype, the fact that interpreted languages execute more slowly than compiled languages is not a concern. Also, interpreted languages often have excellent debugging facilities, which can facilitate quick turnaround in response to user complaints.

But what happens when you've reached the last iteration in the prototyping sequence and the customer is happy with the current status? You could simply say, "Okay, here it is; I'm done." But if you've written the code in an interpreted language, the performance at that stage may be inadequate for a production system. Also, you've probably neglected code documentation since you've been working on "just a prototype." The customer may not recognize the difference between a working model and a finished product, which would contain all of the software quality-assurance characteristics that we build into our systems. A prototype may well have traded both quality and efficiency for the ability to quickly demonstrate desired system capabilities. If you don't redo the final prototype, you may wind up with these shortcomings as integral parts of the delivered software; this may or may not be a problem.

The alternative is to keep the screens, reports, user interface and algorithms from the prototype, but recode all the procedural routines in a more efficient language, such as C, COBOL or Pascal. Unfortunately, programs in such languages often are more costly to maintain than the simpler code generated in a prototyping environment. The choice depends on the application itself. It certainly can be aggravating to finally get the system just the way the user wants it, only to know that you have to rewrite much of it in another language before you're done. The saving grace is that the iterative-prototyping process, emphasizing as it does user involvement, has done a good job of pinning down the user's precise system requirements, so the recoding step should go quite smoothly.

Unfortunately, GEM doesn't lend itself easily to the prototyping model. Atari BASIC did just fine, as long as you worked within its limitations. Getting an application up and running in a full-screen windowing environment, like GEM or the Macintosh, takes quite a bit of effort; it's not something you throw together in a few hours just to see how your ideas might look and act. But software-engineering researchers (yes, Virginia, there are some) are actively pursuing the development of rapid prototyping tools for windowing environments. Patience, please, for a few more years.

Incremental Development

Another software-development approach that can be effective is incremental development. In this scheme, you begin with the specifications for the first part of the target system and go ahead and develop it using whatever methodology is most appropriate. Placing a completed portion of the ultimate system into the hands of the users quickly provides them with some real functionality while leaving everyone more time to refine the specifications for the rest of the system.

Incremental development breaks the development project into a series of small individual projects that can be completed independently, each portion being appended to the parts you've already finished. The rapid turnaround between the customer's request and delivery of something useful reduces the usual gap between initial expectations and those that would prevail much later when the complete system otherwise would be delivered. This method also can reduce the time-dependent evolution of system specifications, which always seems to be one step ahead of a traditional development effort.

Obviously, not every project lends itself to incremental development. Some systems really have to be done as a massive whole that is integrated all at once. Another concern is that each sequentially completed portion of the system must link up nicely with the parts already in place. I've found this method to work well if a first cut at the overall system specification has been com-

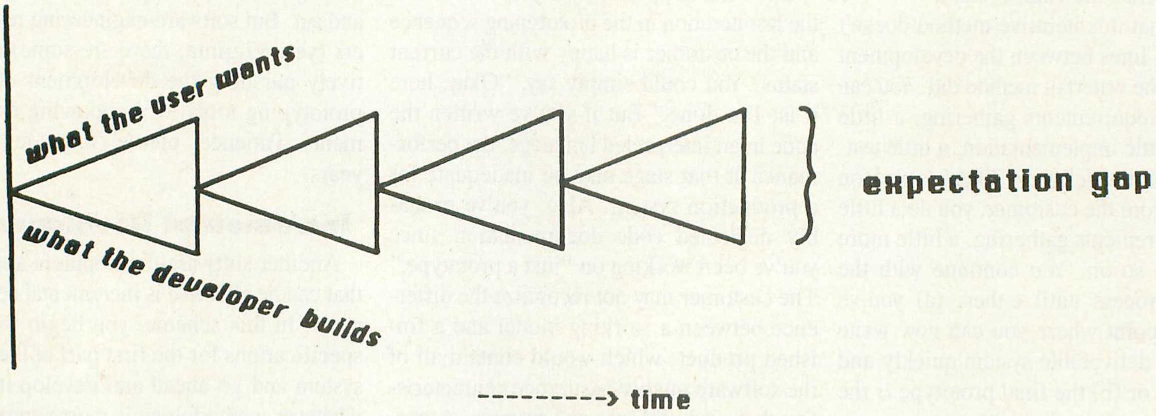


FIGURE 3. EXPECTATION GAP FOR ITERATIVE PROTOTYPING DEVELOPMENT.

pleted, so that we can intelligently partition the system into chunks that can be implemented individually. The detailed design of the subsequent pieces can wait until the first section has been fully implemented.

One danger with incremental development is that changes made in the specifications for the later parts of the system can result in the portions completed initially needing major revision or even being scrapped. On the other hand, it may turn out that the latter sections don't even have to be completed, due to the customer's experience with the part you finish first. It's hard to accurately predict how things will turn out with incremental development, but keep it in mind as an option for your next big project.

Fourth-Generation Languages

One other methodology for system development bypasses the traditional code-writing phase entirely, using what are referred to as "fourth-generation languages," or 4GLs (sometimes more generally termed "fourth-generation techniques," 4GTs).

You're used to the traditional software-development process using third-generation high-level languages like C, BASIC or FORTRAN. These "procedural" languages require the programmer to write in grim detail the instructions for each task performed in a program. We must individually manipulate each piece of data, handle all the screen-display interactions, define printout formats line by line, individually validate each user entry, handle the details of every file access and so on. The many lines of code required for all these functions lead to a major maintenance effort whenever a large system is enhanced or a bug eradicated.

Fourth-generation languages allow a "non-procedural" approach to certain domains of software development. A 4GL is often supplied as part of a relational database software package. (See the April '88 issue of ST-LOG for an informative introduction to relational databases by Frank Cohen.) Tools are provided to help implement the user interface and printed-output designs, and some kind of high-level language is available for communicating with the database. But you, both as developer and as user, are insulated from the nitty-gritty details of reading and writing files, performing basic arithmetic operations and writing all the code needed to handle complex screen displays.

At their best, 4GLs eliminate many of the tedious steps endured by the experienced applications developer. But this pot of gold doesn't come free. The main drawbacks of contemporary 4GLs (of which there are many) are that they are CPU hogs (computerese for saying that they execute inefficiently) and that they can be applied only to limited classes of fairly specialized applications. As I mentioned, these applications generally center about information stored in relational databases. Of course, a huge assortment of business applications fits in this very category, but the 4GL hasn't had much of an impact on scientific computing yet.

Not surprisingly, writing an application using a 4GL usually involves some programming, although at a higher level of abstraction than you see with a 3GL. Many fewer lines of code may be needed to perform a specific task using a 4GL. The most common language used for relational database accesses is called "Structured Query Language" or SQL (pronounced "sequel"). Rather than making you mess with opening files, reading records and so on, a query language like SQL lets you tell the computer what you want to do in English-like sentences.

Let's consider a simple example. Relational databases consist of two-dimensional tables containing rows and columns of information. Suppose you have a table called MAGAZINES, which lists all the magazines to which you subscribe (maybe you're a library). When you set up this table, you created columns with names like TITLE, COST, EXPIRATION DATE and THEME. Each row you add to the table would contain information for a particular magazine. ANALOG might be entered with a title of (guess what) "ANALOG", a cost of "$28," an expiration date of "8/90" and a theme of "Atari ST computing."

Imagine that you want to write a program to extract specific rows from this MAGAZINES table according to particular criteria, sort them in some way and display certain columns from the rows retrieved. Specifically, let's find the titles of all the magazines that contain the keyword "Atari" in the THEME column. You can probably imagine the many steps needed to do this in a language like BASIC. But in SQL, you'd write a concise program statement, such as: SELECT TITLE FROM MAGAZINES WHERE THEME CONTAINS "Atari." Note the close resemblance between this SQL statement and the English description of what we want to do.

This is not a tutorial on SQL, so we won't pursue it further. The main idea is that a query language like SQL lets the developer or customer do the conceptual work, while the computer handles the low-level stuff. Just because a 4GL is nonprocedural, doesn't mean there isn't any programming involved; it's just done at a higher level of abstraction. In fact, the SQL statements needed for elaborate queries from multiple tables in a large database can get horrifyingly complex.

There's some overlap of system tools between 4GL and prototyping environments. Both systems usually include a screen painter, which lets you quickly design a form on the screen to facilitate data entry by (or display to) the user, along with easily handling data validation and displaying help screens. Both systems often include a report painter and report-generator capability to easily lay out and produce even sophisticated printouts of information retrieved from the database. Some 4GLs involve a language for defining reports that looks suspiciously procedural to me. And a smart 4GL can actually generate the code automatically for many of the database accesses (queries and updates) you need to perform.

On the plus side, a 4GL environment can be a real asset for rapid system-development (for the right kind of system), and by removing much of the detailed code-writing, the maintenance requirements for a system built using a 4GL can be much reduced from those for a comparable system based on a 3GL. However, designing complex customized screens or writing complex queries can result in the same system developer's headaches we're used to from older technologies. And sometimes your system requirements may involve capabilities beyond those of a 4GL, such as graphics or heavy-duty computations. Many 4GLs deal with this by providing ways to call a program written in a 3GL from within the 4GL environment. Does this average out to a 3.5-generation language?

One other advantage touted for the 4GLs is the goal of having much of the simpler applications development performed by the end user, rather than by a professional software developer. This is certainly a worthy goal in light of the serious work backlogs plaguing many software-development shops. While this may be possible with some current 4GLs, at least one of the most popular relational database systems available has a 4GL so complex (but powerful) that it turns my hair white. This may all be a clever ploy to provide permanent job security for the professional software developer.

Looking Back

Today we've explored a variety of strategies for developing applications software packages, including some interesting alternatives to the traditional waterfall method. These techniques are by no means mutually exclusive. Often you can benefit from a hybrid approach: some initial analysis for a general picture of the problem, followed by a little prototyping to get something into the user's hands, then either incremental or full-blown waterfall development, depending on how firm the requirements are when you're ready to start.

The goal of each of these development life cycles is to meet our definition of high-quality software as closely as possible. In case you've forgotten, high-quality software is delivered on time and on budget, and it meets the user's needs. The developer's challenge is to be able to share the customer's vision of the final system, then refine it (based on his own experience) for performance, ease of use and reliability. In future articles we'll talk more about software quality-assurance (making sure these goals are met) and software project-management (mechanisms for meeting the goals).

Karl Wiegers, Ph.D., spent the '70s learning how to be an organic chemist, then spent the '80s wrestling with computers. He is now a software engineer in the Eastman Kodak Company photographic research labs. He hasn't yet selected a career for the '90s.

B&C ComputerVisions

3257 KIFER ROAD
SANTA CLARA, CA 95051
(408) 749-1003
(408) 749-9389 FAX

STORE HOURS
TUE - FRI 10am - 6pm
SAT - 10am - 5pm
CLOSED SUN - MON

8-BIT INTEGRATED CIRCUITS

$3.50 EACH:

ASSEM REV.A
BASIC REV.A
MPU 6507
PIA 6532
800 CPU 6502
810 ROM C
800 ANTIC
OS ROMS (499B-599B)
RAM 6810
POKEY
PIA 6502

$4.50 EACH:

1771 FDC
1050 ROM
XL CPU 14806
XL/XE MMU
GTIA
XL DELAY

$12.00 EACH:

ASSEM REV.B
VCS TIA 444
FREDDIE
XE GATE ARRAY
BASIC REV C
XL/XE OS
XL ANTIC
850 ROM B

NEW PRINTED CIRCUIT BOARDS WITH PARTS

800 MAIN/CHIPS ...	10.00	800 10K O/S ..	10.00
800 16K RAM	10.00	800 POWER	5.00
810 SIDE WITH D/S	15.00	800 CPU W/GTIA	10.00
810 ANALOG	10.00	810 POWER	15.00

JOYSTICKS

ATARI STANDARD(2)	12.00	WICO 3-WAY ..	14.95
EPYX 200 XJ	16.95	WICO BAT HNDL.	22.00
EPYX 500XJ	19.95	POWER PLAYER .	29.95
SLICK STICK	9.95	ERGOSTICK	24.95
TAC-2	12.99	STARMASTER ...	14.95
TAC-3	14.95	TAC-5	19.95

RUN MAC SOFTWARE ON YOUR ST!!

MAGIC SAC	59.95	64K MAC ROMS .	9.95
SPECTRE	179.95	128K MAC ROMS	159.95
TRANSLATOR ONE ..	279.95	MAC FINDER ..	44.95
DISCOVERY OPT.0	188.00	DISC.OPT.2 ...	266.00

8-BIT DISK DRIVES

ATARI 810	130.00
ATARI 810 WITH NO CASE	85.00
ATARI XF551	199.95
B&C 810	130.00
1050 WITH SUPER ARCHIVER	250.00
1050 WITH HAPPY ENHANCEMENT	275.00
1050 WITH US DOUBLER INSTALLED	225.00
1050 RECONDITIONED	175.00
INDUS GT	175.00
ASTRA 1001 WITH PRINTER INTERFACE ...	175.00

POWER PACKS

65XE	12.00	2600	5.00
800/810/1050 ...	10.00	1027	29.95
800XL/130XE	20.00	520 EXTERNAL ...	50.00
830/835	10.00	520/1040 INTRNL	75.00
1030	10.00	ST DISK DRIVE ..	35.00

HAPPY ENHANCEMENT 810 OR 1050 $99.95
1050 SUPER ARCHIVER CHIP $69.95

DE RE
ATARI
$10.00

ST INTEGRATED CIRCUITS

DISK CONTROLLER WD1772	25.00
PHOTO COUPLER PC900	2.95
YAMAHA SOUND CHIP	10.00
6850 ACIA	3.95
68000-8 CPU	27.00
KEYBOARD CHIP	15.00
DMA CONTROLLER	26.00
MMU	28.00
VIDEO SHIFTER	26.00
GLUE CHIP	28.00
BLITTER	39.95

ATARI 520/1040/MEGA ST SOFTWARE

ENTERTAINMENT

10TH FRAME BOWLING .	22.50
A DAY AT HE RACES ..	35.95
ADULT POKER	17.95
ALF'S FIRST ADVENTURE	13.50
ARTURA	35.95
AFTERBURNER	35.95
ALIEN SYNDROME	35.95
AXE OF RAGE	44.95
BACKGAMMON	31.50
BAL. OF POWER 1990 .	44.95
BARD'S TALE	17.95
BATMAN	35.95
BATTLE CHESS	44.95
BATTLEHAWKS 1942 ...	44.95
BISMARK	35.95
BLOCKBUSTER	44.96
BORODINO	53.95
CAPTAIN BLOOD	44.95
CARRIER COMMAND	14.95
CENTERFOLD SQUARES .	26.95
COLOSSUS CHESS	44.95
DEJAVU	44.95
DEJAVU II	44.95
BATTLE TECH	44.95
DEATH BRINGER	35.95
DEAMON'S WINTER	35.95
DOUBLE DRAGON	35.95
DRAGONNINJA	35.95
DUNGEON MASTER	35.95
EMPIRE	44.95
F-16 COMBAT PILOT ..	44.95
FALCON	44.95
FALCON MISSION 1 ...	35.95
FIREZONE	31.50
GAUNTLET	44.95
GAUNTLET II	44.95
GOLDEN PATH	4.95
GUILD OF THIEVES ...	9.95
HEROES OF THE LANCE.	35.95
HOSTAGE RESCUE MISS.	40.50
HOLE-IN-ONE MIN.GOLF	26.95
HUNT FOR RED OCTOBER	44.95
IKARI WARRIORS	35.95
IMPOSSIBLE MISSION 2	17.95
INTERNATIONAL SOCCER	35.95
INTO THE EAGLES NEXT	35.95
JAWS	35.95
JET	44.95
JINXTER	9.95
JOKER POKER	53.95
KARATE KID II	35.95
KENNEDY APPROACH ...	44.95
KING OF CHICAGO	21.50
KINGS QUEST	44.95
KINGS QUEST II	44.95
KINGS QUEST III	44.95
KINGS QUEST IV	44.95
KNIGHT ORK	9.95
KULT	44.95
LAST DUEL	17.95
LEADERBOARD DUEL PACK	22.50
L.E.D. STORM	17.95
LEISURE SUIT LARRY ..	35.95
LEISURE SUIT LARRY II	44.95
LOST DUTCHMAN'S MINE	44.95
MANHUNTER	44.96
MARBLE MADNESS	31.50
METROCROSS	22.50
MICROLEAGUE B BALL II	53.95
MOEBIUS	35.95
NINE PRINCESS AMBER .	13.50
OBLITERATOR	22.50
OIDS	31.50
OFF SHORE WARRIOR ..	35.95
OPERATION WOLF	35.95
OUTRUN	44.95
PALADIN	35.95
PALADIN QUEST DISK .	22.50
PAWN	9.95
PIRATES	40.50
PLATOON	35.95
POLICE QUEST	44.95
POLICE QUEST 2	44.95
POWER PLAY	35.95
POPULOUS	44.95
PREDATOR	35.95
PSYCHO	26.95
ROBOCOP	35.95
RUNNING MAN	44.95
SCRABBLE	35.95
SENTRY	9.95
SHADOWGATE	44.95
SPACE QUEST	44.95
SPACE QUEST II	44.95
SPACE QUEST III	44.95
STARGLIDER	14.95
STRIP POKER II	35.95
SUPER HANG-ON	35.95
TANGLEWOOD	35.95
TASS TIMES TONE TOWN	35.95
TECHNOCOP	44.95
TEST DRIVE	35.95
TETRIS	26.95
TRACKER	9.95
TV SPORTS FOOTBALL..	44.95
UMS	14.95
VEGAS CRAPS	31.50
VEGAS GAMBLER	31.50
WESTERN GAMES	26.95
WINTER GAMES	35.95
WAR IN MIDDLE EARTH	44.95
XYBOTS	35.95
ZAK MCKRACKEN	40.50
ZANY GOLF	17.95
ZERO GRAVITY	26.95

MULTI-GAME PACKS

ACTION ST	35.95
ARCADE FORCE FOUR ..	44.95
AWSOME ARCADE ACTION	44.95
COMPUTER HITS	53.95
COMPUTER HITS II ...	35.95
FIVE STAR GAME PACK	44.95
MEGA PACK #1	35.95
MEGA PACK #2	35.95
TRIAD VOL. 1	53.95

PROGRAMMING

ASSEMPRO	53.95
C-BREEZE	44.95
DEVPACK	89.95
GFA BASIC 3.0	89.95
HISOFT BASIC	71.95
HISOFT C INTERPRETER	89.95
LASER C	179.95
LASER C DB	62.95
LOGO ST	9.95
MARK WILLIAMS C	161.95
MARK W. C DEBUGGER .	62.95
TEMPUS II	62.95

UTILITY

BACKUP	35.95
CODEHEAD UTILITIES .	26.95
G+ PLUS	31.50
GOGO ST	31.50
HOTWIRE	35.95
IB DISK UTILITIES ..	31.50
MULTIDESK	26.95
NEO DESK 2.0	44.95
PROCOPY	31.50
R.A.I.D.	44.95
REVOLVER	44.95
TURBO ST	44.95
WERKS	40.50

PRODUCTIVITY

1ST WORD PLUS	89.95
ART & FILM DIRECTOR	71.95
AWARD MAKER PLUS ...	35.95
BASE TWO	53.95
CAD 3D 2.0	80.95
CALAMUS	269.95
CALAMUS FONT EDITOR.	89.95
CERTIFICATE MAKER ..	26.95
COMPUTE YOUR ROOTS .	35.95
CYBER CONTROL	53.95
CYBER PAINT	71.95
CYBER TEXTURE	44.95
DATAMAMAGER ST	71.95
D.E.G.A.S. ELITE ...	53.95
EASYDRAW/SUPERCHARG	134.95
EASY TOOLS	44.95
EZ-SCORE PLUS	134.95
EZ-TRACK	58.50
EDIT TRACK	179.95
FLASH	26.95
HABACOMM	9.95
HABAVIEW	19.95
HI-TECH MAIL PRO ...	53.95
HI-TECH ACCTS PAYBLE	62.95
HI-TECH GEN. LEDGER	62.95
INTERLINK	35.95
INVENTORY MASTER ...	89.95
KEY MARK	13.40
LABEL MASTER ELITE .	40.50
LDW POWER	134.50
LOTTO-FILE	17.95
LOTTO LUCK	26.95
LUDWIG	134.95
LOGISTIX JR. VERS ..	89.95
MICROSOFT WRITE	49.95
MIDISOFT STUDIO	89.95
MUSIC STUDIO 88	53.95
NEOCHROME	35.95
OCP ART STUDIO	9.95
PAGE STREAM	179.95
PARTNER ST	44.50
PHASAR	80.95
PRINTMASTER PLUS ...	35.95
PUBLISHER ST	116.95
SHADOW	26.95
STOS	53.95
STOS COMPILER	31.50
STOS SPRITES 600 ...	26.95
STOS MAESTRO	44.95
SUPERBASE PERSONAL .	134.95
SWIFTCALC	71.95
TOUCHUP	161.95
WORDUP	71.95
WORD WRITER ST	71.95
WORD PERFECT	195.00

EDUCATION

FIRST SHAPES	44.95
FIRST LETTERS/WORDS	44.95
SPELLING BEE	26.95
MATH TALK	44.95
MATH BLASTER PLUS ..	35.95
MAVIS BEACON TYPING	44.95
SKYPLOT	89.95
TBI PRE-CALCULUS ...	44.95
TBI CALCULUS	44.95
TBI ALGEBRA I	44.95
UNICORN:	
AESOP'S FABLES ...	44.95
ALL ABOUT AMERICA.	53.95
ANIMAL KINGDOM ...	35.95
ARIBIAN NIGHTS ...	44.95
DECIMAL DUNGEON ..	35.95
FRACTION ACTION ..	35.95
MAGICAL MYTHS	44.95
READ & RHYME	35.95
READARAMA	44.95
WORD MASTER	44.95

PRICES SUBJECT TO CHANGE WITHOUT NOTICE - ALL SALES ARE FINAL

B&C ComputerVisions

3257 KIFER ROAD
SANTA CLARA, CA 95051
(408) 749-1003
(408) 749-9389 FAX

STORE HOURS
TUE - FRI 10am - 6pm
SAT - 10am - 5pm
CLOSED SUN - MON

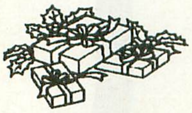
800/XL/XE SOFTWARE
ALL TITLES ON DISK

ENTERTAINMENT

12 ADAMS ADVENTURES ..	14.95
221B BAKER STREET	8.95
ALIANTS	26.95
ALT. REALITY CITY.....	26.95
ALT.REAL DUNGEON	26.95
ASSULT FORCE 3D.......	19.95
AUTO DUEL	35.95
BEYOND CASTLE WOLF....	14.95
BANDITS	5.35
BISMARK	26.95
BLOCK BUSTER	7.95
BORDINO:1812	22.50
BOULDERDASH CONSTR.SET	17.95
BRUCE LEE	8.95
CASTLE WOLFENSTEIN....	14.95
CHAMP. LODE RUNNER ...	26.95
CONFLICT IN VIET NAM .	10.95
CONAN	8.95
COSMIC TUNNELS	9.95
CRUSADE IN EUROPE	10.95
D-BUG	7.95
DALLAS QUEST	7.95
DELUXE INVADERS	7.95
DECISION IN DESERT ...	10.95
F-15 STRIKE EAGLE	14.95
FIGHT NIGHT	17.95
GAUNTLET (64K)	31.50
DEEPER DUNGEONS.....	22.50
GRAPHIC ADVENTURES 1-6	26.95
GUNSLINGER	26.95
HARD HAT MAC	7.95
INFILTRATOR	13.50
JAWBREAKER	9.95
KARATEKA	13.50
KNICKERBOCKERS	13.50
KENNEDY APPROACH	10.95
LAST V-8	8.95
L.A. SWAT/PANTHER	8.95
LEADERBOARD	13.50
LODE RUNNER	13.50
MICROLEAGUE BASEBALL..	35.95
NAPOLEON AT WATERLOO .	22.50
NEVER ENDING STORY ...	8.95
MONTEZUMA'S REVENGE...	14.95
MOUSEQUEST............	17.95
MOON SHUTTLE	7.95
NATO COMMANDER	10.95
NIBBLER	12.95
NINJA	8.95
OIL'S WELL	9.95
O'RILEY'S MINE	9.95
PENSATE	14.95
PIE MAN	14.95
PIRATES OF BARB. COAST	22.50
PITFALL/DEMON ATTACK .	13.50
PREPPIE I & II	9.95
ROME & THE BARBARIANS	17.95
SILENT SERVICE	12.95
SOLO FLIGHT	12.95
SPITFIRE 40	31.50
SPY VS. SPY I & II ...	9.95
STARBASE HYPERION	7.95
STARFLEET I	44.95
STAR RAIDERS II	17.95
STOCKMARKET	22.50
STRIP POKER	26.95
SUMMER GAMES	17.95
TAX DODGE	9.95
TEMPLE OF APSHAI	9.95
THE HULK	5.35
THEATRE EUROPE	8.95
TOP GUNNER	10.95
TRAILBLAZER	26.95
ULTIMA II	35.95
ULTIMA III	35.95
ULTIMA IV	53.95
UNIVERSE	44.95
WINTER CHALLENGE	13.50
ZAXXON (400/800)......	13.50

PROGRAMMING

DOS 2.5	7.95
DOS XE................	10.00
DISK I/O	26.95
KYAN PASCAL	62.95
LIGHTSPEED C	35.95



PRODUCTIVITY

ANIMATION STATION	79.95
ATARIWRITER+/SPELLCHK	44.95
ATARIWRITER 80	
REQUIRES XEP80	44.95
ATARI BOOKKEEPER	14.95
AWARDWARE (1050)	13.50
BANK STREET WRITER....	14.95
BLAZING PADDLES	31.50
COMPUTE YOUR ROOTS ...	35.95
DATAMANAGER	8.95
ELECTRONIC CHECKBOOK .	8.95
FAMILY FINANCE	6.95
FORMS GENERATOR	21.50
HOME ACCOUNTANT	19.95
HOME FILING MANAGER. .	6.95
HOMEPAK	24.95
INVENTORY MASTER	80.95
LETTER WIZARD	29.95
MONEY MANAGER	8.95
MUSIC CONSTRUCTION SET	13.50
NEWSROOM (1050 - 64K).	13.50
NEWS STATION	26.95
NEWS STA. COMPANION. .	26.95
PAGE DESIGNER	26.95
PAINT	12.95
PRINT POWER (1050)....	13.50
PRINTKIT (1050)	13.50
PRINTSHOP	34.95
P.S. COMPANION (64K) .	24.95
P.S.GRAPHICS LIBRARY 1	17.95
P.S.GRAPHICS LIBRARY 2	17.95
P.S.GRAPHICS LIBRARY 3	17.95
PROOF READER	17.95
PUBLISHING PRO	35.95
RUBBER STAMP	26.95
SYNTREND	14.95
SUPER MAILER	35.95
THE LOTTO PROGRAM	17.95
TIMEWISE	6.95
TURBOWORD 80	*44.95
TURBOFILE 80	*44.95
* REQUIRES XEP80	
VIDEO TITLESHOP (64K).	26.95
VIRTUOSO	29.95
VISICALC	24.95

EDUCATION

BUZZWORD	22.50
FRACTIONS MULTPLY/DIV	7.95
GRANDMA'S HOUSE (-10)	9.95
HEY DIDDLE (AGE 3-10).	9.95
LINEAR EQUATIONS	7.95
LINKWORD: SPANISH	22.50
LINKWORD: GERMAN	22.50
LINKWORD: FRENCH	22.50
LINKWORD: ITALIAN	22.50
MASTER TYPE	14.95
PRE-SAT WORD ATTACK ..	14.95
SAT PRACTICE TESTS ...	19.95
SNOOPER TROOPS CASE #1	9.95
SNOOPER TROOPS CASE #2	9.95
STATES AND CAPITALS ..	9.95
SPELLING BEE (AG 5-10)	14.95
TOUCH TYPING	9.95
QUIZ MASTER CONSTR....	8.95
QUIZ MSTR. USA CONSTR.	8.95
AMERICAN EDUCATION:	
A+ BIOLOGY G10+	17.95
A+ GERMAN	17.95
A+ GRAMMER G4+	17.95
A+ READING COMP G1-8	17.95
A+ SCIENCE G3-4	17.95
A+ SCIENCE G5-6	17.95
A+ SCIENCE G7-8	17.95
A+ SPANISH	17.95
A+ SPELLING G2-8 ...	35.95
A+ U.S. GOV. G10+ ..	17.95
A+ U.S. HIST. G5+ ..	17.95
A+ VOCABULARY G4+...	17.95
A+ WORLD GEOG. G8+ .	17.95
A+ WORLD HIST. G8+ .	17.95
(G = GRADE LEVEL)	
ATARI:	
CONCENTRATION	9.95
DIVISION DRILL	9.95
GRAPHING	9.95
INSTRUCTIONAL COMPUT	9.95
PLAYER MAKER	9.95
PREFIXES	9.95
SCREEN MAKER	9.95
SECRET FORMULA INTERM	9.95
SECRET FORMULA ADVANC	9.95
SPELL IN CONTEXT 8..	9.95
BUY 2, GET ONE FREE!!	
CBS (AGE 3-6):	
ASTROGROVER	8.95
BIG BIRD SPEC DELIVE	8.95
ERNIE'S MAGIC SHAPE.	8.95
DESIGNWARE:	
MATHMAZE (6-11)	26.95
MISSION ALGEBRA (13+)	13.50
SPELLICOPTER (6-11).	26.95
TINK TONK (AGE 4-6):	
ABC'S	8.95
COUNT AND ADD	8.95
SMART THINKER	8.95
SPELLING	8.95
SUBTRACTION	8.95
THINKING SKILLS	8.95
ALL 6 TINK TONKS..	39.95
UNICORN:	
10 LITTLE ROBOTS	
(PRE-SCHOOL)	26.95
FUN BUNCH (6-ADULT)	26.95
RACECAR RITHMETIC	
(AGE 6+)	26.95
WEEKLY READER (PRE-SCHOOL):	
STICKY BEAR SHAPES .	26.95
STICKY BEAR NUMBERS.	26.95
STICKY BEAR ABC'S ..	26.96
STICKY BEAR OPPOSITE	26.95
SB BASKET BOUNCE ...	26.95
STICKY BEAR BOP	26.95
RUN FOR IT	26.95
PIC BUILDER	26.95

800/XL/XE SOFTWARE
ALL TITLES ON CARTRIDGE

ENTERTAINMENT

3D TIC-TAC-TOE	9.95
AIRBALL (XL/XE)	24.95
ALIEN AMBUSH	9.95
ACE OF ACES .(XL/XE) .	24.95
ARCHON	14.95
ASTEROIDS	15.95
ASTRO CHASE	14.95
ATARI TENNIS	9.95
ATLANTIS	14.95
BALL BLAZER	14.95
BARNYARD BLASTER	24.95*
BASKETBALL	14.95
BATTLEZONE	19.95
BLUE MAX	19.95
BOULDERS & BOMBS	14.95
CAVERNS OF MARS	14.95
CENTIPEDE	14.95
CHICKEN	7.00
CHOPLIFTER	19.95
CLAIM JUMPER (400/800)	9.95
CLOUDBURST	9.95
CRIME BUSTER	24.95*
CROSSBOW	24.95*
CROSSFIRE	9.95
CRYSTAL CASTLES(XL/XE)	19.95
DARK CHAMBERS (XL/XE).	24.95
DAVIDS MIDNIGHT MAGIC	19.95
DEFENDER	9.95
DESERT FALCON	14.95
DIG DUG	19.95
DONKEY KONG	10.00
CART ONLY	5.00
DONKEY KONG JR.	19.95
EASTERN FRONT (1941).	19.95
E.T. PHONE HOME	9.95
FIGHT NIGHT	19.95
FINAL LEGACY	19.95
FOOD FIGHT (XL/XE)...	19.95
FOOTBALL	14.95
FROGGER	14.95
GALAXIAN	9.95
GATO	24.95
GORF (400/800)	9.95
HARDBALL	19.95
INTO EAGLES NEST(XL/XE)	19.95
JOUST	19.95
JUNGLE HUNT	12.95
KABOOM!	14.95
KARATEKA	19.95
KRAZY ANTICS	14.95
LODE RUNNER	24.95
MARIO BROS (XL/XE)...	19.95
MEGAMANIA	9.95
MILLIPEDE	9.95
MISSILE COMMAND	10.00
MOON PATROL	19.95
MR. COOL	9.95
MS. PAC MAN	19.95
NECROMANCER	19.95
ONE ON ONE (XL/XE)...	19.95
PAC MAN (CART ONLY)..	5.00
PENGO	19.95
PLATTERMANIA	9.95
POLE POSITION	19.95
POPEYE	14.95
Q-BERT	14.95
QIX	9.95
RESCUE ON FRACTALAS .	19.95
RETURN OF THE JEDI ..	14.95
ROBOTRON:2084	14.95
SLIME (400/800)......	7.00
SPRINGER	7.95
SPACE INVADERS	14.95
STAR RAIDERS	5.00
STAR RAIDERS II	19.95
SUBMARINEE COMMANDER	14.95
SUMMER GAMES (XL/XE).	24.95
SUPER BREAKOUT	9.95
SUPER COBRA	14.95
THUNDERFOX	19.95
TURMOIL	9.95

PROGRAMMING

ACTION!	71.95
ACTION! TOOLKIT-DISK .	26.95
BASIC XL	53.95
BASIC XL TOOLKIT-DISK.	26.95
BASIC XE	71.95
DIAMOND (GEM O/S)	69.95
LOGO	29.95
MAC/65	71.95
MAC/65 TOOLKIT-DISK...	26.95
PILOT	19.95
SPARTA DOS X	71.95

PRODUCTIVITY

ATARIWRITER	29.95
ATARIWRITER (CART ONLY)	19.95
FUN WITH ART	14.95
MICROFILER	22.50

EDUCATION

ATARI LIGHT MODULE	
(REQ. STARTER SET) .	9.95
MATH ENCOUNTERS	9.95
FISHER PRICE (PRE SCHOOL):	
DANCE FANTASY	8.95
LINKING LOGIC	8.95
LOGIC LEVELS	8.95
MEMORY MANOR	8.95
ALL 4 FOR ONLY	24.95
SPINNAKER (AGE 3-10):	
ALF IN COLOR CAVES .	9.95
ALPHABET ZOO	9.95
DELTA DRAWING	9.95
FACEMAKER	9.95
KIDS ON KEYS	9.95
KINDERCOMP	9.95
STORY MACHINE (XL/XE)	9.95
(AGE 7 - ADULT):	
ADV.CREATOR (400/800).	9.95
FRACTION FEVER	9.95
(* = REQ. LIGNT GUN)	

SPECIAL PRICE
ATARI
XE GAME MACHING
NOW ONLY
$99.95
LIMITED TIME ONLY

LIGHTGUN	34.95

SUPER SPECIALS
RECONDITIONED ATARI MERCHANDISE 30 DAY WARRANTY

800 (48K) COMPUTER $79.95 **1030 MODEM WITH EXPRESS! $24.95**	**SPACE AGE JOYSTICK $5.00.** **ATARI TRACKBALL $9.95**	**3-1/2" DISKS AS LOW AS 75 CENTS** 10 FOR $8.95 100 FOR $80 1000 FOR $750 **DOUBLE SIDED WITH OLD DEMO SOFTWARE**	**1020 COLOR PRINTER/PLOTTER $19.95** (new in box) 40 COLUMNS WIDE INC. PENS, PAPER, ETC.	**600XL COMPUTERS 64K - $64.95 16K - $49.95** **ATARI NUMERIC KEYPAD $7.95**	**5-1/4" DISKETTES AS LOW AS 20 CENTS** 10 FOR $4.00 100 FOR $29.95 1000 FOR $200 **MOST ARE UNNOTCHED WITH OLD SOFTWARE**

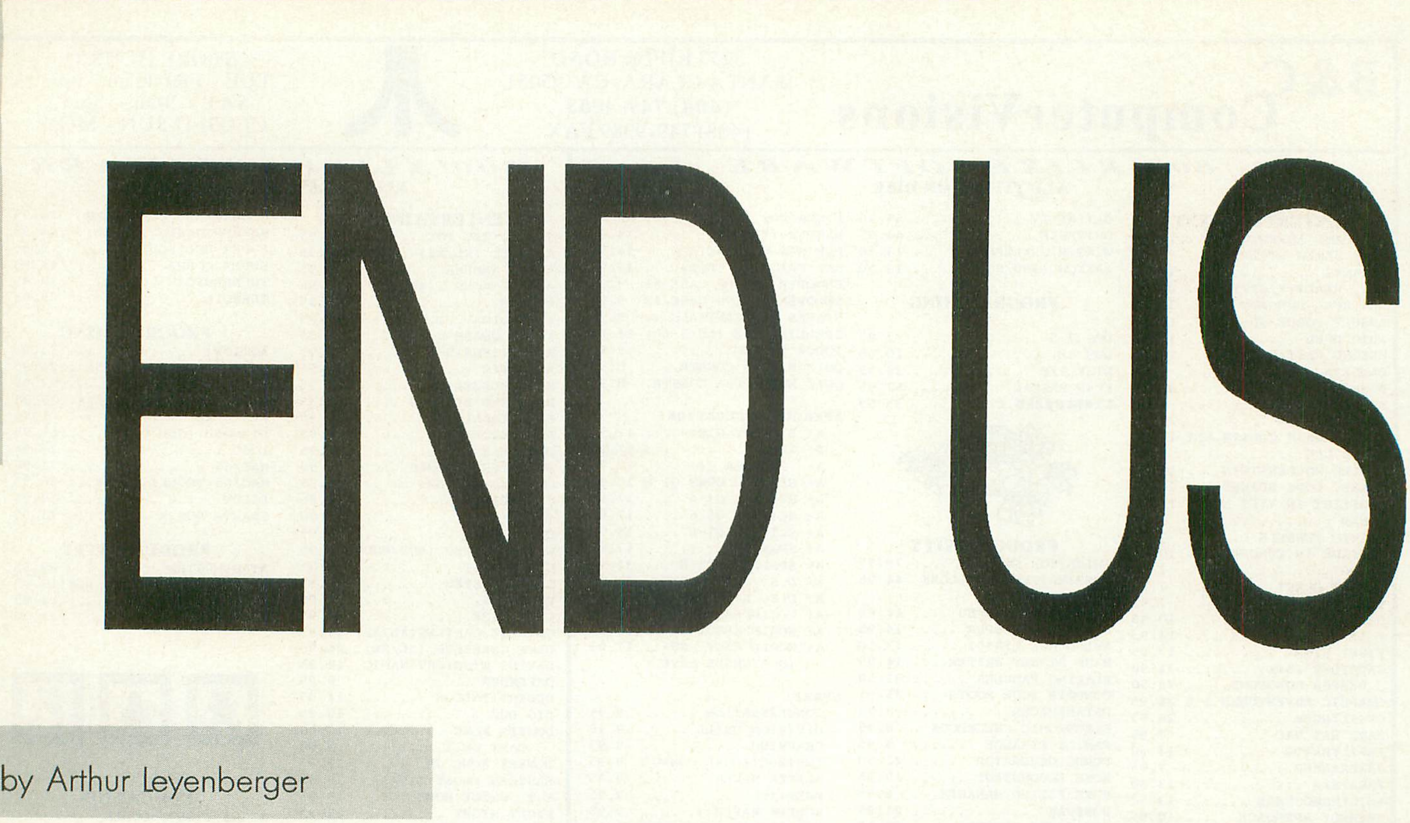
SHIPPING INFORMATION - Prices do not include shipping and handling. Add $5.00 for small items ($8.00 Min. for Canada). Add $8.00 for disk drive. Add $2.75 for C.O.D. Calif. res. include 7% sales tax. Mastercard and Visa accepted if your telephone is listed in your local phone directory. Orders may be pre-paid with money order, cashier check, or personal check. Personal checks are held for three weeks before order is processed. C.O.D orders are shipped via UPS and must be paid with cash, cashier check or money order. International and APO orders must be pre-paid with cashier check or money order. $20.00 minimum on all orders. All sales are final - no refunds - prices are subject to change.

Phone orders accepted **TUESDAY THROUGH FRIDAY** from 10:00 am to 6:00 pm PST.
We carry a full line of **ATARI** products - large public domain library - write or call for free catalogue

PRICES SUBJECT TO CHANGE WITHOUT NOTICE - ALL SALES ARE FINAL

CIRCLE #109 ON READER SERVICE CARD.

END US



by Arthur Leyenberger

I don't know. Maybe it's me. Regardless of how much I want Atari to be successful, something always happens with them that frustrates the heck out of me.

Maybe being an Atari user for so long has clouded my judgment. As an Atari user, the inner workings of Atari Corp. is really none of my business. Really. I'm also an IBM PC user. I don't get all excited when IBM does something I disagree with or something that appears to be stupid.

Atari appears to be getting their act together. They promised that 1989 would see a renewed effort on their part to promote and sell the ST in the United States. It may not actually happen until the first half of 1990. That's okay with me—at least they are trying. But in the midst of this effort, the revolving door of Atari management has opened and closed once again.

Assistant vice-president of sales Joe Mendolia is no longer with Atari. I don't know the details of his leaving. Whether he quit or was fired doesn't matter to me. It just indicates once again that the Tramiels are difficult to work for.

In my column in the October ST-LOG I mentioned the new User Group Coordinator, Chris Roberts. He had big plans for uniting the users' groups across the country via a special-interest group on a telecommunications service. Other plans included the publishing of a users' group newsletter by Atari to improve communications with Atari Corp. Now, before the ink has dried on the pages of ST-LOG, Chris is no longer with Atari Corp.

Atari has said many times that they are relying on users' groups to help spread the ST gospel. It's a two-way street. If they want the support of users' groups, they need to support the user groups.

Interface Wars: The Continuing Saga

I have written many times about the absurdity of the interface wars, otherwise known as the "look and feel" lawsuits that have plagued the microcomputer industry. It seems that Apple Computer has spent more money suing other companies with products that have allegedly infringed on the look and feel of the Macintosh Desktop than they have on research and development. You probably recall that Apple threatened to sue Digital Research because the GEM Desktop looked too much like the Mac Desktop. As a result, DR modified the GEM Desktop (for MS-DOS machines) so it now is less intuitive and more difficult to use. Fortunately, the GEM Desktop on the ST has not been affected.

Apple actually brought suit against Microsoft for their Windows product and Hewlett-Packard for their New Wave graphical user interface. Luckily, the courts put a stop to Apple's shenanigans. The court ruled that a prior agreement between Apple and Microsoft did indeed grant Microsoft broad rights to what Apple claims was theirs. In effect, the judge threw out the case.

Although Apple still has some sort of claim concerning icons and their manipulation and the concept of overlapping windows, this decision should put a stop to Apple's attempt to sue anybody with a graphical interface. Further, if this does happen, users will benefit from a more or less standard interface.

All graphical user interfaces have essentially the same elements: multiple windows, drop-down menus and icons. It's about time users need not have to relearn the basics of program operation. For example, in the MS-DOS world, the F1 function key is typically used to invoke a help function. This has become standard practice and makes programs easier to use. ST software using the GEM interface is made easier to use because program functions are accessed in a consistent manner. Although a user still has to learn the specific commands of a new program, the interface does not interfere with the learning process.

Perhaps the recent court decision against Apple Computers will send a signal to other companies that have been pursuing "look and feel" lawsuits. I hope it does. Programs are becoming more complex all the time, and anything that can help the user is much needed.

ER

Something Old, Something New

At the June 1985 Consumer Electronics Show, the major Atari excitement centered around the CD-ROM player for the ST. As you may recall, it was shown together with Groliers Encyclopedia, a CD-ROM application. Atari announced that both would be available by the end of the year. Most of us assumed that it was going to be the end of *that* year, but since the Atari CD-ROM player has yet to materialize, we can't be sure which year Atari had in mind.

Also shown over four years ago were several products for the then-recently introduced 65XE and 130XE computers. One product, meant exclusively for the 130XE computer, was a Lotus *1-2-3* clone called *The Professional*. Developed by VIP Software, it too was going to be available by the end of the year and sell for under $100. What was interesting about *The Professional* was that it used windowing and pull-down menus similar to those found on the ST.

In order for *The Professional* to operate as promised, a graphical interface was needed for the 8-bit computers. Sure enough, Atari was demonstrating one of those too. Appropriately (if unimaginatively) called the GEM Desktop, the new Atari software was a GEM lookalike program that was said to run on all 8-bit Atari computers. You guessed it, VIP Software was the developer. And, of course, the program was promised to be available by Christmas.

I don't mention these never-released 8-bit products to embarrass Atari. Instead, I want to show how the excitement over the ST (at the time) had carried over into the 8-bit world. It seemed feasible that, for the most part, what could be done on the ST could be duplicated on the 8-bit computer with a little imagination and a lot of programming acumen.

Atari never did deliver on their promise to develop and market a GEM-like interface for the 8-bit computers. However, within the last year, two companies were each working on similar graphical interface products. Total Control Systems' Graphic Operating Environment (GOE) and Reeve Software's Graphical Operating System (GOS) were both scheduled to be out by now, but to date, only Diamond GOS from Reeve Software (29W 150 Old Farm Lane, Warrenville, IL 60555; 312-393-2317) has appeared.

The Diamond GOS package consists of a supercartridge containing the GOS and the desktop software, a disk containing various utilities for configuring the Diamond environment, another disk with the *Diamond Paint* program and two manuals (Diamond GOS user manual and Diamond GOS programmer's manual). The cartridge is the piggyback type that allows you to insert another cartridge into the top of it. To use Diamond, you need a mouse (not included in the package). Although the product can work with joysticks, trackballs and touch tablets, the manual highly recommends the use of an ST mouse. So do I.

Diamond GOS is similar to the GEM Desktop. The initial configuration (which you can modify to suit your own system) consists of disk drive and trash can icons, one window in the center of the screen and four menu names (Desk, File, Disk and Options). With the mouse plugged into Port 1 (the second joystick port), you have complete control over the desktop by pointing, clicking and dragging.

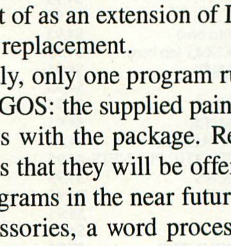
Only two windows can be opened at once on the desktop (compared to four on the ST). Interestingly, as you click on either of the two windows, the operating system rereads the disk in the selected drive. Pressing the escape key will update the window if you happen to change disks in the drive that has the active window.

Another interesting difference between GOS and the ST GEM desktop is what happens when you make a particular window active by clicking on it. The screen is repainted when you do this and the underlying window information is lost. Like the GEM desktop, once you have configured your GOS desktop with particular windows and drive icons, you can save the desktop to disk. That way, whenever you boot your system with GOS, your familiar desktop configuration will be displayed.

As this is not a full review of Diamond GOS (see the September '89 ANALOG Computing), I won't describe all the features of the program. However, it is significant that someone (Reeve Software) has been able to create a graphical user interface for the 8-bit Atari computer. Since Diamond GOS works with most 8-bit disk-operating systems—Atari DOS 2.5, Atari DOS XE, Sparta DOS X, etc.—it can be thought of as an extension of DOS rather than a replacement.

Currently, only one program runs under Diamond GOS: the supplied paint program that comes with the package. Reeve Software states that they will be offering additional programs in the near future, such as desk accessories, a word processor and a desktop-publishing program. If Diamond GOS catches on, perhaps other software suppliers will write programs to run under the system.

The cartridge version of Diamond GOS sells for $79.95. You'll have to add another $35 to the total price if you don't happen to have an ST mouse. There is no question that Diamond GOS brings a functional, modern graphical user interface to the 8-bit Atari. You'll have to decide if it is worth the price given the lack of additional software that is now available to work with it.



Arthur Leyenberger is a freelance writer who lives in beautiful New Jersey. He can be reached on CompuServe at 71266,46 or on Delphi as ARTL.

ATARIWRITER 80

- 80 COLUMN EDITING
- EDITABLE PRINTER DRIVER
- 30,000 WORD DICTIONARY
- MAIL MERGE
- ON 5¼ DISKETTE

$49

ATARI XEP 80

- CONNECTS TO ANY COMPOSITE MONITOR
- 80 COL. DISPLAY
- PRINTER INTERFACE

$79

ATARI 80 Col. PACKAGE

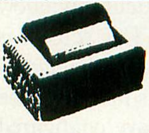
- 12" MONO MONITOR
- ATARI XEP 80
- ATARIWRITER 80

$179

EVEN BETTER W/ Printer

- 12" MONO MONITOR
- ATARIWRITER 80
- ATARI XEP 80
- 1025 PRINTER

$229



1020 COLOR PRINTER PLOTTER

NEW! $14.95

COMPLETE WITH:
- 2 PEN SETS
- 1 ROLL PAPER
- POWER SUPPLY & CABLE

EXTRA PEN SETS
COLOR $3.98 BLACK $.89

1200XL 64K COMPUTER

256 COLORS
XEGS COMPATIBLE
FREE PAC-MAN

$49 RECONDITIONED

$69 FACTORY NEW!!

1025 DOT MATRIX PRINTER

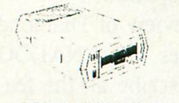
RECON-DITIONED $59

800 48K MEMORY COMPUTER

RECON-DITIONED $49

HARDWARE & ACCESSORIES

850 PRINTER INTERFACE	$79.95
JOYSTICK EXT. CABLES 10'	$2.95
ATARI TWO JOYSTICK SET	$9.95
SPACE AGE JOYSTICK	$14.95
WICO COMMAND JOYSTICK	$14.95
WICO THE BOSS JOYSTICK	$19.95
ATARI TRACKBALLS	$9.95
ATARI SX212 MODEM	$89.95
AVATEX 1200HC MODEM	$89.95
DOS 2.5 W/ MANUAL	$4.95
600XL (NO Transformer)	$19.95
400,800,850,1200XL Transforner	$14.95
XL/XE Transformer	$24.95



DISK DRIVES FOR 800/XL/XE

COMPLETE WITH:
POWER SUPPLY
I/O CABLE
DOS 2.5 W/
MANUAL

ATARI 810 $129.00
ATARI 1050 $169.00
RECONDITIONED

THE BEST LIGHT GUN

FOR YOUR 800/XL/XE

$35

OR THE PACKAGE
LIGHT GUN
CRIME BUSTERS
CROSSBOW
BARNYARD BLASTER
$109

THE NEWSROOM

Desktop Publisher By SPRINGBOARD

$15

CARTRIDGES FOR THE 800/XL/XE

BASIC CARTRIDGE	$4.95	DANCE FANTASY	$8.95	ROBOTRON	$19.95	BLUE MAX	$19.95
BASIC TUTOR (2 BOOKS)	$4.95	LOGIC LEVELS	$8.95	TENNIS	$19.95	STAR RAIDERS II	$19.95
QIX	$4.95	MEMORY MANOR	$8.95	FINAL LEGACY	$19.95	DAVID'S MIDNIGHT MAGIC	$19.95
TURMOIL	$4.95	LINKING LOGIC	$8.95	MARIO BROS	$19.95	ARCHON	$19.95
PAC-MAN (no box)	$4.95	DELTA DRAWING	$9.95	DONKEY KONG JR	$19.95	KARATEKA	$19.95
DONKEY KONG (no box)	$4.95	HEY DIDDLE DIDDLE	$9.95	JUNGLE HUNT	$19.95	CHOPLIFTER	$19.95
GORF (400,800)	$4.95	GRANDMA'S HOUSE	$9.95	MOON PATROL	$19.95	GATO	$24.95
CHICKEN	$4.95	FRACTION FEVER	$9.95	BATTLEZONE	$19.95	ACE OF ACES	$24.95
SLIME (400,800)	$4.95	ALPHABET ZOO	$9.95	FOOD FIGHT	$19.95	LODE RUNNER	$24.95
CLAIM JUMPER	$4.95	ALF	$9.95	HARDBALL	$19.95	BARNYARD BLASTER (LG)	$24.95
DELUXE INVADERS	$4.95	ASTRO CHASE	$14.95	FIGHT NIGHT	$19.95	DARK CHAMBERS	$29.95
JOURNEY TO THE PLANETS	$4.95	FROGGER	$14.95	ONE ON ONE BASKETBALL	$19.95	AIRBALL	$29.95
STAR RAIDERS	$4.95	DIG DUG	$14.95	DESERT FALCON	$19.95	SUMMER GAMES	$29.95
DEFENDER	$4.95	MILLIPEDE	$14.95	NECROMANCER	$19.95	CROSSBOW (LG)	$29.95
ET	$4.95	SKY WRITER	$14.95	RESCUE ON FRACTALUS	$19.95	EAGLES NEST	$29.95
FACEMAKER	$4.95	FOOTBALL	$14.95	BALLBLAZER	$19.95	CRIME BUSTERS (LG)	$29.95
MATH ENCOUNTER	$7.95					MICROFILER (database)	$39.95

DISK SOFTWARE FOR THE 800/XL/XE

DAVID'S MIDNIGHT MAGIC	$4.95	NATO COMMANDER	$4.95	SECRET MISSION	$4.95	SILICON DREAMS(1050)	$9.95
SPIDERMAN	$4.95	CROSSCHECK	$4.95	VOODOO CASTLE	$4.95	JEWELS of DARKNESS(1050)	$9.95
CRYSTAL RAIDERS	$4.95	MISSION ASTEROID	$4.95	STRANGE ODYSSEY	$4.95	SILENT SERVICE	$9.95
REPTON	$4.95	COMMBAT	$4.95	REPTON	$4.95	DISPATCH RIDER	$9.95
BANDITS (48K 400,800)	$4.95	PREPPIE I	$4.95	HULK	$4.95	SILICON DREAMS	$9.95
CLAIM JUMPER	$4.95	THE COUNT	$4.95	ALIEN AMBUSH	$4.95	F-15 STRIKE EAGLE	$9.95
DECISION IN THE DESERT	$4.95	FREAKY FACTORY	$4.95	JAWBREAKER	$7.95	THE NEWSROOM(1050-64K)	$15.00
CRUSADE IN EUROPE	$4.95	LASER HAWK	$4.95	SPY VS SPY	$9.95	BOOKKEEPER W/ num keypad	$29.95
KENNEDY APPROACH	$4.95	CRYSTAL RAIDERS	$4.95	TOP GUNNER COLLECTION (3 GAMES)	$9.95	HOME ACCOUTANT	$29.95
CONFLICT IN VIETNAM	$4.95	ADVENTURELAND	$4.95				

SPECIAL DEALER PRICES AVAILABLE ON MANY TITLES!!

NEW ARRIVALS

DECISION IN THE DESERT	$4.95
CRUSADE IN EUROPE	$4.95
KENNEDY APPROACH	$4.95
CONFLICT IN VIETNAM	$4.95
NATO COMMANDER	$4.95
F-15 STRIKE EAGLE	$9.95
SILENT SERVICE	$9.95
SPY VS SPY	$9.95

WICO COMMAND CONTROL

SALE

ONLY $14.95

Reg. $29.95

COMPOSITE MONO MONITORS

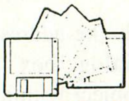
$59

LIGHT SPEED C DEVELOPMENT KIT

$35.95

FLAT RATE $5.00 SHIPPING

ON ALL GROUND ORDERS IN THE CONTINENTAL U.S. *



SONY DISK SPECIAL

3.5 BULK PACKS WITH LABELS

1 - $.99 25 - $24.75 1000 - $750.00

ST COLOR MONITORS $299

3 TO CHOOSE FROM!

SM124 MONO MONITORS $149

$5 SHIPPING

MEMORY UPGRADES

TECH SPECIALTIES	2/4Meg	$149.95
NEW WORLD UPGRADE	2/4Meg	$139.95
ARCO BOARD	1/2/4Meg	$169.95
256K/1Meg CHIPS		$CALL

ST TURBO SPEED!!

J.A.T.O. BOARD 8/16Mhz	$99
CMI PROCESSOR ACCELERATOR	$299
Math Co-processor for CMI 12Mhz	$99

STAR PRINTERS

STAR NX1000	9 Pin	$199
STAR NX1000 Rainbow	9 Pin	$299
STAR NX2400 4 Fonts	24 Pin	$399

1200XL PACKAGE $99

- 64K 8-bit COMPUTER
- 1020 COLOR PRINTER PLOTTER
- 1025 DOT MATRIX PRINTER (80 COL)
- PAC-MAN CARTRIDGE
- BASIC WITH TUTOR SET

Give the kids something else to play with besides your ST!

HIT DISK PACKAGE ST

- SLAGON
- GOLDRUNNER
- JUPITER PROBE
- KARATE KID PART II

$29.95

FREE! LIMITED OFFER!

$800 OF FREE SOFTWARE!

WITH PURCHASE OF ANY NEW ST!

YOU GET: Knight Orc, Tracker,*Advanced Art Studio, Guild of Theives, The Pawn, Golden Path, U.M.S., Jinxter, Sentry, Starglider, Hit Disk, Goldrunner II, The Grail, Invasion, Soccer, Zero Gravity, Tetra Quest, Paint Pro, Datatrieve, Neocrome, Planetarium, Magical Math, Spelling Bee

520ST W/SSDD Drive	(USED)	$299
520STFM W/DSDD Drive	(NEW)	$499
1040STFM W/DSDD Drive	(NEW)	$699
Mega 2ST Computer	(NEW)	$1199
Mega 4ST Computer	(NEW)	$1799
JRI GENLOCK SYSTEM		$650
ST/IME MODULE	$5 SHIPPING	
Battery Backed up Clock		$59
IMG SCAN Printer Scanner		$89
RC AEROCHOPPER		
RC Flight Simul. w/Futaba control		$199
WICO COMMAND JOYSTICKS		$14.95

*Works only with TOS 1.0
We reserve the right to make exchanges for out of stock items

ST HARD DRIVE SAVINGS

ATARI MEGAFILE 30	$529
AWESOME 60+ (60Meg 28ms)	$799
AWESOME 80+ (80Meg 28ms)	$999

PCBoard Designer $49

Reg. $199.95

THE BEST MOUSE

$5 SHIPPING

REPLACEMENT MOUSE FOR THE ST!

$49

RAINBIRD TITLES ST

Golden Path	$4.95
Knight Orc	$9.95
Tracker	$9.95
Advanced Art Studio	$9.95
Guild of Thieves	$9.95
The Pawn	$9.95
Jinxter	$9.95
Sentry	$9.95
StarGlider	$9.95
Universal Military Simulator	$14.95
Carrier Command	$29.95

ALL ITEMS REG. $39.95 - $49.95

SPECIAL SOFTWARE SALE ST

HIT DISK (4 games)	Reg	$59.95	NOW!	$29.95
GOLDRUNNER II	Reg	$39.95	NOW!	$19.95
GRAIL	Reg	$19.95	NOW!	$12.95
INVASION	Reg	$29.95	NOW!	$19.95
SOCCER	Reg	$39.95	NOW!	$19.95
ZERO GRAVITY	Reg	$29.95	NOW!	$19.95
TETRA QUEST	Reg	$29.95	NOW!	$19.95
PAINT PRO	Reg	$49.95	NOW!	$19.95
DATA RETRIEVE	Reg	$49.95	NOW!	$19.95
H & D BASE	Reg	$99.95	NOW!	$19.95
Atari PLANARIUM	Reg	$29.95	NOW!	$19.95

MAC EMULATOR FOR YOUR ST

MAGIC SAC PLUS w/Roms	$79.95
SPECTRE 128 V1.9	$169.95
MAC ROMS 64K	$19.95
MAC ROMS 128K	$129.95

SAN JOSE COMPUTER

THE ATARI STORE

640 BLOSSOM HILL RD. SAN JOSE, CA 95123
STORE (408) 224-8575 • FAX (408) 224-8574

* SHIPPING: ADD $5.00 TO ALL ORDERS. AIR AND INTERNATIONAL SHIPPING EXTRA. THAT'S IT.

WARRANTY: 90 DAY WARRANTY ON ALL ITEMS. TAX: CALIFORNIA RESIDENTS ADD 7% SALES TAX.

PREPAYMENT: USE VISA, MASTERCARD, MONEY ORDER, CASHIER'S CHECK OR PERSONAL CHECK. PERSONAL CHECK MUST CLEAR PRIOR TO SHIPMENT. C.O.D.: CASH, CASHIER'S CHECK OR M.O. ONLY.

Prices subject to change without notice.

Brand and/or product names are trademarks or registered trademarks of their respective holders.

Ad produced on an ATARI ST using CALAMUS and printed on an ATARI SLM804 PostScript compatible laser printer.

CIRCLE #110 ON READER SERVICE CARD.

C-manship: A Complete GEM Application, Part 5

by Clayton Walnum

Last time, we added the code needed to create a new *MicroCheck ST* account. Unfortunately, once the account was created, we still weren't able to open it. This month we'll add the program segment that'll not only handle that task but also will enable us to modify the date shown in the date information box at the bottom of the *MicroCheck ST* screen.

Listing 1 is the new source code for this month. You should merge it with the combined source code from the previous months and delete the *open_acct()*, *do_new_mnth()*, *save_month()* and *get_new_date()* stubs from the previous portion.

Now compile the program and run it. Start a new account. After that procedure is complete, a dialog box will appear, asking for the month you want to work on. Select the month. The account will be opened and the information boxes on the screen will be updated for that account.

Now choose the New Date option of the Utilities drop-down menu. Another dialog will appear. Type in a new date. When you select the dialog's OK button, the date you have typed will appear in the date-information box at the bottom of the screen.

Let's take a look at the new functions.

Function *open_acct()*

This function is called whenever the user wants to open an account or has just finished creating one. It gets as input a pointer to the filename of the account the user wants to open. At the beginning of the function, it attempts to open the .MCK file for the account. If it fails, an alert box warns the user, and no further processing is done.

If the *fopen()* call is successful, we read in the information that's stored in the file. The file format is shown below. All of the information is in character format except the account balance, which is a long integer:

```
Bytes 1-26 Name
Bytes 27-52 Street Address
Bytes 53-68 City
Bytes 69-78 Unused
Bytes 79-81 State
Bytes 92-95 Account balance (long int)
```

As the data is read in, it's formatted the way it will appear in the check-entry dialog box. After reading all the data, we close the file and plug the pointers to the name and address string into the *ob_spec* for each of the appropriate fields in the check-entry dialog box.

```
check_addr[CHKNAME]. ob_spec = chkname;
check_addr[CHKSTREET]. ob_spec = chkstreet;
check_addr[CHKNAME]. ob_spec = chkcity;
```

In the above, *check_addr* is the address of the check-entry dialog box; CHKNAME, CHKSTREET and CHKCITY are the names of string objects inside the dialog box; and *ob_spec* is the pointer to the string to be displayed for that object.

After setting the dialog-box strings, we call *do_new_mnth()*, which gets the month selection from the user and calls the functions necessary to actually open the files. If the account gets opened okay, the flag *loaded* will be TRUE, and we'll call *set_menu_entries()* in order to enable and disable the appropriate entries in the drop-down menu.

Function *do_new_mnth()*

Here, we first set the title string of the month-selection dialog box to "NEW MONTH" by placing a pointer to the string (*newm*) into the object's *ob_spec* field:

```
cancdial_addr[CANCSTRG]. ob_spec = newm;
```

Here, *cancdial_addr* is the address of the month-selection dialog box and CANCSTRG is a string object within the dialog box.

The integer value *choice*, the button on which the user clicked to exit the month-selection dialog box, is returned from a call to *get_month()*, the function that handles the dialog itself. If the user exited with the OK button, we save the current month's data if it needs to be (*saved* equals FALSE) and call *open_new_month()* to open the files.

Function *save_month()*

In this function we first take the filename of the file to save (the pointer to which is passed into the function as *file*) and change the extension to "BAK." We then delete any backup file that may already exist for that month and rename the old data file as the new backup file. We then open a new file with the filename pointed to by *file* (warning the user with an alert box if we get an error), after which we write that month's data out to the file.

The first two bytes written are the number of transactions in the file (in integer form). Then, using a *for* loop, we call *save_check()* for each check record in the check structure, writing the data to disk, after which we close the file.

Now all we have to do is save the new account balance. Our call to *fseek()* moves the file pointer 91 bytes from the beginning of the file, which is where the balance is stored. We save the balance and close the file.

Function *open_new_month()*

The first task here is to discover which month the user selected from the month-selection dialog box. We do that by using a *for* loop to scan through each of the button objects in the dialog to see which one is selected. (Note that this technique will work only if the button objects were created in numerical order when the dialog was first designed.) Based on which button was selected, we set the integer *mnth* equal to a number from 0 to 12. The value 0 represents the Month 0 file, with the values 1 through 12 representing January through

December, respectively. All that's left now is a call to *open_month()* to read in and process the data for the new month selected.

Function *open_month()*

Since we're now opening a new file, we set the flag *saved* to TRUE. This flag will remain TRUE until we modify the data somehow. Next, we initialize some variables, then construct the filename for the month we'll be opening.

After opening the file, if we find that the transaction count is zero (by reading the first two bytes from the file), we ask the user if he'd like to start a new month. We have to do this because if the user has transactions entered into his .AUT file (automatic transactions), they will be added automatically to this month's file when it's opened. This gives the user a chance to change his mind before the transactions are entered.

If the user chooses to open the file, we call *load_auto()* to load any automatic transactions. If the user chooses not to open the file, we set everything back the way it was and exit the function.

Assuming we've opened the file, the flag *do_it* will be TRUE, so we clear the window, set the *loaded* flag to TRUE, set up some strings for the display and store the current month into *month*. Using a *while* loop, we read in all the checks from the file, keeping a count on the number of deposits and the number of checks as we do. Finally, we initialize some strings and variables, copy the account name and the string ":Edit Mode" into the window's title bar, close the file and vamoose.

Summing It Up

The rest of the functions presented this month, though they have important roles in the workings of *MicroCheck ST,* do not really need much discussion. Most of the programming theory used in them has already been covered, so I'll give you only a quick rundown on what they do:

load_auto() loads any automatic transactions that may be in the user's .AUT file.

save_check() saves the data for a check to disk.

read_check() reads the data for a check from the disk.

clear_window() blanks out the program's window with a white rectangle.

get_month() brings up the month-selection dialog box and retrieves the user's choice.

get_new_date() allows the user to change the program's displayed date via a dialog box.

chk_date() simply makes sure the date dialog box in *get_new_date()* was filled in correctly by the user.

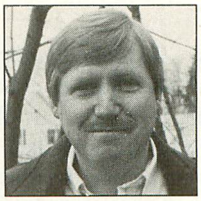
updte_buttons() places new data in the information boxes on the bottom of the screen.

And Now, the Great Cop-Out

It's become apparent to me that to continue on through the entire source code for *MicroCheck ST* would be forcing you to sit through a lot of boring repetition. The fact is that, in the code we've covered so far, we've looked at all the major topics I wanted to discuss and seen how they work in a full-scale program. I say, "Enough is enough!"

So we're calling it quits. On this month's disk you will find not only the code for the functions we discussed here, but the entire source file for *MicroCheck ST.* Study it if you're really interested in all the minor details.

In the next *C-manship,* we'll find a brand-new topic for discussion. (And no, at this point I haven't the vaguest idea what it will be.)



Clayton Walnum is the Executive Editor of ANALOG Computing *as well as the Associate Editor of* VIDEOGAMES & COMPUTER ENTERTAINMENT.

ATTENTION PROGRAMMERS!

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

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

Send your programs and articles to:
ANALOG COMPUTING
P.O. BOX 1413-M.O., Manchester, CT 06040-1413

Personal Publisher:

A Holiday Overview of DTP-Related Products

by Donavan Vicha

The past few *Personal Publisher* columns have been hard-core how-to tips-and-tricks-type articles. While I have hardly scratched the surface of what can be done using the Atari 16-bit line of computers for personal publishing, I'd like to take a break. Heck, the holidays are just around the corner, and it's time to work on wish lists for Santa. I've been writing about the Atari ST for more than four years now, and most of my wishes have come true by getting an editor or a publisher to send me review copies of their software (and in a few cases, their hardware). I'm in a pretty good position to play Santa's helper in the area of ST electronic publishing, so I'd like to make some recommendations as well as belatedly introduce myself to ANALOG Computing readers.

I caught the computer bug in November 1985 when we purchased a 520 ST for my wife to use as a word processor. We bought a color monitor because we knew our children would eventually want to play with the computer (not to mention me, the big kid in our home). We also purchased an SG-10 dot-matrix printer at the same time. By the summer of 1986, we had upgraded the 520 to one megabyte of RAM. I was using *The FinalWord* for word processing, while my wife Jessie had switched from *1st Word* to *ST Writer 1.7*, and I approached the now-defunct *ST Applications* with the idea of writing software reviews.

I worked as senior production editor for Contemporary Books, the largest trade book publisher outside New York City, and was very interested in desktop publishing. I saw no reason why the ST wasn't the perfect platform to learn electronic layout and design. Unfortunately, there wasn't much available for the machine at the time. But the next three years saw a lot of catching up by both Atari and third-party developers.

In April 1988, I quit my job and began floundering about as a freelance editor/designer. And now I'm about to take a job with a big financial services publisher as manager of composition services and will be working on getting the company into desktop publishing. Our home office now boasts our original one-megabyte 520 with color monitor hooked up to the SG-10, Supra 2400 modem and an SH204 20-megabyte hard drive; a Mega 2 with mono monitor hooked up to an Astra drive, a Blaser Star HPLJII-compatible laser printer and an NEC LC-890 PostScript laser printer.

So what do *I* wish for from Santa? A Mega ST4. An SLM804 with *UltraScript* and all of its fonts. A SyQuest 44 removable-cartridge hard disk. And at the very top of an impossible list, a 19′ Moniterm Viking monitor.

If your wish list looks like mine, this column will probably be of little interest. But if you've got your sights on more reasonable gifts, the suggestions below should help you tailor your list so that it includes exactly those products that fit your needs.

The High End

At the very high end are products much like those on my own wish list. I'll mention only one here. I've seen the SLM804 in action, and while I'm warned its installation process is complicated, I still want it over the two excellent laser printers I have. 'Nuff said, right? With *UltraScript*, the Atari laser is a killer hardware package. The lack of onboard RAM, rather than being a disadvantage, makes for superior speed in output, which is essential for professional publishing projects. It may also be possible, someday, to convert the laser for higher dpi output (the LaserMaster card for HP lasers, boosting output to 600x400 dpi, is proof that it is possible), so let no one tell you the Atari laser printer will become obsolete in a few years.

There are four DTP packages currently available for the ST, five if you include *Publishing Partner*, which has been supremely upgraded to *PageStream*. The other three packages are *Timeworks Desktop Publisher ST*, *Calamus* and *Fleet Street Publisher 2.0*. I can recommend *Fleet Street* only if you already have the other three and a yen to complete your collection. *Fleet Street* has great potential with many excellent, albeit cumbersome, type-handling features; but it lacks the ability to print multiple-page documents and to flow text from one page to the next and back—two functions the other programs handle in *pro forma* fashion.

Let me state here that all of the DTP

packages have their strengths and weaknesses. You must realize that such is the nature of electronic layout and typesetting, that no software package is perfect. Even the top programs for the Macintosh and IBM suffer in one way or another. Nevertheless, *Fleet Street Publisher* is the weakest package in the lot due to the faults mentioned above.

PageStream is the must-have program for professional publishing and creative layout needs on the ST. With Version 1.6, *PageStream* handles design and layout superbly, with tagging, rotation of text or graphics, text wrapping and a growing library of excellent fonts. With its powerful toolbox and manipulation functions (duplication, macros, rotation and grouping), you don't really need a drawing program for creating graphics separately. It handles color printing, as well as printing on just about every conceivable printer, low-end to high-end. Memory management is not its greatest strength, but it is the least stingy program RAM-wise and in terms of disk space than the other programs, meaning you can get by with the minimum configuration of a 520 and a single-sided drive.

PageStream remains the most user-friendly interface when it comes to creatively using the multitude of powerful functions built into this program. That does not mean it is the simplest to use, but there is usually more than one way to achieve what you want to do. Also, unlike *Calamus* or *Fleet Street*, the interface does not break down into so many specialized modes that you become easily confused. *PageStream* is the only DTP program I can recommend to both classes of users, beginner and expert. While the learning curve for *PageStream* for new users may be longer than that of *Desktop Publisher*, it extends over a much larger number of powerful, creative functions than the Timeworks program.

Calamus is also powerful, but when compared to the size of *PageStream* in terms of code alone, I still have to ask, "Where's the beef?" (*PageStream* is half the size of *Calamus*.) *Calamus* falls short of *PageStream* in a couple of categories that may or may not be critical to you: It does not support color or PostScript, and, although *Calamus* handles such esoteric functions as chapter numbering and footnoting (*PageStream* does not), it seems needlessly complex in its user interface, which requires a variety of frame types with their own idiosyncratic uses.

Calamus is a high-end product for professional-level users rather than beginners with no training in traditional layout and design. *Calamus*'s CompuGraphic fonts are expensive, but that is offset by its excellent, albeit also expensive, font editor. Unfortunately, the present CompuGraphic fonts do not compare favorably with *PageStream* fonts, and certainly not with PostScript fonts. This may be changed in the future since certain parts of the fonts' codes have been left out for the sake of file size, a regrettable but fixable error on ISD's part.

Calamus Outline promises some spectacular drawing functions that will definitely bring advantages to the use of *Calamus*. To me, it remains to be seen where *Calamus* is going—it looks great, but it also requires a lot of RAM to run.

Thus, I would recommend *Desktop Publisher ST* as the backup system to *PageStream* and perhaps the first program of choice for anyone looking to get his or her feet wet in electronic publishing. It is the easiest DTP program to learn and handles the majority of the functions most people would require of a publishing system. Its strengths lie in its text-tagging system and its user-friendly interface. *Desktop Publisher* requires that you do things the way it dictates, but you can accomplish those things in a simple manner, unlike *Calamus* and *Fleet Street*. It lacks the creative bells and whistles of *PageStream* and is unfortunately tied to Atari's GDOS patch, thus installation of fonts other than those that come with the program is a burden. Another unfortunate tie-in flaw is that it does not recognize an older 520's RAM upgrade, so beware.

PageStream remains the most user-friendly interface when it comes to creatively using the multitude of powerful functions built into this program. That does not mean it is the simplest to use, but there is usually more than one way to achieve what you want to do.

Desktop Publisher does handle PostScript well, and if you have two megabytes of RAM, you have the equivalent fonts of a LaserWriter Plus laser printer. It has been bundled with Mega systems using *UltraScript*, so I assume it has been upgraded to print a PostScript file to disk; but as a registered owner of the program, I have not been made aware of any updates of or improvements on the program. No new fonts have come out except GDOS fonts, which can be obtained on the networks. While Timeworks has been a strong supporter of the ST, we will have to wait for IBM and Apple upgrades before seeing improvements (or fonts) for *Desktop Publisher ST.*

In the cause of making life with GDOS and *Desktop Publisher* easier, there are a number of stocking stuffers I will recommend later in this column.

Graphics and Scanners

Easy recommendation: *Easy Draw 3.0.* If you have the upgrade, then you have the *EZD Supercharger.* Do you have *Easy Tools*? I described this accessory two months ago, so I'll be brief: Get it! Got it? Then get *OSpooler*, which also works with other GDOS-dependent programs, allowing you to get back to work while part of your RAM handles printing. Be sure, however, that you have the necessary updated drivers for your particular printer, since the *OSpooler* package comes only with Epson 9-pin drivers.

Touch Up from Migraph is another easy recommendation, especially if you use a lot of scanned artwork. Again, I've recently described this package and its uses a couple of months ago. It is available with Migraph's Hand Scanner, which I have yet to see due to a local (Chicago) distributor blowing its shipping responsibilities, leaving red faces in Federal Way, Washington (Migraph's headquarters). I'm looking forward to this package and plan on a full-blown comparison of various scanners in a forthcoming column.

I really see no other drawing package as necessary for DTP work. *DEGAS Elite* and *NeoChrome* do not deliver the high resolution you need for laser printer output. *Touch Up* allows you to "upgrade" a wide variety of graphic formats to a resolution that matches your printer, so go with it.

While we're on the subject of scanners, the high end of the scale is probably beyond Santa's ability to deliver: Navarone's ST Scan in its flatbed version is excellent.

(CONTINUED ON PAGE 56)

IAN'S

BY IAN CHADWICK

Design sense is usually a virtue acquired through training, not one obtained from reading a DTP manual. In publishing of any sort, it is inextricably linked with typography—a discipline, a craft that requires dedication, study and commitment. Without study, can you tell how many points of leading would be optimal in a 19-pica-wide body of text with ten-point type?

This column is about desktop publishing (DTP) and the arts and crafts associated with it. Try this little quiz, and count one point for each correct answer:

1. Which part of a character is most significant for recognition: the right or left?

2. What is the range of type sizes considered most legible?

3. What's the optimum word length per line for readability?

4. Which is more legible for a body of text: serif or sans serif typefaces?

5. Does text in all uppercase speed up or slow down reading?

6. Which numbers are best for legibility: Old Style or Modern Roman?

7. What's the optimum type size for newspaper headlines?

8. Do paragraph indents affect reading speed? If so, how?

9. Does justification affect legibility? If so, how?

10. Which is better: two smaller columns of text across a page or one large column?

11. Which is faster to read: large text or small text?

12. What's the optimum stroke thickness of a character?

13. What's an uncial?

14. What's the difference between the ascent (or ascender) line and the capline?

15. What is a counter?

16. Indicate whether the following typefaces are serif or sans serif:

a. Bodoni

b. Times Roman

c. Univers

d. Zapf Book

e. Futura

(Count one-half point for each correct choice.)

17. An em space is:

a. the size of a capital M.

b. a square unit of measurement, each side equal to the point size of the font.

c. one-quarter inch.

d. a measurement that relates the height of a font to the width of the first character in the paragraph.

18. Point size is measured from:

a. the descent line to the ascent line.

b. the baseline to the capline.

c. the baseline to the x-height.

d. the ascent line to the baseline.

19. There are:

a. 32 picas to an inch, six points to a pica.

b. 12 points to an inch, picas vary according to typeface.

c. 12 points to a pica, six picas to an inch.

d. one pica = .125 inch, points vary according to typeface.

20. A ligature is:

a. the spring device that holds metal type in place.

b. the rocker arm assembly used to press the plate onto the page for printing.

c. two or more characters designed as a distinct unit.

d. a device to separate lines of type to make them more legible.

Answers: 1. The right. The upper half is also more significant than the lower half.

2. Nine to 12 points, depending on the x-height. Generally, the smallest type size used is six point, although classified ads often use 5½ point.

3. Ten to 12 words, or 18 to 24 picas.

4. Serif. The serifs "lead" the eye forward. Sans serif fonts are best for headlines.

5. Uppercase slows down reading speed roughly 13%, from 5.38 words per second to 4.74.

6. Old Style, with differing descenders, also called "unaligned" numbers because they don't all lie on the baseline.

7. Fourteen to 30 points.

8. Yes: indentations of two to three ems increase reading speed by making it easier for the eye to locate the start.

9. No significant difference. Unjustified ("rag right") text is preferred because it's easier to correct and uses fewer hyphens.

10. Two. One column of 32 picas takes longer to read than two columns of 17 picas each.

11. Smaller. Larger text increases the number of fixations and forces the reader to perceive words in sections, rather than as a whole. Too small makes it hard to read, however.

12. Eighteen percent of the character height or width.

13. A modification of uppercase (capital) letters in which the sharp corners, vertices and edges are replaced with curves. This was developed because the style was more suited to writing on soft material, such as papyrus.

14. In many fonts, ascenders actually reach above the height of the capital letters, hence the capline—the maximum extent of the uppercase characters—is usually lower than the

QUEST

ascent line.

15. A counter is the fully or partially enclosed part of a character. The lines surrounding the counter are called the bowls.

16. a: serif, b: serif, c: sans serif, d: serif, e: sans serif

17. b. It was called an "em" space (or a "mutton" or "em quad") because the "M" fit in it. But it is not equal in size to a capital "M."

18. a. Point size is a vertical measurement of the body of a typeface.

19. c. There are 72 points to an inch. Okay, now what's agate? (Hint: It has to do with classified ads.)

20. c. Studies show ligatures also improve reading speed and legibility.

How did you do?

20+: You're a knowledgeable typographer, typesetter or graphic artist, probably already a professional in your field.

16-19: You have sufficient knowledge of typography to use DTP programs successfully, and you've obviously studied and applied yourself. You probably understand design concepts too.

10-15: You're an amateur—unpolished but with potential to be good at DTP. You need to study more. Stick to user-group newsletters until you can better your score.

6-10: You're either young or inexperienced, maybe both, and don't have enough skills to do very well at DTP. Avoid publishing anything on your own and never design fonts. Study, study, study.

0-5: You write in crayon but can't stay within the lines. You probably can't write legibly and you certainly can't use DTP within a mile of efficiency, let alone pleasantness. Stick to sheepherding or something less taxing.

Having the tool doesn't mean you have the talent. I have the usual plethora of power tools in my basement: saws, hammers, screwdrivers, planes. Although I can use them all reasonably well, I'm not a carpenter.

Desktop publishing is the same. It's a two-edged sword. On the one hand, it provides the ability for end users to design, create and print a lot of their own documents. This breaks the chains that have tied many companies to the traditional printers, especially in the area of forms production, where a considerable amount of DTP effort takes place.

Look at these figures, taken from a recent IBM press release:

* The Fortune 1,000 companies spend six to ten percent of their revenue on publishing.

* Forms represent 33% of all business documents.

* Between 20% and 30% of forms are thrown out due to regulation changes or obsolescence.

* Form-creation time takes an average of six weeks from design to production.

That makes you think, doesn't it? Forms seem, from the outside, like accountancy: dull, meticulous, drab. But look around you: We find forms even in our homes. Business and government run on forms, people think in terms of forms, you fill in forms to do almost everything. Forms production is a *big* business and an enormous source of revenue. It's also a major target of software publishers at the moment, who are beginning to realize that even a small share in the market means big bucks.

Add to forms such publications as newsletters, corporate reports, books, magazines, catalogues, brochures, menus and flyers. DTP offers not only the capability to design, but also escape from expensive paste-up and layout charges. And if limited quantities are required, these documents can even be print-

My unnamed friend suffers from the "instant expert" syndrome. He assumes that once you've got the software, you're an expert at the craft, that somehow all you need to know is the technical end of things and the rest will take care of itself. I worry that he's working on a font, but I lay awake at night knowing that he's doing a manual!

Business and government run on forms, people think in terms of forms, you fill in forms to do almost everything. Forms production is a *big* business and an enormous source of revenue. It's also a major target of software publishers at the moment, who are beginning to realize that even a small share in the market means big bucks.

ed internally at considerable savings and without the associated storage costs. It is certainly tempting to get into DTP, and many companies are doing so. Given the cost savings, it's no wonder.

If nothing else, think of the trees DTP saves!

The other side of the coin is the ease with which one can produce an incredibly ugly, badly designed, crowded, opaque and unreadable document. All the basic rules of design get ignored. And it shows.

Some rare human beings may have natural skill. They are diamonds in the rough, intuitively piecing together the text and graphic elements onto the page to create flawless, award-worthy design. Ninety-nine point nine percent of the other amateurs who attempt it merely make a mess of things.

Design sense is usually a virtue acquired through training, not one obtained from reading a DTP manual. In publishing of any sort, it is inextricably linked with typography—a discipline, a craft that requires dedication, study and commitment. Without study, can you tell how many points of leading would be optimal in a 19-pica-wide body of text with ten-point type? Without study, how the heck will you even know what a pica is? Or leading? Or the size of a point?

Along with DTP come the dreaded font editors, subtle termites that chew into the marrow of the art. It seems so easy to design a font. All it takes is a little bit of artistic talent, then you draw a few lines and make it look pretty, right? Wrong. Back as early as 1525, Albrecht Dürer published his *On the Just Shaping of Letters*, a technical treatise on the design of characters. It's a pretty tough little book; complex and demanding even by today's standards. Modern font design is even more so, despite the availability of tools such as the *Calamus Font Editor* (which is, stupidly enough, a desk accessory, proving once again that the Europeans don't always think things through before they publish . . .).

Current ST font editors make you construct a font one character at a time. That's a great approach for an adventure game; for typography, however, it's unsound. You need to construct characters in relation to other characters, to see not only the individual constructions but the effects of all of them together, how they look as words, sentences and paragraphs.

I sat looking over someone's shoulder recently and watched him design a font of his own for a manual he's working on. I asked him if it was going to be serif or sans serif.

"Serif," he answered smartly.

"What kind of serif? A cove serif? Or a square serif? How about a square cove serif? Thin-line serif? Exaggerated serif? Slab serif? Wedge serif? Triangle serif? Bracketed serif?"

"Huh?" he replied. "I think I'll make it sans serif instead."

"Will that be a square-normal-end sans serif? Or a square perpendicular end? Or a flared end? Maybe a rounded end?"

He huffed and puffed, but didn't answer.

"Will you use a standard midline, a constant midline or maybe a high standard midline with a pointed apex? Straight, bowed or concave arms? Horizontal, wedge or vertical openings? Gradual strokes with diagonal stress? Abrupt with vertical stress? Old Style, modern or condensed proportion? What's the relationship between stroke width and character height? Between descender and x-height? What about aligned numbers? Tapered or straight terminals?"

"Listen," he snarled, "I'm only working on the kerning lines right now. I'll deal with the details later."

"Do you know," I asked innocently, "what the term kern meant originally?"

He threw me out.

My unnamed friend suffers from the "instant expert" syndrome. He assumes that once you've got the software, you're an expert at the craft, that somehow all you need to know is the technical end of things and the rest will take care of itself. I worry that he's working on a font, but I lay awake at nights knowing that he's doing a manual!

Type design is a craft several centuries old. How can anyone expect to get involved in it without at least reading the basics? It's not merely a matter of creating a few good-looking characters. Sure, you can open any book of type and plagiarize like crazy (ignoring for the moment the copyright issue, since typefaces are mostly copyright and can't be copied). But how can you be sure the font is properly proportioned? If the thickness of the stroke improves or limits legibility?

But type is only a microcosm, the atoms of the DTP universe. What about the macrocosm: designing publications like books, magazines, even menus. Where will the design knowledge come from? Certainly not from the DTP programs themselves. *Calamus*, for example, is the best program of the lot but hasn't a whit of design direction in the docs (nor a bibliography or index).

Okay, so what's the point of all of this? Simply that owning the tool doesn't make you an adept with it. Neither does, by the way, knowing the commands. That only makes you technically able to run the program. There's no osmosis by which you'll absorb the necessary skills by merely learning to manipulate the command structure.

This fallacy is often reinforced by the software itself, which too often proves to be written by programmers who themselves lack the requisite typographic or typesetting skills. Here's a small test to illustrate my point: Try to replicate Dürer's alphabet using any current ST font editor. The font editors want to box everything into nice, neat frames with absolute limits. But Dürer (among others) wasn't designing for a computer, so he lets lots of strokes, tails and terminals escape the frame. If you start building the character set with *A*, you run into trouble when you reach *Q*, which has an unusually long tail—as long as other characters are wide.

The user quite often falls into these programs, unaware of the depth and complexity involved. It's like quicksand, trying to make a whole font conform to one character's outlines. The only source of direction is to be found in the manual. Oops, another problem.

MichTron's font editor doesn't even give you enough information to be able to use the program fully, let alone understand even the rudiments of type design. ISD's *Calamus Font Editor* suffers equally from fuzzy—albeit better—docs, which, among other ills, perpetuate the erroneous belief that the em-quad is the size of a capital "M." Necessary bibliographies of source references are notably missing from both programs, leaving me to assume that programmers without any background in the art created both programs. That's much like having a shoe salesman design airplanes.

The answer? I heartily recommend that you leave the DTP design work (including font construction) to those with the training, or that you go out and get the training yourself before you continue. There are many books on design and typography available. If you're doing this work without having read some of the more popular titles, then you're stumbling about in the dark. It doesn't get easier until you start reading.

Try to replicate Dürer's alphabet using any current ST font editor. The font editors want to box everything into nice, neat frames with absolute limits. But Dürer (among others) wasn't designing for a computer, so he lets lots of strokes, tails and terminals escape the frame.

Ian Chadwick is a Canadian freelance writer who also does volunteer work at the Toronto Humane Society. He and his wife share their small house with dogs, cats, ferrets and, at odd times, a time-traveling stenonychosaurus.

(CONTINUED FROM PAGE 51)

Hooked to an SLM804, you have the equivalent of a phcrocopy machine with 300 dpi capability when saving to disk. Halftones are very good if you have a hard disk for storing them. ST Scan allows you to make a full 8½"×11" page scan and then rubberband the part you want for your file—or do the whole thing! You don't have to do 300 dpi, and you shouldn't unless you plan on using a laser printer. Other settings are 75, 150 and 200. You can "print" to disk or send to printer (with the SLM, results are immediate and excellent).

On the lower end of the market, we are fortunate to have two good scanners that attach to the print head of your dot-matrix printer, transforming it into a scanning device. PictaScan from E. Arthur Brown is simple to set up and obtains good results in .PIx format only. IMG Scan is more difficult to set up and fine-tune but allows you to create .IMG files capable of handling a variety of dpi settings (with a max of 1,000 dpi!). IMG Scan has a lot of built-in fine-tuning of gray scales and contrast, but nothing approaching *Touch Up*'s editing abilities. You can use *Touch Up* with your IMG Scan file or get a public-domain program called AIM or a little-seen program called *Image Pro*, offered by Visual Solutions (last known address: P.O. Box 831132, Dallas, TX 75381-1332). Personally, I've had better luck with PictaScan, but I know several IMG Scan users who are happy with their high-resolution files.

Text Handlers

If you are serious about professional publishing with your ST system, *WordPerfect* is your only choice for a word processor. No other program boasts as many powerful functions as thesaurus, spell-checker, redlining, outlining, recordable macros, crash-proof automatic backup and formatted-file compatibility with the rest of the computer world. There are many books on using *WordPerfect* available, written mainly for IBM systems, that can be of great help in mastering its many power-user functions. To be able to take a client's IBM-based text file into your system will be an important asset to your business, and that alone should require *WordPerfect* in your library.

To me, *WordPerfect* is also the best program for plain old text handling. *ST Writer* is no longer burdened by its incorrigible text-file format, but it still hasn't mastered the best that GEM offers and still uses/displays its archaic control-key formatting commands. Both *1st Word Plus* and *Word Writer*, so much alike in functioning and features, handle block functions and reformatting poorly in comparison with *WordPerfect*. The price of *WordPerfect* keeps coming down, but cost should be your least concern. With its built-in macros, *WordPerfect* obtains optimum flexibility in its use, an advantage few other ST word processors have.

If the price really matters, you might consider *WordUp*—although it lacks speller and thesaurus functions—mainly because it too has a better grip on GEM and text handling for block functions and formatting. *WordUp 2.0*, unseen as yet, looks to have taken care of the few remaining annoyances in its functions and has added extra features that should make it an excellent low-end word processor with the added benefit of wrapping text around graphics.

If you need to generate form letters with business graphics (charts, tables and such), I recommend Atari's *Wordflair*, which I have only seen demonstrated, but the demo was impressive. *Wordflair* integrates the use of database and spreadsheet information with word processing. It uses GDOS fonts and allows you to see your entire page, much like a DTP program does. Here's a word processor of the third generation, handling text and graphics with the ease of a desktop-publishing program but retaining the simplicity of a word processor.

PictaScan from E. Arthur Brown is simple to set up and obtains good results in .PIx format only. IMG Scan is more difficult to set up and fine-tune but allows you to create .IMG files capable of handling a variety of dpi settings (with a max of 1,000 dpi!).

Stocking Stuffers

For those who plan on owning, or already have, GDOS-oriented programs, such as *Desktop Publisher ST*, *WordUp*, *Easy Draw* or *Touch Up*, there is one utility that you must have. *G+Plus* from CodeHead Software is undeniably the most necessary utility you can add to your setup. It replaces the GDOS patch with its own auto-program, and with its desk accessory installed and configured properly, you won't need to reboot in order to switch from one GDOS program to another. By allowing you to use your own names for your ASSIGN.SYS files, the accessory can be set up to know where to find the .SYS file for the program you want to run. The accessory also improves the performance of screen redraws of polylines and prevents the gradual slowdown of operation that GDOS programs seem to experience.

Utilities that also make life easier include several shareware programs: *Mouse Doubler*, written by Kyle Cordes, speeds up mouse movement, and *Mousetrap* prevents menus from dropping down unless you click the right mouse button. (Don't use *Mousetrap* with programs that use right mouse clicks—*PageStream* and the Migraph drawing programs, for example.)

For increased mousing speed, there are also two other alternatives: Best Electronics sells The Best Mouse, a sexy, round, two-button mouse, and Datel Products carries the Datel Mouse. Both are about twice as fast as the Atari mouse and cost about the same.

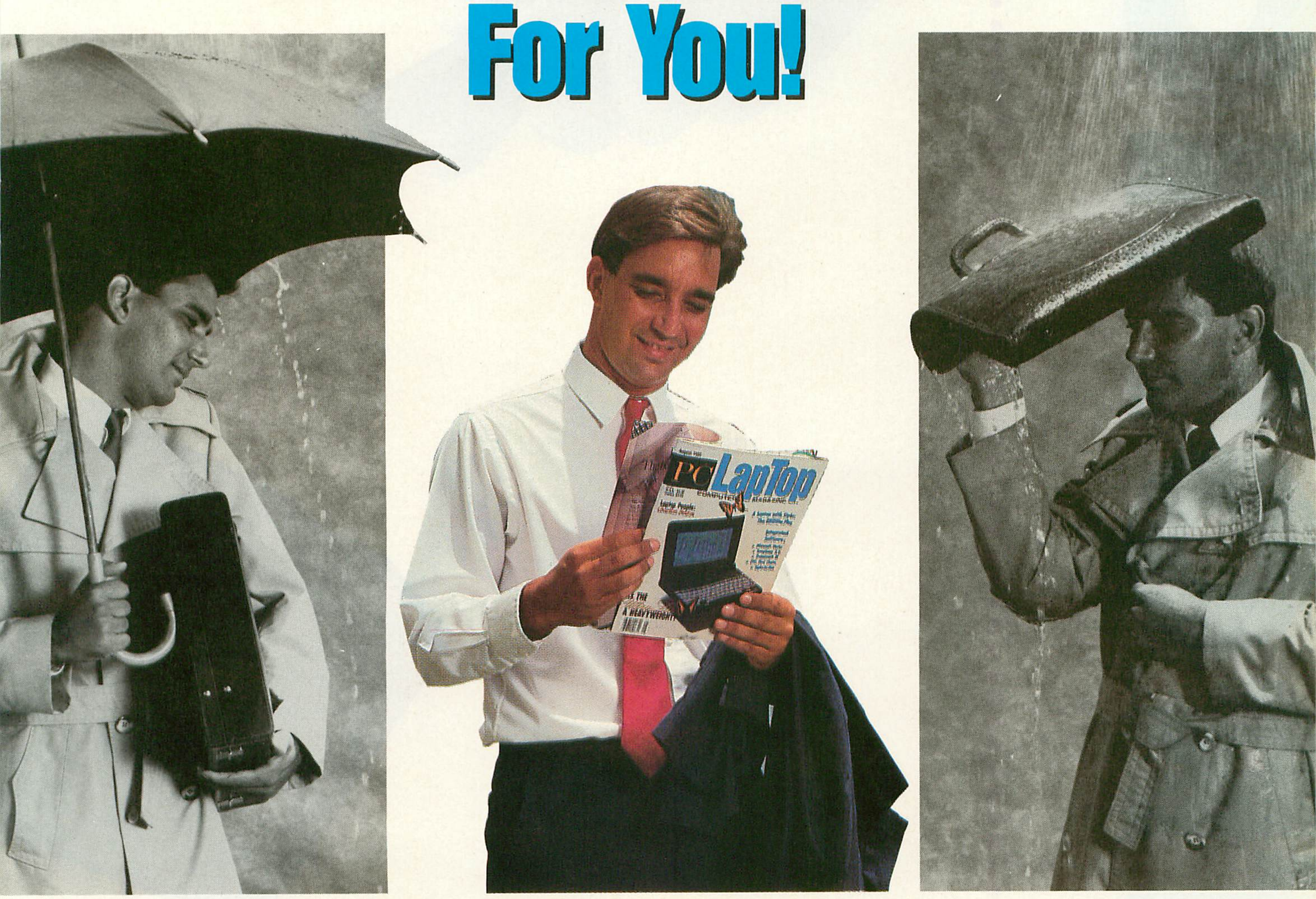
There are two more commercial utilities that anyone dealing with a hard drive and the kinds of programs personal publishing entails should consider absolute necessities: *Universal Item Selector II* (A&D Software), which replaces the current GEM Item Selector, bringing with it a multitude of convenience features (point-and-click selection of drive, multiple copy and move selections, reference of date and size of files, etc.) and *NeoDesk 2.0* (Gribnif), which replaces the GEM desktop, providing customized icons that can be placed on the desktop to allow running programs without digging into folders (and a control-key combo for closing directory windows several folders down).

John Brochu's *PicSwitch.07* is an excellent way to preview many art files (unfortunately, .IMG is not among those formats) that you can find on the public-domain disks or the networks. David Mumper's *TinyView* programs are also excellent art/graphic files management tools. If you register by paying a small shareware donation, you can get the latest updates: *TinyStuf* compresses a variety of formats into compact size and can be imported into *PageStream* in .TNY format, and *TinyView* is an excellent file viewer that can be scripted and can deliver cel-animation speed from a hard drive filled with .TNY pictures.

Finally, fonts and font editors fill an im-

(CONTINUED ON PAGE 112)

We Make It Clear For You!



NOW MONTHLY!

SUBSCRIBE TO PC LAPTOP
12 MONTHLY ISSUES ONLY $24.95
SAVE OVER $22 OFF THE COVER PRICE!

From reviews to interviews, features to applications, **PC LapTop** brings you all the information you need to get the most out of your laptop computer. And now, due to popular demand, you can get **PC LapTop** every month. So fill out the coupon below and get 12 monthly, easy-to-understand issues for only $24.95—you'll save over $22 off the cover price. Send in your order today!

YES! SIGN ME UP FOR 12 MONTHLY ISSUES OF **PC LapTop** FOR ONLY **$24.95.**
I'LL SAVE OVER $22 OFF THE COVER PRICE AND PAY JUST **$2.08** AN ISSUE

NAME ______________________________
ADDRESS ______________________________
CITY ____________________ STATE ________ ZIP __________
□ PAYMENT ENCLOSED □ BILL ME
□ CHARGE MY □ VISA □ MC # ____________________ EXP ______
SIGNATURE ______________________________

MONEY BACK ON UNMAILED ISSUES IF NOT SATISFIED! FOREIGN ADD $10. MAKE CHECKS PAYABLE TO LFP, INC. P.O. BOX 16927, N. HOLLYWOOD, CA 91615. YOUR FIRST ISSUE WILL ARRIVE IN 6 to 8 WEEKS. WATCH FOR IT! CJYYV

ELECTRONICS

WHAT'S NEW IN CONSUMER ELECTRONICS

by Arthur Leyenberger

We are living in a great age: the age of consumer electronics. Combine that with a great country that allows us the freedom and opportunity to benefit from technology in the areas of industry, medicine, education and most of all, leisure. Today, 98% of American households have television. Color TV is in 96% of our homes. Videocassette recorders can be found in 64% of our households.

These are startling numbers, but they show how pervasive consumer electronics is. However, consumer electronics is not a static thing. Technological improvements beget new products at an ever-increasing pace. The 1980s saw a tidal wave of new entertainment, educational and commercial products, all of which have had a profound impact on American lifestyles.

Think for a minute about the last decade. Compact discs have been around only for the past few years. We are now experiencing the second generation (and boom) of video games. Stereo TV-broadcasting is relatively new. Facsimile devices, telephone-answering machines, cordless phones, electronic musical instruments, cellular telephones, handheld reference products—the list goes on. Never in the 69-year history of consumer electronics has the quantity of new products, product categories and technologies been so great.

In 1988 the total retail value of all of the consumer electronics products sold in America was a staggering $43 billion. How-

ever, there is a great deal more to the story of the consumer electronics industry than just what it generates in annual sales. Its impact can be felt throughout the whole economy.

For every worker employed directly by manufacturers, distributors and retailers, there are others whose jobs are generated indirectly by the electronics industry. Truck drivers move products to market, construction workers build plants and install new equipment, and a host of manufacturing and service companies provide everything from cafeteria catering to cardboard shipping cartons.

It all seems to come together twice each year at the Consumer Electronics Show. Here, scores of retailers, manufacturers, corporate buyers, consultants and the press spend four long days examining what's new in consumer electronics. Sometimes there are major new product introductions, such as the compact disc. Other times, technology moves more slowly, with incremental improvements in existing products.

But there is always something new and exciting to see. The following, then, is what's new in consumer electronics.

Video Games

For the second time in as many shows, video games seemed to be the theme. I saw video games everywhere—from the impressive "Nintendo Village" to dozens of booths displaying new video games as well as video-game accessories. Yes, the Summer Consumer Electronics Show could have been called the Summer Video Games Show.

Several new video-game machines debuted at this year's show. New products were being shown by NEC, Sega, Nintendo and Atari. These games included two new 16-bit home units and two portable video-game machines.

NEC debuted the TurboGrafx-16, a new expandable home video game that uses a 16-bit graphics processor for improved graphics, faster action and more colors. In addition, the unit has a larger memory, which means more complex games can be designed for it. The increased RAM also allows more "point-of-view" displays, such as those found in flight simulators and perspective adventure games.

With the improved graphics of the TurboGrafx-16, programmers can design games with shaded figures, lines that are less jagged (with the use of anti-aliasing—the process of creating smooth lines and arcs) and much more detail. The game can also display up to 16 sprites or characters on the screen at once. Sprites are objects that move independently on the screen; more sprites mean more realistic graphics.

SONY HCP-C10 PLAIN PAPER COPIER

SONY TCM-27 LCD CASSETTE RECORDER

The TurboGrafx-16 also has improved sound capabilities. Sound effects and music are produced in stereo and, with the use of an optional adapter, the unit can be connected directly to an amplifier. The tinny, computerized sound of older video games is replaced by fuller, more complex sound on the new NEC video-game console.

The new NEC console uses software on credit card-sized ROM cards. The games are packaged in CD-style jewel boxes. Games will also be available on CD, which will require the use of the optional NEC TurboGrafx-CD player. The storage capacity of a CD game is equivalent to 2,000 game cards, which will allow for games with very complex game fields as well as outstanding sound. The CD player will also play standard audio CDs and is expected to sell for $399.

In addition to the optional CD player, other expansion accessories will be offered. The TurboBooster is a device that connects the video game directly to audio/video equipment, providing direct video and stereo audio. The TurboTap adapter handles up to five players simultaneously. A TurboStick controller (joystick) that features variable turbo-fire and slow-motion features will also be available.

The TurboGrafx-16 system comes with the game unit, one controller and the *Keith Courage in Alpha Zones* game. The controller, which will also be available separately, has a three-level fire control that expands its use. Retail price for the NEC TurboGrafx-16 video-game console will be $199, and it will be available by the time you read this.

Sega's new video-game console is called Genesis. It is said to be the first true arcade-quality video game on the market. Genesis is a video-game system that uses the same 16-bit technology as many coin-operated arcade games. Like the Atari ST, it uses a 68000 microprocessor as well as a custom graphics chip that drastically improves graphics, sound and overall game play.

The Genesis system consists of a console that attaches to any TV or monitor and an arcade-quality joypad. Features of the game include high-resolution graphics, the ability to display more than 500 colors and two simultaneous and independently scrolling game fields. In addition, the unit has stereo sound, which can be heard through headphones connected directly to the headphone jack or via an amplifier or receiver connected to the unit.

Among the software titles that will be available when the Genesis is released are such arcade classics as *Altered Beast*, *Space Harrier II* and *Super Thunder Blade*; action games, like *Ghouls 'n Ghosts* and *Thunder Force II*; and sports games, such as *Tommy Lasorda Baseball*, basketball, soccer and golf. With the optional powerbase converter, more than 80 existing Sega Master System 8-bit cartridges can be played on the unit. SegaScope 3-D glasses and 3-D games are also playable on the

SONY MAVICA STILL IMAGE VIDEO CAMERA

SONY GV-9
VIDEO WALKMAN TV/VCR

Genesis system with the power base converter.

Another optional accessory for the Genesis system is TeleGenesis, a device that allows two Genesis system players to compete with each other via a phone line. TeleGenesis is a modem that connects the two players' games together where each uses a TeleGenesis game cartridge. For example, in *TeleGenesis Baseball*, each player sees the action from his or her perspective—as either the pitcher or batter—and this perspective is maintained throughout the game. TeleGenesis game play can also include expanded team play, where two players compete against another duo.

In addition to the game console and one joypad controller, the Genesis system package will include the arcade game *Altered Beast*. Genesis will be available by the time you read this and will sell for about $200.

Although Nintendo did not debut a 16-bit video-game unit at the show, they did announce a new handheld unit. Called the Game Boy, the unit operates on AA batteries and weighs about ten ounces. It has a monochrome nonbacklit LCD screen and features digital stereo sound, a stereo headphone jack and the ability to link two units together for multiple-player competition. Two-person games, such as baseball, soccer and tennis, can be used via the Video-Link cable.

The Game Boy will sell for $90 and be packed with one game: *Tetris*. Other popular titles, such as *Super Mario Brothers*, will be available immediately. Game cards for the Game Boy will retail for about $20.

The fourth new video game announced at CES was from Atari. Called Lynx, it is a self-contained, handheld video-game system. The Atari portable game uses a 16-MHz processor and has a built-in 3.5-inch LCD screen that can display graphics with up to 16 simultaneous colors from a palette of 4,096 colors.

Features of the Atari game include four-channel sound, a headphone jack and the ability to rotate the screen image 180 degrees so that both left- and right-handed players can play. Also, it can be used individually or linked with up to eight other units for multi-player games. The system has 64K RAM and runs on AA batteries. It can also be powered by an AC adapter or used with a cigarette-lighter adapter.

Lynx will sell for $150 and be available soon. Games will sell for about $35 and come on credit card-sized ROMs. Included with the system will be *California Games* by Epyx. Other games, such as *Blue Lightning*, *Time Quests and Treasure Chests*, *Gates of Zendocon*, *Impossible Mission* and *Monster Demolition*, will also be available.

Other video-game offerings ranged from the exciting to the almost wacky. One of the most exciting—even incredible—was ESP's Battletech Center. ESP's booth was drawing huge crowds throughout the four days of CES.

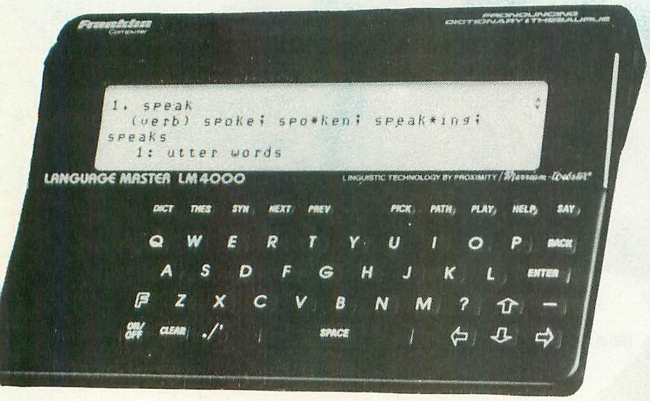
Battletech Center is an entirely new form of entertainment. Essentially, it's a military simulator at a price level that would be available to consumers via an entertainment center or a mini-entertainment park. The result is a blend of state-of-the-art graphics and a fantasy role-playing environment.

At a typical Battletech Center, two teams of four players compete in a computer-generated environment that can be configured for giant robots, tanks, jet-fighters—just about anything. The Battletech Center will operate in a 5,000-square-foot location, encompassing a lobby area, two combat-information centers, embarkation halls and two launch areas. All aspects of the center are designed to provide information for the participants and heighten the experience.

The lobby area will be decorated as a wardroom of a giant starship with TV monitors broadcasting fictitious newscasts from the future, cockpit simulator instructions and a "god's-eye view" of games in progress. The combat-information centers and embarkation halls will engulf players with the action and images of the fictional universe via staff, interior-design audio and video information. Here, the players will be handed their orders and formulate team strategies.

Next stop is the launch area. This is dominated by eight cockpits, each measuring 10′ × 3½′ × 5′. The setting is a fictional launch bay, and the players approach the cockpits on a catwalk.

Each person sits in his own enclosed cockpit, whose simulator allows him to see a graphic image of the other player's vehicle, movements and the surrounding terrain. Each cockpit is equipped with a 25-inch color monitor, which serves as the front window of the vehicle and displays main graphics of the surrounding environment and ensuing battle. A secondary screen is used for radar and a vehicle-damage display


Franklin Computer
PRONOUNCING DICTIONARY+THESAURUS
1. speak
(verb) spoke; spo*ken; speak*ing;
speaks
1: utter words
LANGUAGE MASTER LM4000
LINGUISTIC TECHNOLOGY BY PROXIMITY / Merriam-Webster

FRANKLIN COMPUTER LANGUAGE MASTER LM 4000

RICOH'S MC50 PORTABLE COPIER/DIGITIZER

center. Dual joysticks with trigger buttons, weapon-selection controls and throttle are used by the participant.

Other features include directional control pedals, ceiling message center and radio controls for communicating with teammates and attendant. An engine-instrumentation panel and control computer are also contained in the cockpit. Each cockpit's computers uses over 32 megabytes of RAM memory and communicates with the other cockpits via a local-area network. In addition, the main graphic display screen is capable of full real-time scaling, with each computer keeping an accurate mathematical model of the environment, each object, its placement status and the view from any given place.

Another important aspect of the cockpit is the sound system: It incorporates three sound-generation devices, which can reproduce sampled sounds, sampled sounds with frequency control and FM-synthesized sounds. Four speakers surround the player, with the addition of a large subwoofer under the seat. The sound system and placement of the speakers allow a specific sound to be placed anywhere around the player in two-dimensional space.

ESP plans to sell the Battletech Center as a turnkey operation, and they hope to have 150 centers operating around the country by 1990. The Battletech Center experience will last roughly 30 minutes from entering the lobby to exiting the cockpits. Admission is expected to be similar to the cost of going to the movies or bowling.

I had an opportunity to experience the Battletech Center and can only describe it as fantastic. Although the full simulation was not running during the demonstration I witnessed, I can tell you that I've never experienced anything like it. I look forward to one opening near me and plan to be one of its first visitors.

From Safe Care Products comes Homework First, a security device for the Nintendo Entertainment System. Essentially, it's a video-game lock that prevents the Nintendo video game from being used until it is unlocked. Homework First secures the Nintendo control deck by blocking the chamber where the game cartridges are inserted.

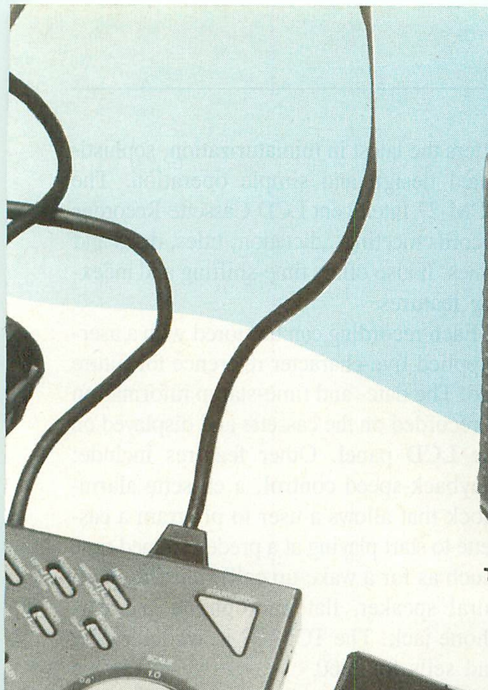
According to Safe Care, "There is a growing concern among parents, pediatrics [sic] and teachers across the country who feel Nintendo video-game addiction is affecting their children's schoolwork and other important responsibilities." Their product allows parents to manage when and how much time their children play Nintendo video games in their households. Price and availability of the combination-lock device are as yet undetermined.

Personal Electronic Products

This consumer-electronics category is quite broad. It consists of electronic watches, home security systems, health-care products, calculators, musical instruments, home automation and information products. Of course, CES provided the opportunity to see hundreds of these types of products in scores of manufacturers' booths. What follows is the cream of the crop.

Sharp was again showing the Wizard, a pocket electronic organizer. This $300 device contains essential business tools, such as a 200-year appointment calendar, phone directory, notepad, calculator and world clock in one handy gadget. The sleek 4″× 6″ unit weighs just eight ounces, has an eight-line by 16-character *LCD display* and is completely menu-driven for ease of use.

The Wizard features the ability to use plug-in IC software cards, such as a dictionary/thesaurus, time-management system and an eight-language translator. When slid into the software port, these cards also serve as a touch-sensitive panel for selecting the card's functions. Each of the cards is priced under $130.



TOSHIBA IC CARD CAMERA MONITOR

OPTONICA VOICE COACH

Sharp has now introduced two new memory cards that expand the original 32K memory capacity of the Wizard. A 32K and a 64K RAM card ($80 and $130, respectively) are now available and provide greater storage to the schedule, telephone and memo modes of the unit. Both cards are powered by their own lithium battery, which maintains the data for up to two years. In addition, private data files, requiring a user password, can also be stored on these cards.

Sharp also introduced the Organizer Link, an interface device for exchanging data with a PC. By linking the Wizard with a PC, the Organizer Link transforms the Wizard into a pocket-sized data bank. It allows users to write and edit data on a PC and then download it to the Wizard or transfer data from the Wizard to the PC.

The Organizer Link includes all necessary software and hardware to run on IBM-compatible PCs. The software is available on both 3½- and 5¼-inch floppy disks, and the hardware interface connects to the serial port of the PC. The Organizer Link retails for $150 and will be available by the time you read this.

Casio was also showing a personal information management tool similar to Sharp's Wizard. Called the BOSS (Business Organizer Scheduling System), the series of products includes a calculator, calendar, schedule keeper with multiple alarms, business-card management function, telephone-number directory and memo functions. In addition, a world-time function is included, which can display the time at virtually any point on the globe.

There are three BOSS models: SF-7000, SF-7500 and SF-8000. All three use a wide, 32-column by six-line LCD screen, have a 200-year calendar that can display two months at a time, and contain a secret-password function that protects private data. Although all three models share the same basic features, they differ in a couple of ways.

The SF-7000 has 32K of memory, whereas the SF-7500 and SF-8000 each has 64K of memory. The SF-8000 differs from the other two in that it has a stroke keyboard instead of a membrane keyboard. In addition, the SF-7500 and SF-8000 can communicate with other BOSS units via the included transfer cable. Using the optional PC-Link allows transfer of data to and from any IBM PC or compatible computer.

The Casio BOSS models sell for $220 (SF-7000), $240 (SF-7500) and $260 (SF-8000). They will be available by the time you read this.

Franklin Computer, makers of the Language Master series of handheld spelling checkers, had two interesting products being demonstrated. First, the Language Master 4000 is an all-in-one speaking electronic dictionary, thesaurus, phonetic spelling-corrector and word-game player that pronounces more than 83,000 words. Words can be entered the way they sound, and within seconds the LM-4000 will display the definition, parts of speech and hyphenation points on an LCD screen. Pressing a single key causes the unit to pronounce the word.

In addition to the above features, the LM-4000 has a built-in vocabulary-building tutorial that boasts a list of 3,500 words frequently found on such tests as the GRE, SAT and GMAT. Students can view randomly selected words and at the touch of a key display their meanings. The new speaking dictionary includes a built-in

ANOTHER INTERESTING PRODUCT IS THE VOICE FROM ADVANCED PRODUCTS AND TECHNOLOGIES CORPORATION. THE VOICE IS A TALKING LANGUAGE-TRANSLATOR THAT WILL ACCEPT VERBAL ENGLISH SENTENCES, TRANSLATE THEM INTO SPANISH AND THEN PRONOUNCE THEM. IT SOUNDS UNBELIEVABLE, BUT I SAW THIS THING WORK. IT'S AMAZING.

speaker, volume control and headphone jack. The LM-4000 will retail for $400 and be available by the time you read this.

The other Franklin product is another breakthrough in itself. It is the world's first electronic version of the Holy Bible. The Electronic Holy Bible is available in either King James or Revised Standard versions and contains both the Old and New Testaments. With the touch of a button, any passage can be located instantly, offering unparalleled accessibility.

For example, typing in "valley," "shadow" and "death" will display the 23rd Psalm. Franklin's proprietary search-technology allows the reader to easily find both familiar and obscure passages based on limited information. In addition, a word-usage thesaurus relates the 16th-century language of the King James version to modern-day English. The Revised Standard version incorporates 20th-century academic advances in understanding the original Greek and Hebrew texts.

The Electronic Holy Bible also includes advanced phonetic spelling designed for use with a biblical vocabulary. Even if the user does not know how to spell a particular word, it can be entered the way it sounds. Also, the built-in thesaurus and knowledge of English word-forms allows the Electronic Bible to offer help in finding passages.

Both versions of the Electronic Bible weigh just 13 ounces and measure 5.6″ × 5.6″ × 1.4″. They both contain a four-line by 37-character LCD display. The display also contains the names of the chapters of the Old and New Testaments. An "electronic bookmark" is also provided for quick return to favorite passages.

Complete with carrying case and batteries, the Franklin Electronic Holy Bible will be available by Christmas and retail for $299.

Another interesting product is the Voice from Advanced Products and Technologies Corporation. The Voice is a talking language-translator that will accept verbal English sentences, translate them into Spanish and then pronounce them. It sounds unbelievable, but I saw this thing work. It's amazing.

The Voice has no keyboard since it recognizes human speech. The output is fluent Castilian Spanish, for now. In the near future, other language cartridges will also be available for the Voice. But you cannot just pick up the Voice and start talking to it. Both you and the Voice need a period of training.

Since the Voice is speaker-dependent, it must learn to recognize your voice and the way you pronounce your words. There are some 500 words, word groups and sentence parts that need to be spoken into the Voice. To train the Voice, you press a key and a word or phrase appears on the LCD screen. As you say the word into the microphone, the Voice digitizes the sound and catalogs it along with what was displayed. The process of training the Voice is said to take less than an hour.

There are also rules you must follow when composing and speaking your sentences. Many voice-input devices have difficulty distinguishing "to," "two" and "too." To get around this, the Voice works mainly with groups of words rather than single words. For example, you cannot say "I would like two of these." Instead, you must say "I would like to have two of these." The Voice understands the context of the word "two" in this situation.

The company says it takes about two weeks of practice to learn how to assemble sentences that the Voice will understand. In addition, the key to proficiency in use is in training the Voice with a natural speaking voice so the words are easily repeatable. Once the Voice is trained, it will hold all the information it has learned about your voice for up to five years.

The Voice translator will sell for $2,000, and the cartridges will retail for $300 each. It is expected to be available by the end of the year.

Sony introduced a new line called "personal automation products" at CES. One is a handheld microcassette recorder that offers the latest in miniaturization, sophisticated design and simple operation. The TCM-27 Intelligent LCD Cassette-Recorder records meetings, dictation, titles, dates and times. It also offers time-shifting and indexing features.

Each recording can be stored with a user-supplied five-character reference for future use. The date- and time-stamp information is recorded on the cassette and displayed on the LCD panel. Other features include: playback-speed control, a cassette alarm-clock that allows a user to program a cassette to start playing at a predetermined time (such as for a wake-up call), built-in monaural speaker, flat microphone and earphone jack. The TCM-27 is available now and sells for $160.

Another new Sony product is the world's smallest pocket copier: the HCP-C10. The copier is a one-line plain paper copier and printer that allows you to reproduce printed materials virtually anytime, anywhere. Weighing just 4.6 ounces (with battery) and measuring a scant 3.9″ × 2.8″ × 1″, the Sony Pocket Copier will copy eight inches of a standard-type line. The rechargeable six-volt battery lasts for hours and a power-off function conserves battery power.

The pocket copier uses replaceable ribbon cartridges that print approximately 50 feet and cost $4.49. An optional AC adapter is also available for $25. The Sony HCP-C10 Pocket Copier is available now and sells for $160.

Olympus, the inventor of the microcassette tape-recording format, has a new microcassette recorder called the S811. Primarily meant for dictation, the new recorder weighs only 5.4 ounces and features electronic indexing (cue marking) to allow users to quickly find any recorded section on the tape. The built-in microphone has two levels of sensitivity: one just for the speaker's voice, which eliminates extraneous noise, and one for recording more than one person, such as in a conference.

All operation of the S811 is controlled by a single slide switch. With a single movement of the thumb, users can switch from dictation to stop, listen or review. Other features of the recorder include end-of-tape alarm and two tape speeds.

The Olympus S811 is a professional-quality dictation recorder that is available now. It has a suggested retail price of $289.

Olympus also introduced the Woodbury Cellular Interface System, the first product to allow direct connection to a cellular phone for recording. The interface connects between the handset and the transceiver by means of simple phone jacks. A very high impedance connection is used so as not to interfere or degrade the quality of the audio signal.

The unit automatically attenuates the outgoing audio signal and amplifies the incoming audio signal to achieve a balance between volume levels. The two signals are then mixed and fed to an output plug that fits Olympus microcassette tape recorders. Both voice-actuated and nonvoice-actuated recorders will work with the Woodbury cellular interface.

The interface system has been designed to work with Panasonic, Audiovox, Mitsubishi, NEC, OKI, Motorola, Mobira, Radio Shack and Uniden cellular phones. Available soon, it will sell for $100.

Ricoh has announced the arrival of the portable office with three separate but compatible products called the Portable Digital Information System. The ultra-compact system is quite literally an office in a briefcase, since it consists of a book-size copier, an IBM-compatible image scanner and a FAX module. All units are lightweight and battery-powered.

The heart of the system is the MC-50 compact flatbed copier/digitizer. It makes half-page copies of any flat surface without the need to roll over the surface like some other portable copiers. It will print on plain or thermal paper and acetate sheets (for vugraphs). Using a 798-pixel image sensor, it produces crisp, clear copies with a 200 dot-per-inch density and a 200 line-per-inch horizontal and vertical resolution. Standard copy size is in the A-6 format.

The second module is the IM-A image controller and scanner. It expands the copier's capabilities, allowing you to make enlarged (up to 200%) or reduced (50%) copies, copy halftones (photographs) with up to 16 shades of gray and reversed prints where whites become blacks and vice versa. It will also produce mirror-image copies. The IM-A unit can connect to an IBM PC or compatible, comes with its own software and creates files compatible with most desktop-publishing programs.

The IM-F50 FAX unit transforms the copier into a transceiver, allowing it to both send and receive facsimile transmissions. It will send 4″× 6″ documents, which will print out full size at the receiving end. Larger-size pages sent to the FAX unit are automatically reduced to 4″× 6″ and printed by the copier. All data transmissions are sent in half-duplex mode and at a speed of 4800 baud.

The Ricoh MC50 copier sells for $540, the IM-A Image Controller is $510, and the IM-F50 costs $540. All products are available now. Perhaps not everyone needs a $1,500 office in a briefcase, but if you are a salesperson, travel extensively, do research or perform a host of other activities that can benefit from portable copying and FAXing capabilities, the Ricoh Portable Digital Information System may be worth investigating.

Video

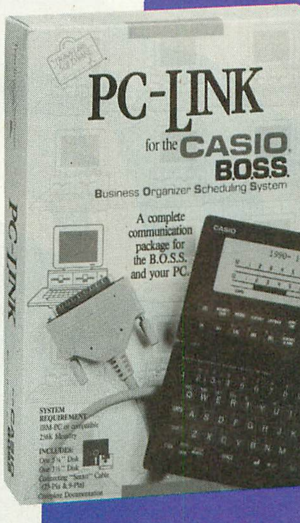
Several companies were showing LCD projection-televisions. Although this sounds like a contradiction, these systems use LCD panels to separate the source beam into red, green and blue spectral components. Then a variable zoom lens combines and projects the image onto a screen. By using LCD panels instead of CRTs, the projector is smaller and lighter. Further, LCD projection-TVs are typically brighter than conventional projection-TVs.

Toshiba was showing their LCD projection system on a 78-inch screen. The system can project images onto screens ranging from 20 to 100 inches (diagonally) in size. Video resolution of the Toshiba unit is 340 horizontal by 240 vertical lines. The LCD projector is being targeted for educational and business purposes, as a movie system for airlines and for home-entertainment systems, enabling consumers to enjoy movies with image quality equal to that provided by theatres. Price and availability of the Toshiba LCD projection system have not been announced.

Sharp was showing a projection-TV system called the XV-100. The Sharp model is similar to the Toshiba unit, except it has a projection range of from 25 to 100 diagonal inches. The XV-100 offers over 268,000 pixel elements and over 300 lines of horizontal resolution. The variable zoom lens permits changing the image size without moving the projector or losing picture quality.

The Sharp XV-100 weighs about 30 pounds and offers simple installation because it is convergence-free. It features two video inputs and an S-VHS input for connection to VCR, video laser disc or other external signal sources. The Sharp XV-100 will be available by the time you read this and retail for $6,500.

Panasonic was showing an interesting, if somewhat unusual, VCR. The model PV-S4986 is a S-VHS unit with typical features, a remote control and the quality expected of a top-of-the-line S-VHS machine. What makes it unique is its ability to be pro-


PC-LINK
for the CASIO B.O.S.S.
Business Organizer Scheduling System
A complete communication package for the B.O.S.S. and your PC.

CASIO PC-LINK FOR THE B.O.S.S.

CASIO B.O.S.S.
BUSINESS ORGANIZER SCHEDULING SYSTEM

TOSHIBA IC CARD CAMERA

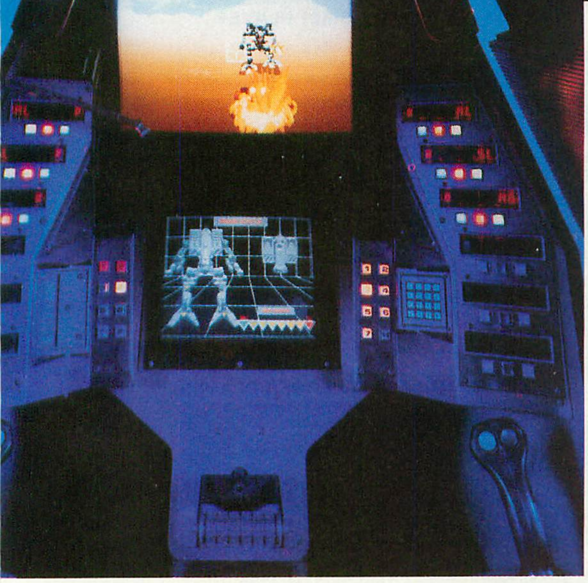
grammed from a telephone. You heard me.

Say you left for vacation and forgot to program the VCR to catch Hulk Hogan in the title match. No problem. Just call up the Panasonic PV-S4986 from any touch-tone phone, and it will verbally prompt you through the programming steps. Once you are finished programming the VCR, the machine will tell you what you just programmed.

I've heard of "on-screen" programming, but "on-the-telephone" programming is. . .well. . .different. The Panasonic VCR also features bar-code programming (for when you *are* at home) with an on-screen display. The Panasonic telephone-programmable VCR should be available by the time you read this.

Laser Sharp

One of the hottest video categories right now is laser videodisc. I know what you are thinking: "Didn't that die a couple of years ago?" The answer is a definite "No!" You see, in the early 1980s two videodisc formats came about. One was developed by RCA and called Select-A-Vision. It used a mechanical stylus that tracked the grooves in the disk similar to a record player. Each time a disc was played, a slight degradation of the disc occurred. About 1985, RCA discontinued production of their videodisc format.



BATTLE TECH CENTER

The other videodisc format was developed jointly by Pioneer and Philips. It's called LaserDisc and uses a laser to read the pits and bumps on the disk surface similar to a compact disc player. Since there is no physical contact with the surface of the disc, the disc itself will last forever (with reasonable care). Moreover, the video quality of the LaserDisc exceeds even that of S-VHS. Typical resolution is over 400 horizontal lines.

Not only does the LaserDisc format have video that is superior to tape, but it uses digital audio, exactly the same as is used by CDs. LaserDiscs offer stereo sound and most are encoded with surround sound, just like in movie theatres. In Japan, consumers favor LaserDisc four to one over videotape.

For most of the last decade, LaserDiscs have never done well. Especially when compared to the overwhelming success of videocassette recorders. This was caused partly by the fact that since LaserDisc players could not record, most consumers thought less of them. Also, LaserDisc players have been relatively expensive—at least $700—which is twice the price of the majority of VCRs.

Despite the lack of mass market acceptance, LaserDisc video has slowly been accepted by videophiles. They know about and can appreciate the superior audio and video qualities of the format. When combined with a top-quality audio system and maybe a large screen (or projection) TV, this format provides the best currently available method of duplicating the theatrical movie experience.

Now, several factors have combined to make LaserDisc video more popular than it has ever been. First, the introduction of CD-Video (which combines five minutes of video information with 20 minutes of audio) has resulted in the arrival of "combi-players." These players are so called because they can play regular audio CDs, CD-Video discs and LaserDiscs. "Combi-" machines are now available from major companies, such as Pioneer, Mitsubishi, Sony, Sharp, Yamaha, Philips, Magnavox and others.

Since these combi-players can play all laser-oriented formats (including the recent three-inch CDs), people who would otherwise not consider a LaserDisc purchase are starting to be interested. The second reason for the renewed interest in LaserDisc video comes from Pioneer, which has recently introduced new player models at record-breaking prices. Their new line includes the CLD-1070, a combi-player that lists for $600. That translates to a discount price of under $500 for a machine that is both a CD player and a LaserDisc player. Pioneer's new CLD-870 is a LaserDisc-only player that retails for $500.

The third major reason for all of the excitement about LaserDisc is the increased availability of titles. Until the last couple of years, new movie titles would be released on tape but not be available on LaserDisc until several months after or perhaps not at all. Now, all that has changed. Major motion pictures are now released in both tape and disc format simultaneously. There are currently over 3,000 LaserDisc titles available and new ones are being released at the rate of over 60 per month.

LaserDiscs generally cost less than videotapes. Most major movies on laser have a $25, $30 or $35 retail price, before discounting. Blockbuster titles are promoted heavily and usually arrive at the store at the $25 price. Also available on LaserDisc are

CANON XAP SHOT
STILL VIDEO CAMERA

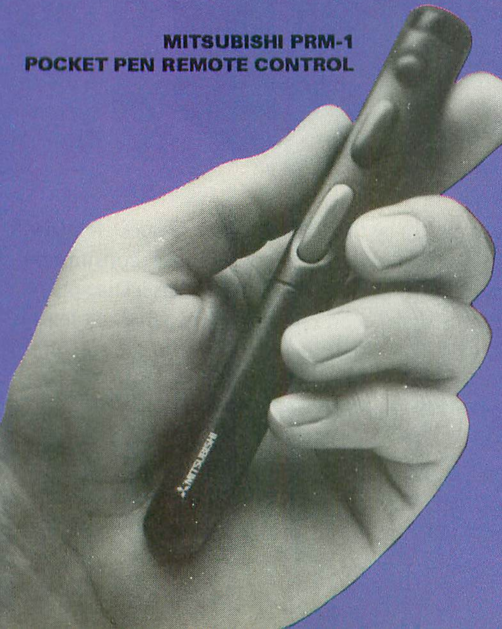
SONY VIDEO HI8 HANDYCAM

YAMAHA CDV-1100 CD VIDEO COMBI PLAYER

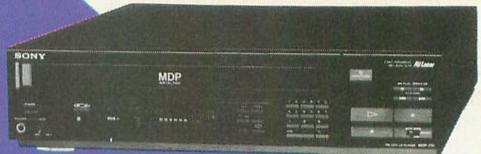
SHARP MULTI DISC CHANGER

SONY MDP-510 AV LASER PLAYER

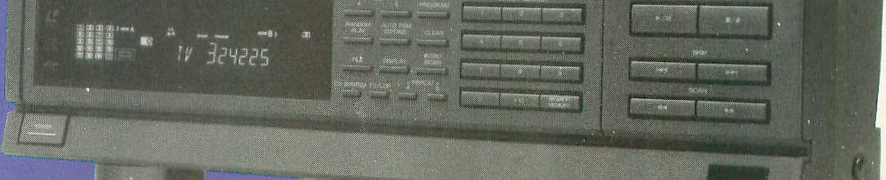
MITSUBISHI PRM-1
POCKET PEN REMOTE CONTROL



SONY MDP-210 AV LASER PLAYER



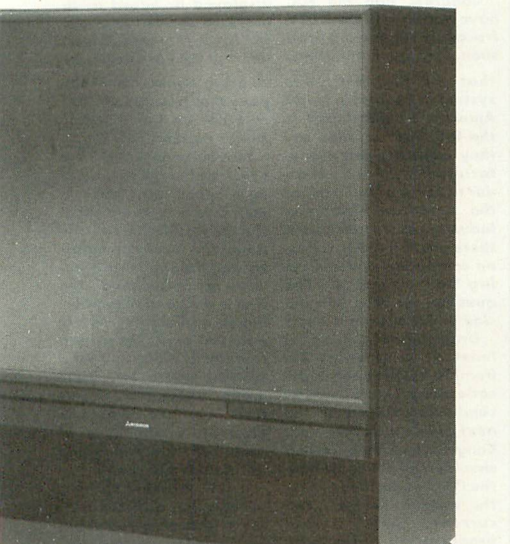
MITSUBISHI M-V7010 LASER DISC PLAYER WITH REMOTE



rock videos, complete operas and special programming, such as the *National Geographic Nature* television series.

The laser-video market in the United States is beginning to expand. With lower-priced players and the increased availability of discs, LaserDisc video is leaving the realm of the video elitist who demands superior quality (and is willing to pay for it) and entering the domain of the typical consumer who wants to get the most for his money.

At CES a year ago, Sony Corporation of America announced a product that was truly revolutionary. It was a portable video product that was designed for individual use. Sony announced a new line of products called Personal Video with the introduction of the GV-8 Video Walkman TV/VCR combination unit. It featured a backlit, three-inch (measured diagonally) color LCD screen and an 8mm video recorder. The GV-8 can be used as a stand-alone device or connected to any TV monitor for large-screen viewing. Also, the unit has audio/video input and output connectors to allow connection to another VCR of any format.



MITUBISHI 70" STEREO MONITOR/RECEIVER WITH REMOTE

OLYMPUS S811 MICROCASSETTE RECORDER

SONY VIDEO HI8 RECORDER

The 8mm VCR portion of the GV-8 features high-speed search, a one-day/one-event timer for time-shifted recordings and a built-in TV tuner with full VHF and UHF channel reception. It also has a sleep timer that automatically turns off the unit after one hour. A linear time-counter displays the tape position in hours, minutes and seconds. The GV-8 operates on either AC, rechargeable nickel-cadmium batteries or alkaline batteries when used with an optional external battery case.

Now Sony has introduced a new model called the GV-9. The new Sony TV/VCR shares most of the features of the original GV-8 but has an 80% larger screen. The new model has a four-inch color LCD screen in addition to its full-featured 8mm videocassette recorder. The GV-9 Video Walkman weighs less than three pounds, operates on batteries for two hours and retails for $1,500.

Taking a lead from Sony, Sharp introduced a portable VHS VCR with a four-inch color LCD monitor. The model VC-V540U is not quite as small as the Sony Video Walkman, but shows innovation nonetheless. The unit weighs roughly ten pounds and measures about 12"×10×5".

Operating on AC, DC or car battery, the Sharp VC-V540U plays full-size VHS video cassettes and contains separate audio/video input jacks. However, it is a player only since it contains no tuner for receiving broadcasts. It will be available this fall and sell for $1,900.

A couple of interesting remote controls were debuted at CES. Optonica showed the Voice Coach, a speaking remote control for their model VC-G99OU S-VHS videocassette recorder. Adding new meaning to the term "user friendly," the Voice Coach verbally walks users through the process of programming their VCR for time-shifting.

With a 50-phrase and 500-word vocabulary, the talking remote control gives easy step-by-step instructions to the user. An on-screen programming display aids the process, resulting in almost mistake-free programming. The remote control also features eject, tracking adjustment and variable-speed slow-motion controls. The VC-G99OU VCR offers four heads, MTS high-fidelity stereo sound, a 209-channel tune, video index search and a one-year/eight-event programmable timer. The Optonica VCR and talking remote control retails for $1,000 and will be available by the time you read this.

The other interesting remote control comes from Mitsubishi. It is the Pocket Pen remote control, a minimalist approach to remote control that is the same size as an ordinary fountain pen. Designed for easy operation, the "pen" remote operates the basic functions of all Mitsubishi televisions up to four generations old. Mitsubishi sees the pen remote as a second remote control for those households in need of such a device.

The Pocket Pen remote features power on/off, channel up/down and volume up/down controls. It is powered by two replaceable "N" batteries. The Mitsubishi pen remote will be available by the time you read this and sell for $25.

Still Video, Still

The wait is over! By the time you read this, both Sony and Canon will be heavily promoting their still-video cameras in time for the holiday season. In fact, as this is being written, print and TV ads have already started appearing.

To refresh your memory, still video refers to using video technology for making still images or snapshots. All still-video cameras use a two-inch floppy disk that stores 50 frames or images. The erasable magnetic disk used to store the electronic images is reusable and is compatible with all makes of still-video cameras. Each image is stored as digital data and can be instantly retrieved and viewed through a television.

All still-video cameras are based on the Hi-Band video standard and have the same quality pictures and basic features. They are capable of capturing images with up to 300 lines of resolution. However, each manufacturer has decided to provide and implement its features in a slightly different way. For example, Canon uses the camera itself to play back the images, whereas Sony requires a separate player to view the image.

Most of the cameras have a built-in flash, can take individual frames or multiple frames per second and are about the same size. Automatic exposure and white balance are also standard features. Other features include self timers, rechargeable batteries and point-and-shoot ease of use.

The Sony Mavica (an acronym for Magnetic Video Camera) MVC-C1 weighs just over a pound and measures 5¾"×2¼×4¼". It includes a 15mm f/2.8 fixed-focus lens that provides sharp images of subjects at a distance of 1.5m to infinity.

The built-in flash is automatically activated in low-light conditions, and the camera is capable of shutter speeds from 1/60th to 1/50th of a second.

In addition to single-picture shooting, the Sony Mavica offers continuous high-speed image recording at either four or nine frames per second. A "blank search" function automatically advances the disk to a blank frame in order to avoid accidentally erasing an image (to erase a picture, the separate playback adapter must be used). The playback adapter, the MAP-T1, is powered by either batteries or AC and can fully charge the camera in one hour. The Sony Mavica retails for $650, and the playback adapter sells for $250. Both are available now.

Canon's Zap Shot still-video camera uses an 11mm f/2.8 fixed-focus lens and features a built-in macro mode for extreme close-ups at 12 inches. The camera has auto-focus capability as well as a built-in flash. Recording and playback functions are all contained in the 15-ounce body.

The Zap Shot can take pictures either individually or at three images per second and contains a built-in self-timer. When taking pictures, you can let the camera automatically find the next-highest blank frame, or you can manually set it to any frame you want. The Zap Shot sells for $1,000 and is available now.

Similar, but not exactly the same as the still-video cameras described above, is Toshiba's IC Card Camera System. The prototype system being demonstrated consisted of a camera, portable player and "Electronic Album" recorder/player. The Toshiba system is really a new type of still camera that offers superb picture quality and uses IC memory cards rather than two-inch floppy disks to store the pictures.

Picture resolution is 400 horizontal lines, which exceeds that of S-VHS video images, and compares to the 300-line resolution of still-video camera systems. Since no disk drive is used, the camera can be more compact and reliable. The camera uses a CCD image sensor with 400,000 pixels and has autofocusing, automatic exposure control, automatic white balance and high-speed sequential (five frames per second) picture-taking functions. Each IC card can store 13 separate images which can be viewed when the camera is connected to a standard NTSC television.

The Portable Player can be used to monitor the stored images on the IC cards and has a four-inch color LCD screen. The Electronic Album is based on a DAT recorder and can store 1,600 pictures on one DAT cassette. Users can overlap an audio message to play concurrently with the photo images. The machine is equipped with a random-access function so that any photo image can be instantly retrieved by keying in its number.

It is expected that by the year 2000, the worldwide market for electronic still cameras will reach ten million units per year. Although the Toshiba IC Card Camera System is still in development, it is expected to be available as a professional system within a year. The cost for the entire system was said to be about $5,000. However, a consumer version of the camera will probably be available in two or three years.

Wrapping Up

I'm never disappointed when I attend the semiannual Consumer Electronics Show. Although I may not be able to afford to buy everything I see and want, I do get the opportunity to view the latest in consumer electronics technology. Being able to see and even use such products as the Voice translator, the IC Card Camera System, Battletech Center, Electronic Bible, Portable Digital Information System, HDTV, LCD projection-TVs, the newest Laser Disk hardware and all of the rest of the neat things is, in a word, exciting.

The advances in technology have been incredible. Sure, most of the new products I see twice a year are meant for entertainment. But more and more it seems that technology is providing a means to improve our lives. Examples: the Electronic Bible will allow more people to study the Holy Bible without having to first become a scholar. With this type of accessibility, perhaps these important words will reach more people and foster better understanding among ourselves. The medical and scientific uses for HDTV have already been proposed and are starting to be implemented. The Voice translator is a first step toward improved communication.

Technology is no doubt making life easier. And in some small ways it is making life better. I am fortunate to be able to view it firsthand and report on it. Enjoy.

HDTV Status

The debate over HDTV (high-definition television) continues. Viewing HDTV is an amazing experience. The picture quality is crystal-clear and almost appears three-dimensional. The audio is digital-quality and has the capability for surround sound. Anyone who has seen and heard an HDTV system can't help but be excited about this technology and have their own opinions about how it should be implemented in North America. However, the issues have more to do with politics and economics than technical considerations.

Fundamentally, HDTV technology is incompatible with the existing NTSC broadcast standard. The amount of information or bandwidth required to produce the high-quality widescreen image is twice that of a single broadcast channel. Therefore, the options boil down to either instituting an entirely new broadcast standard (making the current broadcast-TV system obsolete) or somehow making the new HDTV format compatible with existing standards so that current TVs could at least receive regular broadcasts.

Congress and the FCC have gotten into the HDTV fracas by suggesting (and soon perhaps mandating) that the high-definition system to be used in North America be developed in the U.S. Further, they say they want the sets manufactured in the U.S., in order to cause a "rebirth" of the American electronics industry. As it stands now, there are more than a dozen competing systems vying for approval. The question is: Who will decide which system is best?

Unfortunately, HDTV (at least as a mass-market item) is still probably a decade away. The FCC will be conducting tests over the next couple of years—with Congress looking over their shoulder, no doubt. Of late, the FCC has decided to "let the market decide." The current AM stereo-broadcasting fiasco (with two incompatible systems still on the air) is a result of this policy.

Assuming the contenders can be narrowed down to a choice of just a few, who will decide what the criteria will be to select the system? Hopefully, the choice will be based on technical superiority rather than politics.

DAT Update

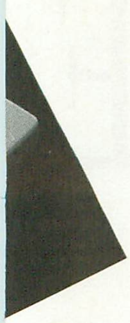
Digital Audio Tape (DAT) has not yet arrived on American shores. The long-awaited audio product is still embroiled in controversy since the Recording Industry Association of America (RIAA) is concerned about consumers making near-perfect recordings of CDs, thereby causing millions of dollars in lost revenue and perhaps the large-scale piracy of CDs. The hardware is currently available in Japan, although it is still rather expensive ($1,000-$2,000) since it has not yet become a mass-market consumer item.

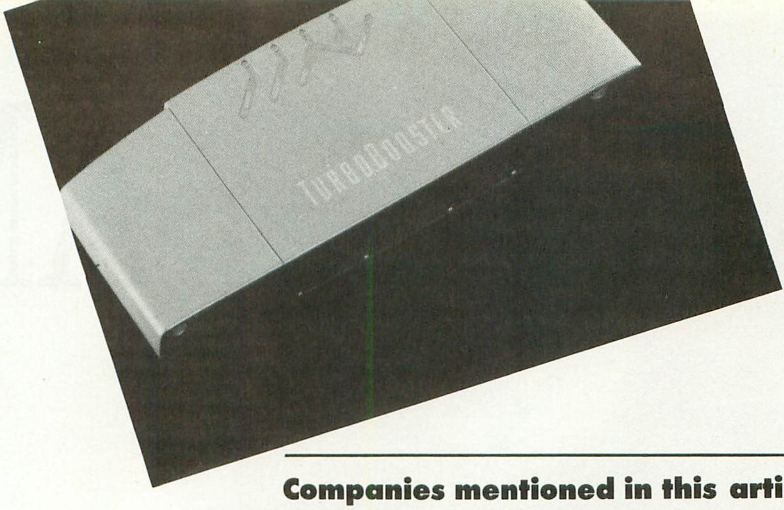
At previous Consumer Electronics Shows, DAT recorders were everywhere. Almost every major audio-and-video electronics company booth had at least one model on display. However, the recorders were not so prominently displayed or hyped as in previous shows. One reason is the threatened lawsuit that the RIAA promises to bring against any company that tries to sell a DAT recorder on the U.S. consumer market.

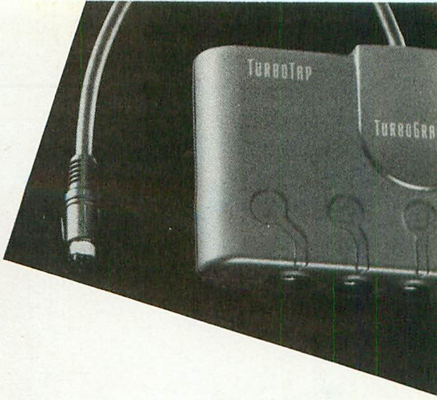
Several events that have occurred independently of each other may, each in their own way, bring about the reconciliation of the American recording industry and the Japanese hardware manufacturers. The first is the possibility of using an anti-copy system in each DAT deck. The current favorite is Philips's Solo II system, which lets users make unlimited copies from CDs, but only a single digital-to-digital copy of each DAT tape. This would not defeat the purpose of the DAT machine for consumers, but it would stifle (or at least slow down) any large-scale pirating attempt.

The second significant event is the marketing of the first DAT deck in the United States. As this is written, Nakamichi has already begun selling a DAT unit through selective retailers. The machine is the Model 1000 and it sells for a cool $10,000. Nakamichi has taken steps to protect their retailers in the event that the RIAA decides to sue. Dealers buy a demo model and act as agents for the product. If the consumer wants to buy one, it is ordered direct from Nakamichi, shipped to the individual and the dealer gets a commission.

The Nakamichi DAT machine has no copy-guard system and, in fact, can make direct digital record-







ings from a CD. It is believed that the RIAA feels that the Model 1000 is aimed at a limited market (given its price) and therefore does not propose a threat. However, if price—and not technology—is going to be a criterion, at exactly what price point does it become an issue to the RIAA? The answer is not clear.

Another area of activity that may have impact on whether (or when) consumer DAT arrives in the U.S. comes from a company called DAT USA International. They are an importing company that has quietly been importing Japanese DAT machines into the U.S. For two years, the company has been selling machines that are considered ''gray-market'' goods, meaning that they arrive through non-normal distribution channels and may not have a U.S. warranty. Now DAT USA intends to cover all machines that it sells with a valid U.S. warranty.

The Japanese companies have everything to gain by providing DAT USA with products. It not only clears the way for the eventual (being optimistic) future efforts of these companies to sell DAT in the U.S., but it also shields these companies from direct action from the RIAA. However, DAT USA expects the consumer electronics industry to come to its aid in the event of litigation by the RIAA.

Will the world's record makers continue to place a stranglehold on the sale of DAT in America? If DAT is dead, other soon-to-be technologies, such as recordable CDs, pose the same alleged threat to the RIAA. Current analog tape-recording is inferior to the original, contains audible wow and flutter and lacks the dynamic range that most consumers have come to expect from CDs. The real question is: When will some company step up to the challenge of introducing a home-recording technology that meets current standards of audio reproduction, RIAA or no RIAA?

Companies mentioned in this article:

Advanced Products and Technologies
15444 N.E. 95th Street
Redmond, WA 98052
(206) 883-8297

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

Canon, Inc.
One Canon Plaza
Lake Success, NY 11042

Casio, Inc.
570 Mt. Pleasant Ave.
Dover, NJ 07801
(201) 361-5400

ESP Corporation
1026 W. Van Buren
Chicago, IL 60607
(312) 243-5660

Franklin Computer Products
122 Burrs Rd.
Mt. Holly, NJ 08060
(609) 261-4800

Mitsubishi International Corporation
Communication Equipment Sales Division
879 Supreme Drive
Bensenville, IL 60106
(312) 860-4200

NEC Home Electronics (U.S.A.) Inc.
1255 Michael Drive
Wood Dale, IL 60191
(312) 860-9500

Nintendo of America, Inc.
P.O. Box 957
Redmond, WA 98052
(206) 882-2040

Olympus Optical Company, Inc.
Crossways Park
Woodbury, NY 11797
(516) 364-3000

Panasonic Co.
One Panasonic Way
Secaucus, NJ 07094
(201) 348-7000

Pioneer Electronics (USA), Inc.
2265 East 220th Street
Long Beach, CA 90810
(213) 835-6177

Ricoh Consumer Products Group
155 Pasaic Avenue
Fairfield, NJ 07006

Safe Care Products
35 West 528 Parsons
Dundee, IL 60118
(312) 551-1199

Sega of America, Inc.
P.O. Box 2167
South San Francisco, CA 94080

Sharp Electronics Corp.
Sharp Plaza
Mahwah, NJ 07430
(201) 529-8981

Sony Corporation of America
9 West 57th Street
New York, NY 10019
(212) 418-942

Toshiba America, Inc.
82 Totowa Rd.
Wayne, NJ 07470
(201) 628-8000

GAME

LOST DUTCHMAN MINE

Magnetic Images Co.
P.O. Box 17422
Phoenix, AZ 85011
$49.95; Color only

Reviewed by Peter A. Smith

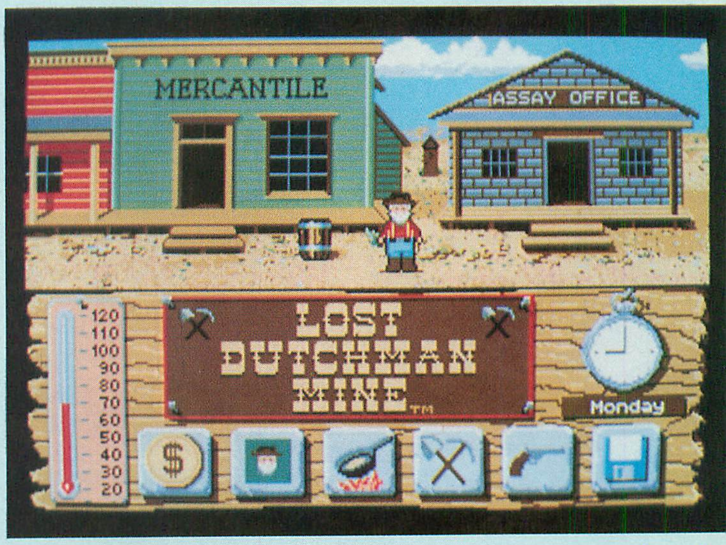
***Lost Dutchman Mine* is a recent release** from Magnetic Images in which you play the part of a grizzled prospector searching for the fabled mine of the title. The game comes on two disks, is not copy-protected and can be installed on a hard disk.

Atmospherically,Magnetic Images has gone the extra yard for this game. The graphics are cartoonish and well executed, the music is entertaining yet unobtrusive (with MIDI keyboards supported), and game play is punctuated by digitized sounds. I may as well note right off that Version 1.00 had a bug in the sound routine, which caused intermittent crashing, but Magnetic Images assures me that this problem has been fixed in Version 1.02.

You start the game in the town of Goldfield with $150. You must use this cash wisely to outfit yourself for exploration. Among your options in town (chosen by guiding your character into different buildings) are buying food and tools, having your gold assayed (turned into cash), or playing poker with Dapper Dan. If you choose the latter, you will find Dan is a fairly decent poker player and his digitized voice really sets the Old West mood.

As time passes, you will need to eat and drink in order to survive. The bottom third of the screen contains a series of icons dealing with cash, health, food, equipment, bullets and the disk. Clicking on cash or bullets shows you how much of each you have. The disk is for saving and restoring games. Clicking on health shows the number of wounds you've sustained, as well as bar graphs representing hunger, thirst and overall health. Clicking on the food icon allows you to eat any of your consumables with a resulting improvement in well-being. Unfortunately, you cannot view your health status and eat at the same time; this necessitates a lot of clicking back and forth to

SHELF



ensure that you've eaten enough without any waste. Last, the equipment icon shows assorted tools and ore carried.

Leaving town amounts to zooming out to an overhead view of the surrounding countryside, where you are portrayed by a square cursor. Using your joystick, you can head to wherever you'd like to search for gold. It's a good idea to visit the river first. There you can fill your canteens, fish for your dinner and pan for gold ore. While in the wilderness, hitting the fire button zooms you into a first-person view of the surrounding area; it is here that you can look for distant mines and caves.

When a mine or cave entrance is approached, a press of the fire button will take you inside, whereupon the view switches to a cutaway of the mine shaft with you at the mouth. While exploring the tunnels, you will find pockets of gold ore. Assuming you brought along a pickax, pressing the fire button will collect the ore. After you have all the ore you can carry, or all that you can find, it's time to journey back to town to get it assayed.

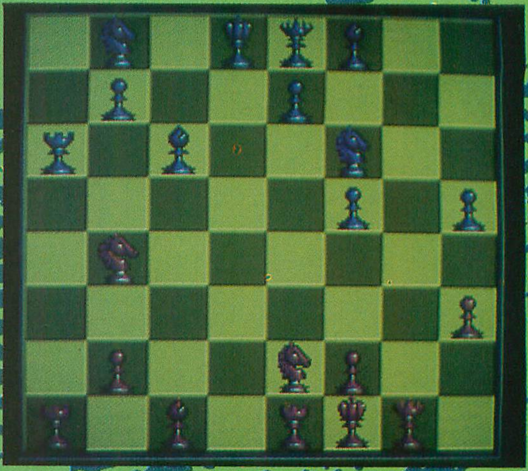
While wandering the wilderness you may be ambushed by a rattlesnake, Indians or bandits. Each encounter is handled in the same manner, offering a choice of running or fighting. If you choose to fight, your joystick will control a cross-hair cursor in a first-person view. Use it to shoot your enemy before he shoots (or bites) you. A rattlesnake can be eaten in a pinch, and sometimes a dead bandit will net you a reward. Choosing to run is always a safe option, but if you are up against Indians or bandits you will forfeit your possessions.

Overall, *Lost Dutchman Mine* is well executed. However, the initial struggle to become well equipped is over all too soon, and gold becomes meaningless as you end up with huge amounts of money and nothing to buy. At this point, the search for the Lost Dutchman Mine is your only concern. Finding the needed clues takes persistence but can become boring. Once a few pieces of the puzzle are found, interest picks up again. After you find the mine and stake a claim, the game ends.

For $49.95, *Lost Dutchman Mine*, though nicely implemented, doesn't provide your hunger for adventure with much to chew on. I couldn't help feeling that I had tasted a great sauce without enough meat.

Battle Chess

Interplay Productions
1575 Corporate Drive
Costa Mesa, CA 92626
(714) 545-9001
$49.95; Color Only
Reviewed by Frank Eva



Chess programs have been around for a long time. With so many of them on the market, one can't help but wonder why someone hasn't tried to add a little spice to this popular but generally unspectacular game. Finally, someone has. Interplay's *Battle Chess* is a visual treat as well as a good version of the standard chess game.

The players' pieces are depicted in animated human form. The pawns and knights are actual soldiers dressed in medieval armor. The bishops look like miniature priests. The rooks (sometimes referred to as "castles") resemble stone watchtowers.

Actual digitized sound effects, such as clanking armor, accompany the animations. When an attack occurs, the animations depict a battle between the two pieces, and although each confrontation has a predetermined outcome, the fight is enjoyable to watch while plotting another move. Most battles are not without humor. When a pawn attacks another pawn, he lances his opponent in the foot; the crippled pawn then hops up and down on his good leg until the final death blow is struck. Then the conquered piece dissolves pixel by pixel and vanishes.

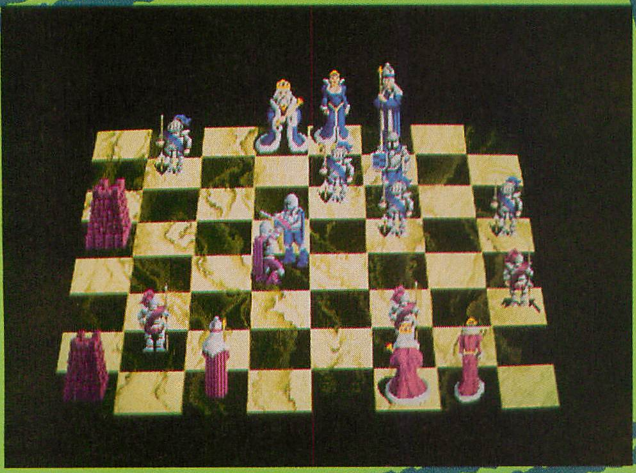
Rooks transform into huge living rock monsters that taunt their opponents when attacking. Some of the pieces put up a valiant struggle; others will simply drop their weapons in fear and wait for the inevitable. When a rook takes a queen, he tips her straight up in the air and swallows her down, whole!

The queen uses magic to do away with her opponents. At her command are lightning bolts and whirlwinds. The king has his own bag of tricks, including shooting an opponent with a gun...foul play! When a knight meets his end, he does so piece by piece; there goes an arm...now, a leg...another leg....

Other features include an opening library of 30,000 moves and the ability to play against the computer, a friend, or have the computer play itself. Also, a chess novice can take back moves or have the computer suggest one. Classic chess problems can be set up on the board so that seasoned players can hone their skills. There is even a replay feature, though I have my doubts about the value of such an option. After all, this isn't Monday-night football.

In *Battle Chess*, the computer's level of competence is determined by the length of time it's allowed to think. There are nine levels ranging

ST

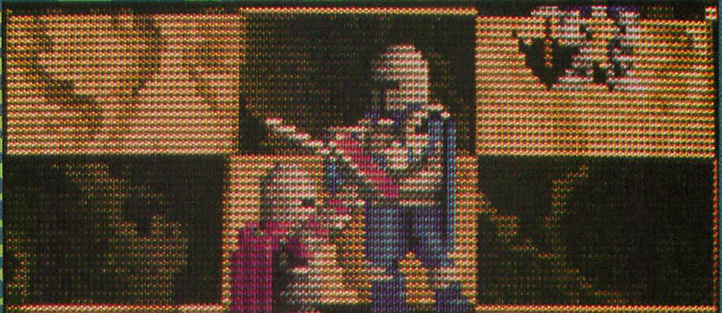


INTERPLAY'S *BATTLE CHESS* IS A VISUAL TREAT AS WELL AS A GOOD VERSION OF THE STANDARD CHESS GAME.

from five seconds to more than 21 minutes. Another level allows input of a time limit from one to 10,000 minutes. The computer can be forced to make a move, even if it isn't quite ready.

Rookies, like me, are probably more interested in a mode of play that simulates another human rookie. In other words, a mode in which the computer makes random mistakes and allows occasional advantages to go to its human counterpart. *Battle Chess* does not offer such an option, but it has one that is even better: modem capability. This makes any player who owns the game a potential opponent.

The program comes on two disks,which are not copy-protected. Interplay recommends copying the masters and playing with only the duplicates. A program is included that automates the process and includes the ability to combine both disks into one double-sided disk or load them onto a hard drive.

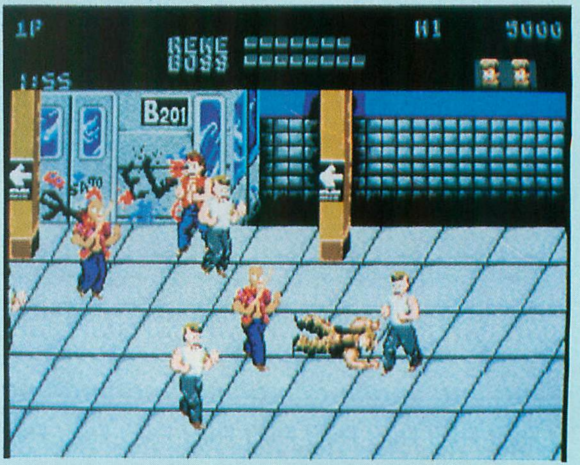


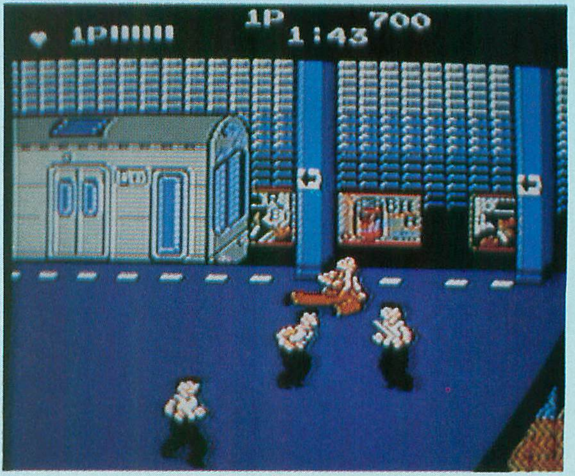
Nice functionality and great artwork, animation and sound effects all blend to make *Battle Chess* a real winner.

ST

RENEGADE

TAITO SOFTWARE INC.
267 WEST ESPLANADE
NORTH VANCOUVER, B.C.
CANADA V7M 1A5
(604) 984-3344
$29.95; COLOR ONLY





REVIEWED BY STEVE PANAK

Often when I look at a game from a company I've never encountered before, I expect the worst. The general rule seems to be that the big publishers get all the good programs, leaving the smaller outfits to fight for what is left at the bottom of the barrel. Most of the small outfits end up fighting over table scraps. And while Taito is by no means a small concern, these musings come to mind because I've just started to look into their catalog and all their games are new to me. Hopefully, some of their future products will have a lot more to offer than *Renegade*. Not that *Renegade* is altogether bad; rather, it is undistinguished.

In each of the game's five levels, you battle against an onslaught of vicious henchmen who block your way to their gang leader. Each thug does his best to take you out, whether it be with a flurry of flying limbs or a well-placed club against your head. Your goal is as simple and primitive as the rules that govern play: survive.

To help you survive, you have only your fists and feet. Of course, a little street smarts helps as well. By moving the joystick, you navigate about on the screen. Pressing the button down activates any of a number of offensive moves, such as a back kick or a punch to the head. The only strategy that develops is the need to take out the men with weapons first and, of course, to keep moving. Standing still for even a moment invites attacks. With only three lives, each of which can stand only about nine hits, you don't want to invite attacks. You'll have enough uninvited ones.

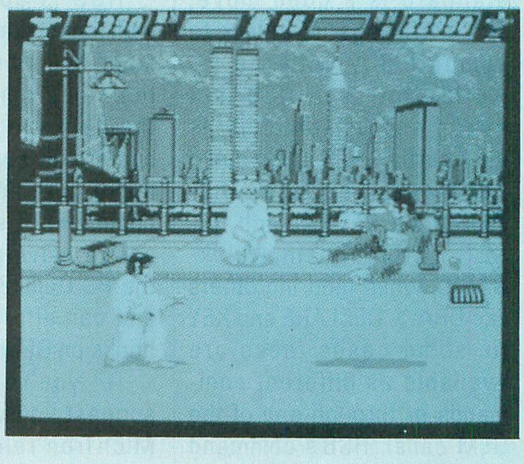
As you deal with each thug, you'll find that it takes more than a knockdown to eliminate your opponent. Fortunately, a renegade makes his own rules, and the first one you might devise is that you *do* hit a man when he's down. That's what the crouch-and-punch move is for. Once beaten senseless, a hood finally disappears. But what about the half-dozen milling behind him? Only by getting past every opponent will you confront their leader.

Unfortunately, the game falters where it counts: graphics and control. The graphics are average, the animation uninspired. But while blocky images can be forgiven, a sloppy control interface cannot. Even though the control pattern is logically laid out and hence simple to learn, it never seems like you're in control. Move the joystick to jump kick, you end up punching. More infuriating is the fact that it doesn't let you direct your attack to the left and right. Rather, it forces you to fight only in the direction you last moved. You have to release the button and turn before you can attack the thug behind you. This costs valuable seconds you can't afford to spend. It would have been much better to be able to attack in all directions as controlled by the joystick movement. Don't get me wrong, you do learn, you will improve. I just didn't enjoy the game enough to try.

Another drawback is the limited longevity of *Renegade*. Once you've completed all levels, there is simply no where else to go. You can try to get more points, but I think by the time you get to the end, you won't care.

All things considered, *Renegade* is an average arcade game. I can take it or leave it. Take a look and decide for yourself.

WORLD KARATE CHAMPIONSHIP



EPYX, 600 GALVESTON DRIVE
P.O. BOX 8020, REDWOOD CITY, CA 94063
(415) 366-0606
$19.95; COLOR ONLY

REVIEWED BY STEVE PANAK

After a temporary absence from the software shelves (due to a copyright infringement lawsuit), Epyx has re-released the award-winning *World Karate Championship*. And while this is not an event guaranteed to stop everyone in their tracks, it should inspire many a budding black belt to don his or her ghia and bow to this honorable opponent.

As you might expect, the basic premise of *Karate* is simple and straightforward. One or two players fight using the ancient martial arts. Experienced players will become the masters of moves ranging from a simple punch to flips and back-spin kicks. And although the game is at its best when it sets two friends at each other's throats, the single player will find the computer to be a worthy opponent. Perhaps I should say a relentless and ruthless opponent. It will take you quite a while to score any meaningful victories. Luckily, a notable feature allows two players to gang up on a single foe. But regardless of the skill of your adversary, all comers will find something to interest them.

The graphics are superb, the sound bone-crunchingly realistic, the action fast and furious. The little extras, such as the stars spinning around the head of a prone casualty or the realistic manner in which an unconscious body hits the ground, are what distinguish *Karate* from all the other fight games out there.

While the bulk of the screen forms the fighting arena, the top of the display contains scoring and vital statistics. A referee awards each hit or knockdown, the number of points based on the difficulty and viciousness of the blow. Between each round are tests of skill which must be completed before you are awarded advanced belts. You may be required to avoid thrown knives or break bricks with your head.

To keeps things visually interesting, the backdrops for the arena are scenes from around the world. You will find yourself fighting in places as gritty as the streets of New York or as exotic as a desert island, complete with hula-dancing spectators.

All in all, the visuals are excellent and program design good. Unfortunately, the game does have its drawbacks. The worst aspect of the game is the long time it takes to become proficient at it. This is due chiefly to the seemingly random way the control of the many possible moves is arranged about the joystick. Even with all the moves fully documented in the manual, it's too hard. It seems a better arrangement would be to have all the movement options, including jump and crouch, to be active when the button is released. Offensive moves would require the button to be depressed and would be arranged logically, perhaps with the two up diagonals activating high punches and kicks, left and right controlling center kicks and punches and so forth. Without a discernible pattern to aid memory, learning becomes a nightmare.

Despite this obstacle to play, however, those who stick with it will find *World Karate Championship* to be an above average game with a lot to offer. Whether you play it alone or with or against a friend, it's a knockout.

(CONTINUED FROM PAGE 33)

ST or older BASICs: It allows optional line numbers and line labels. The choice of using numbered or unnumbered code is up to the user.

Following the Microsoft standard is a good move, since it brings us closer to compatibility with the PC world. However, HSB is a hybrid and compatibility is only partial. HSB adds some commands, such as REPEAT/END REPEAT and RESET, not found in QuickBASIC, while ignoring others like DRAW, DECLARE and KEY (in some cases these are available as different commands or through one of the GEM calls). HSB's command set may prove a trifle disconcerting to ST users more familiar with other BASIC commands and syntax, but the changeover should not be difficult.

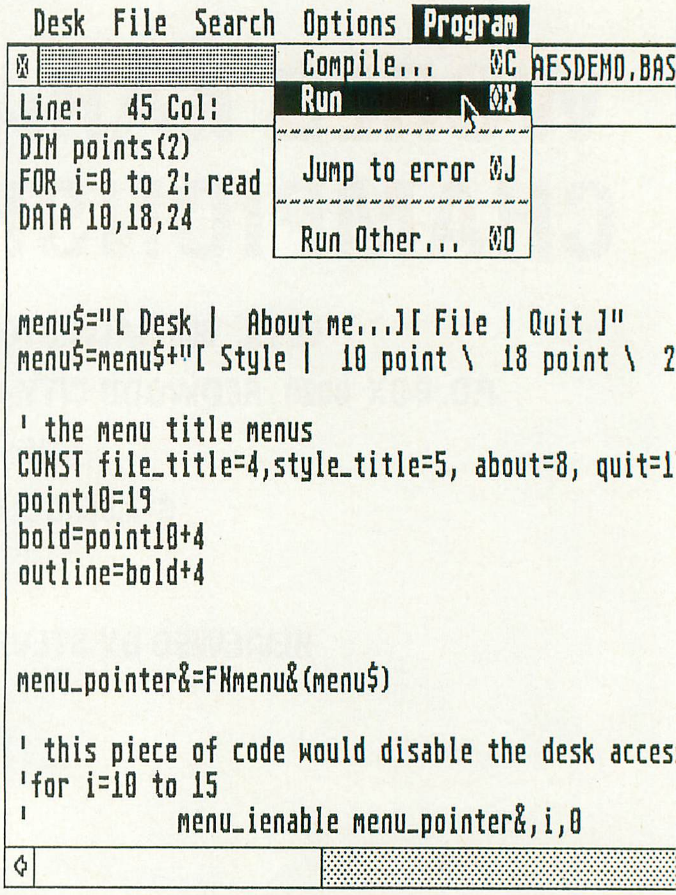
The big question is: Is it worth the effort to move to a new BASIC, especially one that currently has no large base of support on this continent? Should you stick with GFA or should you move up to C?

Professional programmers have disdain for BASIC, although it's quite possible—often even easier— to write commercial applications with a compiled BASIC as long as the result works at a reasonable speed. Professionals tend to use C for its portability, power and wide acceptance across a variety of platforms, such as Unix and MS-DOS. It's mostly novices and aging hackers like myself who choose BASIC.

Most C packages come with better documentation, resource construction kits, utilities and examples, not to mention an enormous local user base for support, literature and code. By now, GFA BASIC has enough of a user base here to offer considerable user support too. HSB has none.

HSB is good and could well have taken the market by storm had it been released at the same time as GFA. But the package flaws and the sparsity of support materials make me hesitate to recommend it strongly, unless you're moving directly from the PC QuickBASIC language and can afford to buy all the books on GEM you'll need.

If you're unsure, I'd suggest you wait and see if MichTron releases a better manual and perhaps some support materials first.



```
DIM points(2)
FOR i=0 to 2: read
DATA 10,18,24


menu$="[ Desk |  About me...][ File | Quit ]"
menu$=menu$+"[ Style |  10 point \  18 point \  2

' the menu title menus
CONST file_title=4,style_title=5, about=8, quit=1
point10=19
bold=point10+4
outline=bold+4


menu_pointer&=FNmenu&(menu$)

' this piece of code would disable the desk acces
'for i=10 to 15
'           menu_ienable menu_pointer&,i,0
```

Entry level vs. Professional level

Although there is a considerable price difference between the two versions of HiSoft BASIC, there really isn't much difference as far as the average programmer is concerned. Certainly not enough to justify $80 worth in this reviewer's mind.

Professional level lets you write your own libraries in assembly language. Somehow, I doubt that assembly language users accomplished enough to write libraries will be using anyone's BASIC.

Professional level has a stand-alone compiler with a command line interface environment. On the PC this is necessary, but on the ST it's peripheral.

The Professional level manual describes conversions from ST BASIC Version 2, has an appendix on assembly language details and memory formats and an ASCII character table.

Professional level has a profile program to help determine where your program spends most of its time. While interesting, this is of minimal use to the casual user.

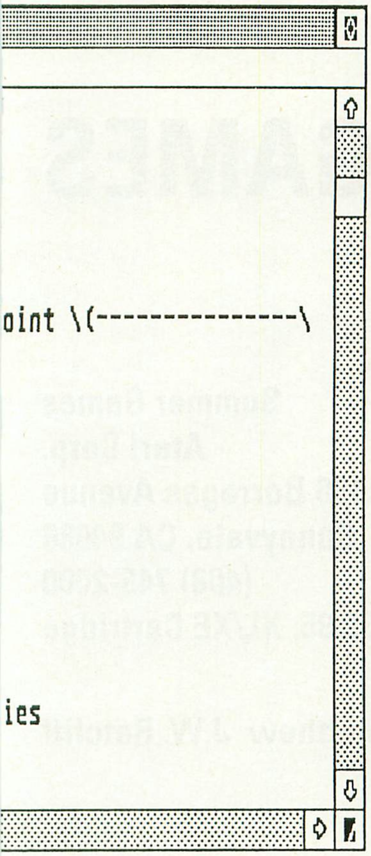
The most important difference is that Professional level can compile code directly into desk accessories. Unfortunately, the appendix describing this powerful feature is, like the rest of the manual, inadequate and contains mistakes. Beginners will not be able to understand the process involved, let alone write an actual accessory, from what little description they give you.

It's hard to see how these features are worth twice the price. The actual program code is some 3K larger in the Professional level, but whether this indicates hidden features or simply improvements and bug fixes, isn't documented.

The Documentation

Like many MichTron products, HiSoft BASIC (HSB) is compromised by the quality of the documentation, which is wholly inadequate for the product. The entry- level version even lacks an index. Without an index, the manual is next to useless. It has a fair number of typos and errors, some serious. For example, it mistakenly labels the VDI call vqf_attributes as vql_attributes (p.305).

Both versions' manuals exhort you to purchase the ST BASIC, Microsoft Quick-



BASIC and Borland's Turbo BASIC handbooks to get the information and explanations you need to use HSB! This astounding suggestion serves only to further point out the weaknesses of the HSB documentation.

The Professional-level manual is somewhat better, with a marginal (but still woefully incomplete) index and many (but not all) of the typos and errors in the junior version corrected. On the other hand, it fails to name chapters and appendices in the table of contents. I don't understand why MichTron has such a difficult time producing documentation that is at least equal to the product.

Worse is the manner in which the GEM and BIOS material is presented. The descriptions are terse, incomplete and opaque, with few examples to ease the confusion. Anyone not intimately familiar with the topic will find it almost impossible to use most of the functions presented here without yet more support books (I heartily recommend Sheldon Leemon's *ST Trilogy*, from Compute! Books). You'll also need a resource construction set (supplied with later versions of GFA BASIC). The GEM and BIOS calls are the real heart of the language and provide complete control over the ST's features. My own opinion is that anyone already familiar enough with these (developers or professional programmers) is currently using a language such as C, so what's the impetus to change to BASIC?

Some areas of the manuals are simply dismal. For example, you are recommended to use an AES constants header file, supplied on disk, but you are only told that these constants are most of those in the original Digital Research documentation; you are not told what they are, how they are used or what's missing. Would that HiSoft had not merely copied the language standard but had also taken the manuals Microsoft provides with QuickBASIC.

Newcomers will find themselves forced to do heavy hack-and-slash experimentation in order to figure out what the language can do and can't do. The examples on the disk do not begin to cover what the program can do, especially in the critical area of GEM. In fairness to buyers, MichTron should redo the manual, add tutorials, examples, a proper index and table of contents, then provide it, free of charge, with suitably abject apologies.

The Classic Sieve of Eratosthenes Benchmark: Two versions

Program 1a: GFA BASIC 2.0

```
Dim Flags(8190)
Cls
Print "Sieve - 25 iterations"
X=Timer
For Iter=1 To 25
  Count=0
  For I=0 To 8190
    Flags(I)=1
  Next I
  For I=0 To 8190
    If Flags(I) Then
      Prime=I+I+3
      K=I+Prime
      While K<=8190
        Flags(K)=0
        K=K+Prime
      Wend
      Count=Count+1
    Endif
  Next I
Next Iter
Xx=Timer
z=(xx-x)/200
Print count;" primes in "; z;" seconds"
end
```

Result:
interpreted: 394.16 seconds
compiled: 122.75 seconds
file size: 6916 bytes

Program 1b: HiSoft BASIC

```
DEFINT A-Z
DIM Flags(8190)
CLS
PRINT "Sieve - 25 iterations"
X# = TIMER
FOR Iter = 1 TO 25
  count = 0
  FOR I = 0 TO 8190
    Flags(I) = 1
  NEXT I
  FOR I = 0 TO 8190
    IF Flags(I) THEN
      Prime = I + I + 3
      K = I + Prime
      WHILE K <= 8190
     Flags(K) = 0
     K = K + Prime
      WEND
      count = count + 1
    END IF
  NEXT I
NEXT Iter
XX# = TIMER
PRINT USING "#### primes in ##.### seconds"; count; XX# - X#
END
```

This version, from Turbo BASIC, ran unmodified in HiSoft BASIC and QuickBASIC when loaded as a text file.
Result:
run from program:
57.33 seconds
compiled: 57.24 seconds
file size: 12588 bytes
file size: 32048

Program 2a: GFA BASIC 2.0

```
Print "BYTE SIEVE, 7000 numbers done 5 times"
T=Timer
Size=7000
Dim Flags(7001)
For J=1 To 5
  Acount=0
  For I=0 To Size
    Flags(I)=1
  Next I
  For I=0 To Size
    If Flags(I)=0 Then
      Goto 170
    Else
      Prime=I+I+3
    Endif
    K=I+Prime
    120:
    If K>Size Then
      Goto 160
    Else
      Flags(K)=0
    Endif
    K=K+Prime
    Goto 120
    160:
    Acount=Acount+1
    170:
  Next I
Next J
X=Timer
Y=(X-T)/200
Print Acount;" primes found"
Print "Took ";Y;" seconds"
```

Result:
interpreted: 73.88 seconds
compiled: 22.46 seconds
file size: 6893 bytes

Program 2b: HiSoft BASIC

```
rem next line turns overflow,
rem array checks and line numbers off
rem $option a-o-n-
defint a-z
10 PRINT "BYTE SIEVE, 7000 numbers done 5 times"
 t!=timer
20 SIZE=7000
30 DIM FLAGS(7001)
   for j=1 to 5
40 ACOUNT=0
50 FOR I=0 TO SIZE
60 FLAGS(I)=1
70 NEXT I
80 FOR I=0 TO SIZE
90 IF FLAGS(I)=0 THEN GOTO 170
100 PRIME=I+I+3
110 K=I+PRIME
120 IF K>SIZE THEN GOTO 160
130 FLAGS(K)=0
140 K=K+PRIME
150 GOTO 120
160 ACOUNT=ACOUNT+1
170 NEXT I
    next j
t!=timer-t!
180 PRINT ACOUNT;"primes found"
print "Took" t!/5 "seconds"
```

This version also ran, unmodified, in QuickBASIC.
Result:
run from program: .87 seconds
compiled: .87 seconds
file size: 10812 bytes

SUMMER GAMES

Summer Games
Atari Corp.
1196 Borregas Avenue
Sunnyvale, CA 94086
(408) 745-2000
$39.95; XL/XE Cartridge

Reviewed by Matthew J.W. Ratcliff

***Summer Games* from Epyx** will take you back to the summer of 1984 to compete in eight different simulations of platform diving, pole vaulting, gymnastics, freestyle relay and 100-meter freestyle swimming, 4x400-meter relay, 100-meter-dash-running and skeet shooting. The graphics are fairly good, but all the characters are two-tone blocky animations. This is a valiant effort to simulate the grandeur and excitement of the Olympic Games, but some events are little more than animations with you as a spectator.

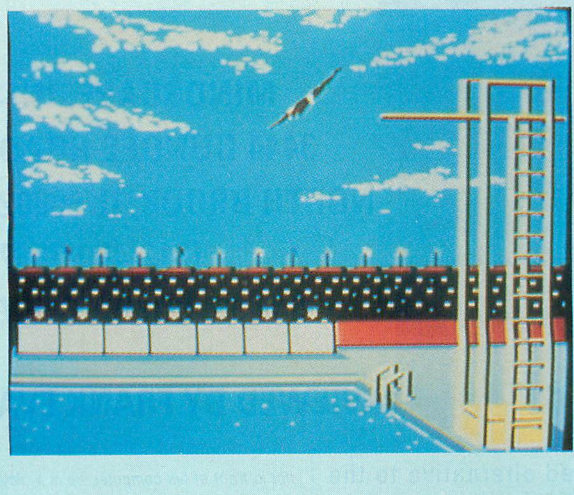
You will employ two basic joystick functions while participating in the *Summer Games*. Most events require precise movements of the stick or actuation of the fire button in specific sequences. Minute variations mean complete failure. The 100-yard dash requires brute-force toggling of the joystick at a ferocious rate to generate speed. The longer events require constant manipulation of the joystick by rote, which quickly becomes tedious.

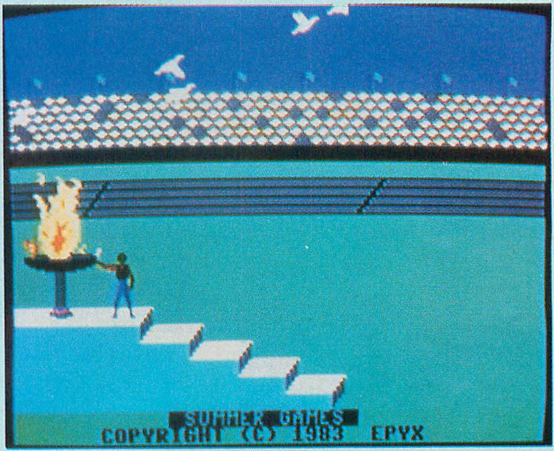
Players may compete in all eight events or a single one, taking turns in all except the 100-meter dash and freestyle swimming, where two can compete head to head. When starting a competition, up to three players may select names and a country to represent. There are flags and national anthems presented for up to nine different countries. If you prefer, you may select any single event for practice to hone your skills for the next big contest.

In the pole-vault event, you are presented with a side view of the track, vault and pit. You select bar height and grip position on the pole. On the approach, you pull the joystick down to plant the vaulting pole. This timing must be exact for a good launch. This is followed by a rapid succession of joystick gyrations and a delicately timed release of the pole. If you let go too soon, your man hits the bar; too late, and the pole will hit the bar. The timing sequence is extremely difficult to master.

Platform diving is my favorite of the *Summer Games* simulations. In competition, each player must perform a forward, backward, reverse and inward dive for a score based on quality and difficulty factor. The joystick directions may be used to control tuck, pike, half-pike and full-layout positions. A high score is awarded for many spins and a vertical entry, either hands or feet first. Once in the air, you select the spin rate by choice of position and finally move to a full-layout position for a hopefully vertical entry. A score is calculated based on difficulty factor and total points awarded by seven judges. Joystick controls for this simulation are logical and easily timed, and the graphic animations of the spinning diver are well done.

The 4x400-meter relay is designed to be a race of strategy. Each runner starts at a normal running speed, with a small horizontal bar at the bottom of the display representing his "wind." The longer the bar, the stronger the runner. Careful control of the joystick among normal





running, coasting and sprinting is required to optimize each runner's performance. Each has a different strength. The runner of the final leg is the best sprinter, for example. This is an accurate simulation of a genuine relay race, but the joystick inputs don't seem to have much control over the final outcome.

The 100-meter dash will test the limits of your joystick handle and your wrist. Wiggle the joystick horizontally or vertically as fast as possible. The faster the stick moves, the faster your character runs.

In the gymnastics simulation, you will attempt a vault from a springboard to the horse and then to the floor. For a high score you must spin in all sorts of ways and land flat-footed. This event requires painfully accurate timing. To get a good vault off the horse, the fire button must be pressed at the precise moment. Miss it, and you can forget about getting any rotation--or much of a score for that matter. I managed to hit it exactly once out of at least 50 attempts. This simulation is just plain frustrating.

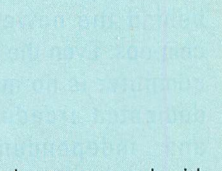
In the freestyle relay you must swim four laps, changing swimmers at the end of each lap. The 100-meter freestyle is identical, except with a single swimmer and only one lap. In these events you press the fire button in time with the swimmer's stroke for a minuscule boost of power. At each turn, change the swimmer's direction with the joystick at just the right moment for a fast turn. Pressing the fire button to the precisely timed metronome of the swimmer's stroke is tiresome and seems to make little difference. In fact, if you do nothing at all in the 100-meter freestyle, the swimmer will complete the race all by himself.

Skeet shooting is the final event of the competition. You control a circular gun sight, which indicates where the buckshot will hit when the trigger is pressed. You must react quickly and accurately as each target is launched. If you waver even slightly from a perfect beeline to the clay pigeon's path, there will not be enough time to catch up and shoot.

The opening and champion ceremonies, which round out the *Summer Games* package, are nicely done animations. This game is too neatly "packaged," however. The events are precisely sequenced with little or no variation allowed. You will spend more time simply watching than actively participating.

Summer Games may prove entertaining while the graphics and action sequences are new to you. After that, however, it degrades rapidly into a boring and tedious collection of monotonous simulations.

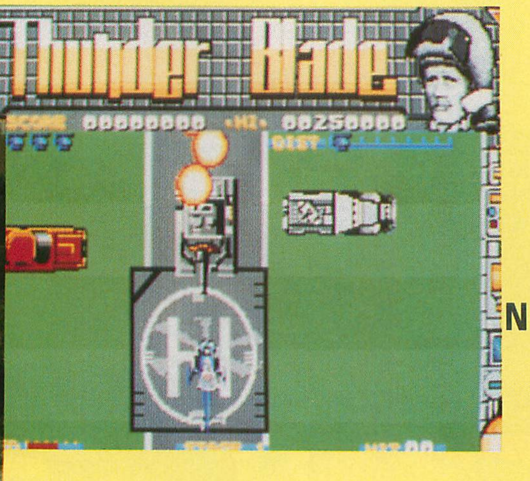
The author wishes to thank Randall's Home Computers of St. Louis, Missouri, for their valuable assistance in preparing this review.



Matthew J.W. Ratcliff *is an electrical engineer at McDonnell Aircraft in St. Louis.*

ST

THUNDER BLADE



MINDSCAPE, INC
3444 DUNDEE ROAD
NORTH BROOK, IL 60062
(312)480-7667
$49.95; COLOR ONLY

REVIEWED BY FRANK EVA

***Thunder Blade*, a mixed** bag of goodies and compromises, has proven that home computers and software will always be a giant step behind the newest arcade coin-ops. Even the best home computer is no match for a dedicated arcade machine, and independent home-computer software programmers may be no match for the in-house specialists at companies like Sega.

For those of you who may never have experienced the original coin-op, *Thunder Blade* combines vertical and full-perspective scrolling in a helicopter attack scenario. Other fabulous features include 3-D movements of the chopper and other vehicles in many of the screens

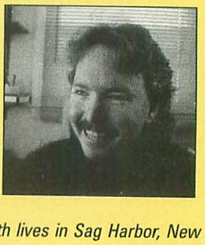
As for the graphic rendition of ST *Thunder Blade*, while a lot of the detail in the original game had to be forfeited, much remains that makes ST *Thunder Blade* look impressive. However, the wasteful title bar hides much of the potential field of play, so much so that shots from opposing helicopters or tanks are frequently seen before the vehicles themselves.

Furthermore, changes have been made in the scenarios. In the original coin-op, it wasn't possible to reverse the vertical scroll, but in the ST version, it is. When a naval vessel must be attacked, the ability to reverse directions actually becomes a hindrance, for no matter how hard you try to keep up your air speed, you seem just able to crawl forward. That's probably more realistic, considering that the ship may be under full-forward power, but it's not the way the original coin-op played.

The compromises made in the joystick mode of play leaves a less-than-desirable configuration. In order to increase the speed of the chopper, the player must use a conflicting stick routine that involves holding down the trigger button while pushing the stick forward. Since the trigger button also fires the weapons systems, the potential for frustration is great. Until the home-computer industry recognizes the need for a reasonably priced multi-buttoned alternative to the venerable old joystick, home versions of such high-powered arcade coin-ops will suffer.

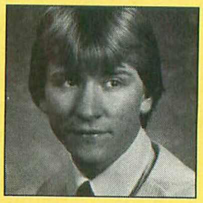
Thank goodness the programmers decided to include a mouse routine! As in *Starglider*, the mouse be-comes the tool of choice. The two independent buttons keep speed control separate from firing control. Control of the chopper's movements is intuitive and smoothly executed.

In conclusion, ST *Thunder Blade* is a disappointment, considering the compromises that were made in converting the original coin-op. But ST owners who purchase *Thunder Blade* will be pleased to know they have a product that is superior to the one for the other systems.



Peter Smith lives in Sag Harbor, New York, with his in-house editor, Julie. When not in front of his computer, he is a bicycle enthusiast, a struggling musician and a poet. He can be reached on GEnie at the address CAPT.COOK.

In addition to being fascinated by computers, Frank Eva is an avid Star Trek *fan. He has been interested in computer games ever since the release of* Pong *and has written several text adventures, as well as some educational-game software*



Steve Panak has written more game reviews for Analog Computing *than anyone on the face of the earth. He lives in Ohio where he plays games on his ST and, with the time remaining, practices law.*

RIO DATEL COMPUTERS

800-782-9110

*ORDERS ONLY MON-SAT 8AM-6PM (PACIFIC TIME)

CUST SERVICE
TECH SUPPORT
702-454-7700
TU-SA 10AM-6PM

AUTHORIZED ATARI SERVICE CENTER

COMPLETE LINE OF HARDWARE & SOFTWARE IN STOCK

FAX: 702-454-7700

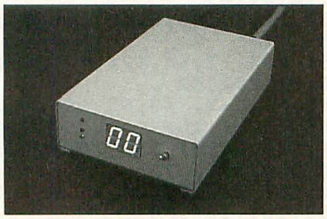
GENISCAN ST

*Easy to handle Scanner featuring 4 1/8" scanning width & 400dpi resolution enables you to reproduce graphics & text on your computer screen *Optical Character Recognition Mode will capture text images from the page & save them in format ready for conversion to text file suitable for word processor *Package includes GS-4000 scanner, interface & Scan Edit software *Powerful partner to Desk Top Publishing *Printout for Epson compatibles *Unmatched range of edit/capture facilities simply not offered by other scanners at this unbeatable price *Adjustable switches for brightness & contrast *Cut & paste editing of images *Save images as IMG, Degas, Neo, etc - **ONLY $299.95**

OCR SOFTWARE - *Teach your Atari to read !*

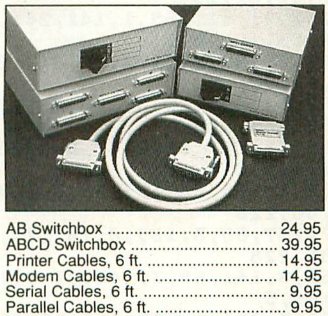
*Optical character recognition software converts scanned text to ascii*Use converted text in your word processor*No more manual transcribing of text*Uses special scanning mode of the Geniscan ST*Learn mode lets you teach your computer any font - **ONLY $89.95**

TRACKMASTER ST



*Trackmaster track display unit is a must for the disk hacker*Simply plugs into disk drive port of the ST*Displays track number, disk side and read/write status as the head moves along the dlsk*Two-digit LED display*Displays up to track 85*Works on both internal and external drives (switchable)*Complete hardware solution - no software required - **ONLY $59.95**

ACCESSORIES



AB Switchbox	24.95
ABCD Switchbox	39.95
Printer Cables, 6 ft.	14.95
Modem Cables, 6 ft.	14.95
Serial Cables, 6 ft.	9.95
Parallel Cables, 6 ft.	9.95
Gender Changer,M/M,F/F	4.95
Null Modem Adaptor	4.95
RS232 Surge Protector	9.95
RS232 Wiring Box	14.95
Diskette Holder,3.5",50 disk capacity	6.95
6 Outlet Surge Protector,Wall Mount	9.95
Bulk 3.5" Disks, 50 Pack	39.50
Printer Ribbons, (Star,Panasonic)	4.95
3.5" Head Cleaning Kit	4.95

EPROM BOARD

*Complete Eprom Board & case *Accepts up to 4 X 27256 Eproms *Mapped into the cartridge area - **ONLY $24.95**

SAMPLER STUDIO

*A complete sound sampling package *100% machine code for real-time functions *Saved files are compatible with other sampling systems *Sequencer for playback via keyboard or MIDI input - **ONLY $99.95**

ADVANCED ART STUDIO

*All the facilities needed for computer artwork, sprite & game design in one powerful but simple to use system *Graphic tools for home & professional use *Animated sprite designer & world map creator *Icon driven toolbox with on-screen help *16 color lo res mode operation - **ONLY $29.95**

SKYPLOT

*Most advanced features of any astronomy prg. *Covers 17,000 stars, planets & nebular objects *On line help *Zoom display thru 12 orders of magnitude *Several star chart formats supported *3D views *Search command *Compute Eclipse, transits, & conjunctions *supports med & hi res display *requires 1 meg mem - **ONLY$49.95**

ST COPY DEEP SCAN BURST NIBBLER

Copy even highly protected programs in under 60 seconds!!
Copy MS DOS protected programs with equal ease !!

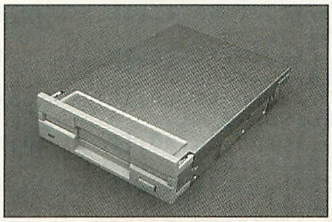
*Works with one drive or two*Copy between 5.25" and 3.5" drives*Multiple copy option allows many backups from one original *Copy 1 or 2 sides up to 85 tracks *Special format parameters for non standard formats *Unique "INFO ANALYSER" displays vital disc parameters including sector distribution, dispersion, interleave, etc *Easy to use Icon driven program takes the mystery out of Disc backup - **ONLY $39.95 Upgrades ONLY $12.00**

3.5" & 5.25" EXTERNAL DRIVES

3.5" - *Slimline extra low profile unit *Top quality drive mechanism*Fully compatible - 1 meg unformatted capacity*Can be formatted to 85 tracks and 11 sectors - **ONLY $159.95**

5.25"- *Operate in 40 or 80 track mode*Rear panel switches to configure ALL combinations of computer model and track modes*Perfect for use with PC Ditto- **ONLY $179.95**

1 MEG INTERNAL DRIVE UPGRADE



*Replace internal 500K drive with a full 1 Meg unit*Top quality drive unit- Easily fitted- Full instructions*Direct plug in replacement*No special skills or tools required*Minor modification to enclosure is required*When considering a drive replacement remember that quality is most important - **ONLY $119.95**

REPLACEMENT MOUSE

Twice the tracking speed of the Atari mouse !

*No more mouse doublers required*Softer more positive button clicking*More consistent double click*Extra long 6 foot connecting cord*highest quality mechanical parts for for longer life and consistent positioning - **ONLY $44.95**

ST SUPERTOOLKIT

*Whole disk Track & Sector Edit *Read up to 85 tracks and 255 sectors! *Super HiRes display works with Mono monitor only *3 different editor modes-Drive, Disk or File *Direct select function for Bootsector, Directory sector *Fully Menu/Icon driven *HEX / ACSII edit modes directly on the disk *Perfect for the disk hacker - **ONLY $24.95**

ST TIMEKEEPER

*Simple plug-in battery-backed calendar clock cartridge*Fits into cartridge port*On-board clock chip*Lithium battery backed to last up to 5 years*Automatically adjusts for leap year etc*Complete with install disk and corner clock display*Exact time, date etc available on power up - **ONLY $49.95**

PC DITTO II - $249.95
PC DITTO II + 5.25" DRIVE - **$419.95**
PC DITTO II + 5.25" DRIVE + DRIVE SWITCH - **$451.95**

AUTHORIZED ATARI SERVICE CENTER

LET US REPAIR YOUR EQUIPMENT - 5 DAY REPAIR OR THE SHIPPING IS FREE - $ CALL

ALL THE LATEST EUROPEAN SOFTWARE

CALL FOR PRICE AND AVAILABILITY

AERCO MEMORY BOARD UPGRADES

ABSOLUTELY THE EASIEST INSTALLATION - UNPOPULATED BOARD - $139.95
* 1/2 MEG RAM - $80.00 * 2MEG RAM - $240.00 * 4MEG RAM - $480.00

COMPLETE LINE OF MIDI PRODUCTS

MANY OTHER ITEMS IN STOCK - IF YOU DON'T SEE WHAT YOU WANT CALL FOR LOW, LOW PRICES

RIO / DATEL
3430 E. TROPICANA AVE. #67
LAS VEGAS, NV 89121

*Add $4.00 shipping/handling in the continental U.S.: $7.00 - PR, AK, HI, FPO, APO: $10.00-Canada/Mex: C.O.D. orders add $3.00 to above charges: SPECIFIY COMPUTER MODEL WITH ORDER: VISA/MC/Checks/Money Orders/C.O.D. Accepted: Please call for return authorization number - returns subject to a 20% restocking fee: We reserve the right to substitute products of different appearance but equal quality and function for items pictured above: Prices subject to change without notice.

800-782-9110
702-454-7700
IN NEVADA

CIRCLE #112 ON READER SERVICE CARD.

(CONTINUED FORM PAGE 7)

LISTING 1: BASIC

```
HK 10 OPEN #1,8,0,"D:DIR3.OBJ"
RF 20 FOR X=1 TO 1454:READ A:PUT #1,A:NEX
   T X
XW 30 CLOSE #1:END
NE 1000 DATA 255,255,0,50,28,50,32,32,32,
   32,32,32,32,32,32,32
GZ 1010 DATA 68,105,114,101,99,116,111,11
   4,121,58,32,68,49,58,42,46
RN 1020 DATA 42,155,27,69,50,64,51,68,114
   ,105,118,101,32,91,49,93
TM 1030 DATA 32,63,32,27,83,101,97,114,99
   ,104,115,112,101,99,32,63
EY 1040 DATA 32,27,67,97,110,110,111,116,
   32,111,112,101,110,32,68,73
TU 1050 DATA 82,33,155,27,155,155,68,73,8
   2,51,44,32,98,121,32,77
OU 1060 DATA 97,116,42,82,97,116,155,27,6
   8,73,82,51,32,98,121,32
KS 1070 DATA 77,97,116,42,82,97,116,44,32
   ,40,99,41,32,65,78,65
DJ 1080 DATA 76,79,71,155,80,114,101,115,
   115,32,91,69,83,67,93,32
FU 1090 DATA 116,119,105,99,101,32,38,32,
   91,82,84,78,93,32,116,111
OD 1100 DATA 32,101,120,105,116,46,155,27
   ,43,45,45,45,45,45,45,45
FE 1110 DATA 45,45,45,45,43,45,45,45,45,4
   5,45,45,45,45,45,45
BK 1120 DATA 43,45,45,45,45,45,45,45,45,4
   5,45,45,43,155,27,32
ZR 1130 DATA 32,32,32,32,32,32,32,32,32,1
   7,18,18,18,18,18,18
FA 1140 DATA 18,18,18,18,18,23,18,18,18,1
   8,18,18,18,18,18,18
DA 1150 DATA 18,23,18,18,18,18,18,18,18,1
   8,18,18,18,5,155,27
DJ 1160 DATA 26,18,18,18,18,18,18,18,18,1
   8,18,18,24,18,18,18
FJ 1170 DATA 18,18,18,18,18,18,18,18,24,1
   8,18,18,18,18,18,18
JE 1180 DATA 18,18,18,65,51,120,51,18,3,1
   55,27,32,32,32,32,32
BQ 1190 DATA 32,32,32,32,32,32,32,32,32,3
   2,32,32,32,32,32,32
AR 1200 DATA 32,32,32,32,32,32,32,32,32,3
   2,32,32,32,32,32,32
HY 1210 DATA 32,32,32,32,32,32,32,32,32,3
   2,32,32,32,155,27,241
TL 1220 DATA 51,236,52,0,155,27,91,82,69,
   84,85,82,78,93,32,109
JK 1230 DATA 111,114,101,44,32,91,69,83,6
   7,93,32,101,120,105,116,32
KY 1240 DATA 63,27,156,27,0,155,253,80,11
   4,105,110,116,101,114,32,101
SW 1250 DATA 114,114,111,114,33,155,27,80
   ,114,105,110,116,111,117,116,32
PD 1260 DATA 100,105,114,32,63,32,40,89,4
   7,78,41,32,27,169,0,141
DZ 1270 DATA 18,52,141,176,55,141,241,51,
   169,0,32,85,55,169,21,162
JD 1280 DATA 50,32,129,55,169,134,162,50,
   32,129,55,169,69,162,50,32
FH 1290 DATA 129,55,32,241,54,201,27,208,
   3,76,37,54,201,155,240,14
BD 1300 DATA 201,49,144,224,201,57,176,22
   0,141,22,50,141,76,50,169,82
BZ 1310 DATA 162,50,32,129,55,173,24,50,7
   2,169,50,162,24,160,40,32
RM 1320 DATA 50,55,173,24,50,201,155,208,
   7,104,141,24,50,76,159,52
BI 1330 DATA 104,162,0,189,24,50,201,155,
   240,3,232,208,246,232,169,27
QQ 1340 DATA 157,24,50,169,37,162,52,32,1
   29,55,32,241,54,201,89,240
OJ 1350 DATA 6,201,121,240,2,169,0,141,17
   6,55,32,72,55,162,16,169
BG 1360 DATA 21,157,68,3,169,50,157,69,3,
   169,0,157,72,3,157,73
GO 1370 DATA 3,157,75,3,169,6,157,74,3,16
   9,3,157,66,3,32,237
LU 1380 DATA 52,232,53,86,228,152,16,17,1
   69,21,162,50,32,129,55,169
TC 1390 DATA 96,162,50,32,129,55,76,82,52
   ,169,242,162,51,32,129,55
MT 1400 DATA 169,0,162,50,32,129,55,169,2
   47,162,50,32,129,55,173,176
DJ 1410 DATA 55,240,21,169,242,162,51,32,
   179,55,169,0,162,50,32,179
KL 1420 DATA 55,169,198,162,50,32,179,55,
   160,0,162,0,32,76,54,173
AU 1430 DATA 241,51,240,3,76,202,53,189,6
   9,51,201,42,208,8,189,70
RI 1440 DATA 51,9,128,157,70,51,169,124,1
   57,69,51,138,24,105,12,170
CZ 1450 DATA 173,241,51,240,3,76,202,53,2
   00,192,3,208,207,169,124,157
OY 1460 DATA 69,51,232,169,155,157,69,51,
   169,27,232,157,69,51,238,18
SD 1470 DATA 52,169,69,162,51,32,129,55,1
   73,176,55,240,10,169,69,162
AA 1480 DATA 51,32,179,55,76,50,53,173,18
   ,52,201,22,144,154,169,244
NT 1490 DATA 162,51,32,129,55,169,0,141,1
   8,52,32,1,55,201,155,208
FP 1500 DATA 10,169,16,162,52,32,129,55,7
   6,50,53,201,27,208,235,169
EQ 1510 DATA 16,162,52,32,129,55,169,30,1
   62,51,32,129,55,76,37,54
WB 1520 DATA 138,24,105,13,170,169,155,20
   0,192,3,240,11,32,48,54,200
ES 1530 DATA 192,3,240,3,32,48,54,169,155
   ,157,69,51,232,169,27,233
NS 1540 DATA 53,228,54,157,69,51,169,69,1
   62,51,32,129,55,173,176,55
LH 1550 DATA 240,14,169,69,162,51,32,179,
   55,169,198,162,50,32,179,55
LO 1560 DATA 169,30,162,51,32,129,55,32,8
   ,56,169,244,162,51,32,129
JE 1570 DATA 55,32,1,55,201,155,208,3,76,
   59,52,201,27,208,242,32
MW 1580 DATA 8,56,169,114,162,50,32,129,5
   5,96,140,74,54,160,11,169
SF 1590 DATA 32,157,69,51,232,136,208,249
   ,169,124,157,69,51,232,172,74
PB 1600 DATA 54,96,0,0,0,0,141,72,54,142,
   73,54,140,74,54,189
HE 1610 DATA 68,51,141,75,54,162,16,169,6
   8,24,109,73,54,157,68,3
DM 1620 DATA 169,51,105,0,157,69,3,169,0,
   141,241,51,157,73,3,169
YJ 1630 DATA 40,157,72,3,169,5,157,66,3,3
   2,86,228,152,16,3,141
RV 1640 DATA 241,51,173,72,54,174,73,54,1
   72,74,54,189,68,51,201,32
PJ 1650 DATA 240,79,201,42,240,69,189,70,
   51,157,72,51,189,69,51,157
NC 1660 DATA 71,51,189,68,51,157,70,51,16
   9,124,157,69,51,169,32,157
AG 1670 DATA 73,51,169,70,157,74,51,169,1
   14,157,75,51,169,101,157,76
CK 1680 DATA 51,157,77,51,169,32,157,78,5
   1,157,79,51,157,80,51,169
YV 1690 DATA 124,157,81,51,169,1,141,241,
   51,208,6,189,68,51,157,229
VC 1700 DATA 54,224,55,69,51,173,75,54,15
   7,68,51,173,72,54,96,169
ST 1710 DATA 51,162,69,160,10,32,50,55,17
   3,69,51,96,75,58,155,162
LM 1720 DATA 32,169,12,157,66,3,32,86,228
   ,169,3,157,66,3,169,254
PJ 1730 DATA 157,68,3,169,54,157,69,3,169
   ,4,157,74,3,32,86,228
ED 1740 DATA 169,7,157,66,3,169,0,157,72,
   3,157,73,3,76,86,228
VS 1750 DATA 142,68,3,141,69,3,140,72,3,1
   62,0,142,73,3,169,5
SL 1760 DATA 141,66,3,76,86,228,162,16,16
   9,12,157,66,3,76,86,228
AF 1770 DATA 83,58,0,72,162,96,169,12,157
   ,66,3,32,86,228,162,96
ZT 1780 DATA 169,3,157,66,3,169,82,157,68
   ,3,169,55,157,69,3,104
RY 1790 DATA 157,75,3,41,240,73,16,9,12,1
   57,74,3,76,86,228,141
OH 1800 DATA 68,3,142,69,3,133,224,134,22
   5,160,0,140,73,3,177,224
WY 1810 DATA 201,27,240,10,200,208,247,23
   8,73,3,230,225,208,240,140,72
DK 1820 DATA 3,169,11,141,66,3,162,0,76,8
   6,228,80,58,155,0,0
```

```
JX 1830 DATA 0,142,177,55,141,178,55,162,
   80,169,12,157,66,3,32,86
HK 1840 DATA 228,169,3,157,66,3,169,8,157
   ,74,3,169,173,157,68,3
UQ 1850 DATA 169,55,157,69,3,32,86,228,15
   2,16,13,169,19,162,52,225
CL 1860 DATA 55,37,56,32,129,55,169,0,141
   ,176,55,96,173,177,55,157
UP 1870 DATA 69,3,173,178,55,157,68,3,169
   ,80,157,72,3,169,0,157
EB 1880 DATA 73,3,169,9,157,66,3,76,86,22
   8,162,16,169,12,157,66
OY 1890 DATA 3,32,86,228,162,32,169,12,15
   7,66,3,32,86,228,162,80
RF 1900 DATA 169,12,157,66,3,76,86,228,22
   4,2,225,2,59,52
```

LISTING 2: ASSEMBLY

```
0 *SAVE#D:DIR3.M65
10 *ASM,#-,#D:DIR3.COM
20 *-----------------------------*
30 * DIR3 - 3 across directory   *
40 * lister by Mat*Rat           *
50 * for Analog Computing (c)1989*
60 * from: Ratware Softworks     *
70 *        32 S. Hartnett Ave.  *
80 *        St. Louis, MO  63135 *
90 *-----------------------------*
0100      .ORG $3200
0110      .OPT OBJ
0120 * Important equates for exciting
0130 * things:
0140 ESC =     27
0150 EOL =     155
0160 *
0170 CIO =     $E456   ;CALL OS HERE
0180 ICCOM = $0342     ;COMMAND TO CIO
0190 ICBADR = $0344    ;BUFFER OR FNAME
0200 ICBLEN = $0348    ;BUFFER LENGTH
0210 ICAUX1 = $034A    ;AUX BYTE #1
0220 ICAUX2 = $034B    ;AUX BYTE #2
0230 *
0240 COPN =   3        ;COMMAND OPEN
0250 CCLOSE = 12       ;COMMAND CLOSE
0260 CGTXR = 5         ;GET TEXT REC
0270 CPTXR = 9         ;PUT TEXT REC
0280 CGBINR = 7        ;GET BINARY REC
0290 CPBINR = 11       ;PUT BINARY REC
0300 CDRAW = 17        ;COMMAND DRAWTO
0310 CFILL = 18        ;COMMAND FILL
0320 *
0330 OPIN =   4        ;OPEN FOR INPUT
0340 OPOUT = 8         ;OPEN FOR OUTPUT
0350 OPDIR = 6         ;OPEN FOR DIR
0360 *
0370 XCORD = $55
0380 YCORD = $54
0390 *
0400 *-----------------------------*
0410 * GL - Get a line of text     *
0420 * macro.  Places hi byte      *
0430 * of string addr in A reg,    *
0440 * low byte of string addr     *
0450 * in X reg, and max len       *
0460 * of string in Y reg          *
0470 * and then calls GETSTRING    *
0480 *-----------------------------*
0490      .MACRO GL
0500        .IF %0<>2
0510        .ERROR "GL error, 2 param"
0520        .ENDIF
0530     LDA # >%1
0540     LDX # <%1
0550        .IF %2<256
0560       LDY #%2
0570        .ELSE
0580       LDY %2
0590        .ENDIF
0600     JSR GETSTRING
0610      .ENDM
0620 *-----------------------------*
0630 * Graphics 0 macro            *
0640 * Execute the equivalent of   *
0650 * an Atari BASIC GRAPHICS 0   *
0660 * command                     *
0670 *-----------------------------*
0680      .MACRO GR0
0690        .IF %0<>0
0700        .ERROR "No param for GR0"
0710        .ENDIF
0720     LDA #0
0730     JSR GRAPHICS
0740      .ENDM
0750 *------------------------------
0760 * Fprint a string at the      *
0770 * X,Y position specified      *
0780 *------------------------------
0790      .MACRO FPXY
0800        .IF %0<>3
0810        .ERROR "FPXY-Param count"
0820        .ENDIF
0830        .IF %1<256
0840       LDA #%1
0850        .ELSE
0860       LDA %1
0870        .ENDIF
0880     STA XCORD
0890        .IF %2<256
0900       LDA #%2
0910        .ELSE
0920       LDA %2
0930        .ENDIF
0940     STA YCORD
0950     LDA # <%3
0960     LDX # >%3
0970     JSR FPRINT
0980      .ENDM
0990 *-----------------------------*
1000 * Fprint macro, no X & Y      *
1010 * specified,so use the current*
1020 * X,Y coordinates             *
1030 *-----------------------------*
1040      .MACRO FP
1050        .IF %0<>1
1060        .ERROR "FP-Param count"
1070        .ENDIF
1080     LDA # <%1
1090     LDX # >%1
1100     JSR FPRINT
1110      .ENDM
1120 *-----------------------------*
1130 * LP - Line print macro       *
1140 * print the text record       *
1150 * pointed to by A (low)       *
1160 * and X (high) registers      *
1170 * on the line printer         *
1180 * Disable printer output      *
1190 * if an error occurs          *
1200 *-----------------------------*
1210      .MACRO LP
1220        .IF %0<>1
1230        .ERROR "LP-Param count"
1240        .ENDIF
1250     LDA # <%1
1260     LDX # >%1
1270     JSR LPRINT
1280      .ENDM
1290 *-----------------------------*
1300 * Data work area              *
1310 *-----------------------------*
1320 DIRINFO .BYTE "             "
1330      .BYTE "Directory: "
1340 DIRSPEC .BYTE "D1:*.*",155,27
1350      .DS 40
1360 DRIVE .BYTE "Drive [1] ? ",27
1370 FILESPEC .BYTE "Searchspec ? "
1380      .BYTE 27
1390 CANTDO .BYTE "Cannot open DIR!"
1400      .BYTE 155,27
1410 RDIR3 .BYTE 155,155
1420      .BYTE "DIR3, by Mat*Rat"
1430      .BYTE 155,27
1440 ESCEXIT .BYTE "DIR3 by Mat*Rat,"
1450      .BYTE " (c) ANALOG",155
1460      .BYTE "Press [ESC]"
```

```
1470       .BYTE " twice & [RTN]"
1480       .BYTE " to exit.",155,27
1490 HBAR .BYTE "+-----------+"
1500       .BYTE "-----------+"
1510       .BYTE "------------+",155,27
1520       .BYTE "            "
1530 SHBAR .BYTE 17,18,18,18,18,18
1540       .BYTE 18,18,18,18,18,18,23
1550       .BYTE 18,18,18,18,18,18
1560       .BYTE 18,18,18,18,18,23
1570       .BYTE 18,18,18,18,18,18
1580       .BYTE 18,18,18,18,18,5
1590       .BYTE 155,27
1600 BHBAR .BYTE 26,18,18,18,18,18
1610       .BYTE 18,18,18,18,18,18,24
1620       .BYTE 18,18,18,18,18,18
1630       .BYTE 18,18,18,18,18,24
1640       .BYTE 18,18,18,18,18,18
1650       .BYTE 18,18,18,18,18,3
1660       .BYTE 155,27
1670 LINBUF .BYTE "            "
1680       .BYTE "            "
1690       .BYTE "            "
1700       .BYTE "            "
1710       .BYTE "            ",155,27
1720       .DS 120
1730 DONEFLG .BYTE 0
1740 LF   .BYTE 155,27
1750 HOLDIT .BYTE "[RETURN] more"
1760       .BYTE ", [ESC] exit ?",27
1770 RETURN .BYTE 156,27
1780 YCOUNT .BYTE 0
1790 NOPRN .BYTE 155,253,"Printer"
1800       .BYTE " error!",155,27
1810 PROUT .BYTE "Printout dir ?"
1820       .BYTE " (Y/N) ",27
1830 *------------------------------*
1840 * Startup the program:          *
1850 * Get user preferences for      *
1860 * drive, searchspec, and        *
1870 * printer output, then list     *
1880 * the directory.                *
1890 *------------------------------*
1900 STARTUP
1910       LDA #0
1920       STA YCOUNT
1930       STA PRCTL
1940       STA DONEFLG
1950        GR0         ; Clear screen
1960        FP   DIRSPEC
1970 DRVRQ
1980        FP   ESCEXIT ; Get drive
1990        FP   DRIVE   ; preference
2000       JSR GETLKEY
2010       CMP #ESC     ; ESC to exit
2020       BNE START1
2030       JMP DNEX
2040 START1
2050       CMP #EOL     ; RTN is
2060       BEQ DEFDRV   ; default drive
2070       CMP #'1
2080       BCC DRVRQ
2090       CMP #'9
2100       BCS DRVRQ    ; Get filespec
2110       STA DIRSPEC+1
2120       STA DRIVE+7  ; RTN is default
2130 DEFDRV
2140        FP   FILESPEC
2150       LDA DIRSPEC+3 ; Save default
2160       PHA
2170        GL   DIRSPEC+3,40
2180       LDA DIRSPEC+3 ; Return only?
2190       CMP #EOL
2200       BNE GOTFS
2210       PLA
2220       STA DIRSPEC+3 ; Keep default
2230       JMP GOTFS1
2240 GOTFS PLA
2250 GOTFS1
2260       LDX #0       ; We use ESC
2270 ADESC LDA DIRSPEC+3,X
2280       CMP #EOL     ; as end of
2290       BEQ PUTESC   ; line mark
2300       INX          ; for FPRINT
```

```
2310       BNE ADESC    ; Adjust it
2320 PUTESC INX         ; for search
2330       LDA #ESC     ; spec
2340       STA DIRSPEC+3,X
2350        FP   PROUT   ; Hard copy?
2360       JSR GETLKEY
2370       CMP #'Y      ; Y or y
2380       BEQ GOTPRN   ; must be
2390       CMP #'y      ; input
2400       BEQ GOTPRN   ; for yes
2410       LDA #0       ; all else NO
2420 GOTPRN STA PRCTL
2430       JSR CLOSE1
2440       LDX #$10
2450       LDA # <DIRSPEC ; Open up
2460       STA ICBADR,X ; the dir
2470       LDA # >DIRSPEC ; spec
2480       STA ICBADR+1,X ; on IOCB
2490       LDA #0       ; #1 for
2500       STA ICBLEN,X ; directory
2510       STA ICBLEN+1,X ; listing
2520       STA ICAUX2,X ; input
2530       LDA #OPDIR
2540       STA ICAUX1,X
2550       LDA #COPN
2560       STA ICCOM,X
2570       JSR CIO
2580       TYA
2590       BPL DODIR
2600        FP   DIRSPEC
2610        FP   CANTDO
2620       JMP DRVRQ
2630 DODIR
2640        FP   LF
2650        FP   DIRINFO ; Show dir
2660        FP   SHBAR   ; and
2670       LDA PRCTL    ; vert bar
2680       BEQ DODIR1   ; print?
2690        LP   LF
2700        LP   DIRINFO ; LPRINT too
2710        LP   HBAR
2720 DODIR1
2730       LDY #0       ; Files/line
2740       LDX #0
2750 DOLINE
2760       JSR GETFN    ; Get a filename
2770       LDA DONEFLG
2780       BEQ CNTDIR
2790       JMP DONEDIR
2800 CNTDIR LDA LINBUF,X
2810       CMP #'*      ; Locked?
2820       BNE DOL1
2830       LDA LINBUF+1,X
2840       ORA #$80     ; Inverse char
2850       STA LINBUF+1,X
2860 DOL1
2870       LDA #'|      ; Make name
2880       STA LINBUF,X ; divider
2890       TXA
2900       CLC
2910       ADC #12      ; Next field
2920       TAX
2930       LDA DONEFLG ; Last one done?
2940       BEQ DOL2     ; no, more files
2950       JMP DONEDIR ; yes, wrapup
2960 DOL2
2970       INY          ; next field
2980       CPY #3       ; 3rd one?
2990       BNE DOLINE   ; no, more
3000       LDA #'|      ; yes, fixup
3010       STA LINBUF,X ; field with
3020       INX          ; next file
3030       LDA #155     ; separator
3040       STA LINBUF,X ; and print
3050       LDA #27      ; out the line
3060       INX
3070       STA LINBUF,X
3080       INC YCOUNT
3090        FP   LINBUF
3100       LDA PRCTL    ; Lprint it if
3110       BEQ WATST    ; PRCTL flag set
3120        LP   LINBUF
3130       JMP DODIR1
3140 WATST LDA YCOUNT ; Scrolling?
```

```
3150      CMP #22
3160      BCC DODIR1
3170       FP  HOLDIT
3180      LDA #0
3190      STA YCOUNT
3200 NXLWAI ;       Yes, pause
3210      JSR GETKEY  ; for user
3220      CMP #EOL
3230      BNE NXCK
3240       FP  RETURN
3250      JMP DODIR1
3260 NXCK CMP #ESC
3270      BNE NXLWAI
3280       FP  RETURN
3290       FP  BHBAR
3300      JMP DNEX
3310 DONEDIR
3320      TXA            ; Done with
3330      CLC            ; DIR, now
3340      ADC #13        ; pad remaining
3350      TAX            ;fields so
3360      LDA #EOL       ; display not
3370      INY            ; 'ragged'
3380      CPY #3
3390      BEQ DONEDIR1
3400      JSR FILLINE
3410      INY
3420      CPY #3
3430      BEQ DONEDIR1
3440      JSR FILLINE
3450 DONEDIR1
3460      LDA #EOL
3470      STA LINBUF,X
3480      INX
3490      LDA #ESC
3500      STA LINBUF,X
3510       FP  LINBUF
3520      LDA PRCTL
3530      BEQ EXITNOW
3540       LP  LINBUF
3550       LP  HBAR
3560 EXITNOW
3570       FP  BHBAR
3580      JSR CLOSEALL
3590       FP  HOLDIT
3600 MORE JSR GETKEY
3610      CMP #EOL
3620      BNE CKEXI
3630      JMP STARTUP
3640 CKEXI CMP #ESC
3650      BNE MORE
3660 DNEX
3670      JSR CLOSEALL
3680       FP  RDIR3
3690      RTS
3700 *------------------------------*
3710 * Fill the next field with     *
3720 * blanks and a vertical bar.   *
3730 * This will prevent unsightly  *
3740 * 'ragged edge' at bottom      *
3750 * directory list               *
3760 *------------------------------*
3770 FILLINE
3780      STY SAVY
3790      LDY #11
3800      LDA #32
3810 FIL  STA LINBUF,X
3820      INX
3830      DEY
3840      BNE FIL
3850      LDA #'|
3860      STA LINBUF,X
3870      INX
3880      LDY SAVY
3890      RTS
3900 *------------------------------*
3910 SAVA .BYTE 0     ; Save registers
3920 SAVX .BYTE 0     ; for GETFN
3930 SAVY .BYTE 0     ; function
3940 LBSV .BYTE 0
```

```
3950 *------------------------------*
3960 * Get a filename from the      *
3970 * opened IOCB #1, for DIR.     *
3980 * Check for end of file and    *
3990 * set DONEFLG if necessary.    *
4000 *------------------------------*
4010 GETFN
4020      STA SAVA
4030      STX SAVX
4040      STY SAVY
4050      LDA LINBUF-1,X
4060      STA LBSV     ; Load in front
4070      LDX #$10     ; of linbuf
4080      LDA # <LINBUF-1
4090      CLC          ; we have to
4100      ADC SAVX     ; chop some
4110      STA ICBADR,X
4120      LDA # >LINBUF+1
4130      ADC #0       ; chars for
4140      STA ICBADR+1,X
4150      LDA #0       ; screen format
4160      STA DONEFLG
4170      STA ICBLEN+1,X
4180      LDA #40
4190      STA ICBLEN,X
4200      LDA #CGTXR
4210      STA ICCOM,X
4220      JSR CIO
4230      TYA
4240      BPL DNGET
4250      STA DONEFLG
4260 DNGET LDA SAVA
4270      LDX SAVX
4280      LDY SAVY
4290      LDA LINBUF-1,X
4300      CMP #32      ; Space? Not EOF
4310      BEQ FNGOT
4320      CMP #'*
4330      BEQ FNFIXUP
4340 * Must be a digit, end of dir
4350 * Patch up Free sectors field
4360 * so it fits in 12 char window
4370      LDA LINBUF+1,X
4380      STA LINBUF+3,X
4390      LDA LINBUF,X
4400      STA LINBUF+2,X
4410      LDA LINBUF-1,X
4420      STA LINBUF+1,X
4430      LDA #'|
4440      STA LINBUF,X
4450      LDA #32
4460      STA LINBUF+4,X
4470      LDA #'F
4480      STA LINBUF+5,X
4490      LDA #'r
4500      STA LINBUF+6,X
4510      LDA #'e
4520      STA LINBUF+7,X
4530      STA LINBUF+8,X
4540      LDA #32
4550      STA LINBUF+9,X
4560      STA LINBUF+10,X
4570      STA LINBUF+11,X
4580      LDA #'|
4590      STA LINBUF+12,X
4600      LDA #1
4610      STA DONEFLG
4620      BNE FNGOT
4630 FNFIXUP LDA LINBUF-1,X
4640      STA LINBUF,X
4650 FNGOT
4660      LDA LBSV
4670      STA LINBUF-1,X
4680      LDA SAVA
4690      RTS
4700 GETLKEY  GL  LINBUF,10
4710      LDA LINBUF
4720      RTS
```

DIR3

PROGRAM LISTING

```
4730 *----------------------------*
4740 * Get a key from the          *
4750 * keyboard through            *
4760 * the K: device 2             *
4770 * and return it in            *
4780 * the A register              *
4790 *----------------------------*
4800 KEY .BYTE "K:",155
4810 GETKEY
4820      LDX #$20
4830      LDA #CCLOSE
4840      STA ICCOM,X
4850      JSR CIO
4860      LDA #COPN
4870      STA ICCOM,X
4880      LDA # <KEY
4890      STA ICBADR,X
4900      LDA # >KEY
4910      STA ICBADR+1,X
4920      LDA #OPIN
4930      STA ICAUX1,X
4940      JSR CIO
4950      LDA #CGBINR
4960      STA ICCOM,X
4970      LDA #0
4980      STA ICBLEN,X
4990      STA ICBLEN+1,X
5000      JMP CIO
5010 *----------------------------*
5020 * Get a string from           *
5030 * the keyboard through the    *
5040 * E: device 0 and return it   *
5050 * in the A register           *
5060 *----------------------------*
5070 GETSTRING
5080      STX ICBADR  ; String addr
5090      STA ICBADR+1
5100      STY ICBLEN  ; Max length
5110      LDX #0
5120      STX ICBLEN+1
5130      LDA #CGTXR
5140      STA ICCOM
5150      JMP CIO
5160 *----------------------------*
5170 * Close IOCB #1, will be used *
5180 * for DIR, filenames IOCB     *
5190 *----------------------------*
5200 CLOSE1
5210      LDX #$10    ; Close IOCB #1
5220      LDA #CCLOSE
5230      STA ICCOM,X
5240      JMP CIO
5250 *----------------------------*
5260 * GRAPHICS g                  *
5270 * ENTRY: A-REG GRAPHICS MODE  *
5280 *  EXIT: Y-REG HAS STATUS     *
5290 *                             *
5300 *----------------------------*
5310 SNAME .BYTE "S:",0 ;OPEN FNAME
5320 GRAPHICS
5330      PHA         ;SAVE 'G'
5340      LDX #6*$10  ;FILE 6
5350      LDA #CCLOSE
5360      STA ICCOM,X
5370      JSR CIO     ;FIRST CLOSE #6
5380 * WE IGNORE ANY ERRORS
5390      LDX #6*$10  ;AGAIN, FILE 6
5400      LDA #COPN   ;OPEN THIS FILE
5410      STA ICCOM,X
5420      LDA # <SNAME
5430      STA ICBADR,X ;USE FILE "S:"
5440      LDA # >SNAME
5450      STA ICBADR+1,X ;POINT AT IT
5460 * ALL IS SET UP FOR OPEN, NOW
5470 * WE TELL CIO WHAT KIND OF OPEN
5480 *
5490      PLA         ;OUR SAVED MODE
5500      STA ICAUX2,X ;GIVEN TO 'S:'
5510 * (NOTE THAT S: IGNORES UPPER
5520 *  BITS OF AUX2)
5530      AND #$F0    ;GET UPPER BITS
5540      EOR #$10    ;AND FLIP BIT 4
5550 * (S: EXPECTS IT TO BE INVERTED
5560 *  FROM WHAT BASIC USAGE IS)
```

```
5570      ORA #$0C    ;ALLOW R/W
5580      STA ICAUX1,X ;FOR CIO AND S:
5590      JMP CIO     ;OPEN S:
5600 *----------------------------*
5610 * Fprint:                     *
5620 * X-Reg: Hi byte adr of string*
5630 * A-Reg: Lo byte adr of string*
5640 * String is terminated with   *
5650 * an escape character. Use to *
5660 * determine its length.       *
5670 * The text may have embedded  *
5680 * return characters, and may  *
5690 * be as long as 65536 bytes   *
5700 * if so desired - just so it  *
5710 * doesn't have an embedded    *
5720 * escape character.           *
5730 *----------------------------*
5740 SADR = $E0         ; Work str ptr
5750 *
5760 FPRINT
5770      STA ICBADR
5780      STX ICBADR+1
5790      STA SADR
5800      STX SADR+1
5810 * Find string length
5820      LDY #0
5830      STY ICBLEN+1
5840 LEN LDA (SADR),Y
5850      CMP #ESC
5860      BEQ GOTLEN
5870      INY
5880      BNE LEN
5890      INC ICBLEN+1
5900      INC SADR+1
5910      BNE LEN
5920 GOTLEN
5930      STY ICBLEN
5940      LDA #CPBINR
5950      STA ICCOM
5960      LDX #0
5970      JMP CIO
5980 *----------------------------*
5990 * LPRINT - Print a line of    *
6000 * text.  Use IOCB #7,         *
6010 * normally reserved for       *
6020 * printer I/O anyway          *
6030 *----------------------------*
6040 PRN .BYTE "P:",155
6050 PRCTL .BYTE 0
6060 LX   .BYTE 0        ; Hi tx adr
6070 LA   .BYTE 0        ; lo tx adr
6080 *----------------------------*
6090 LPRINT
6100      STX LX
6110      STA LA
6120      LDX #$50
6130      LDA #CCLOSE
6140      STA ICCOM,X ; Close it
6150      JSR CIO
6160      LDA #COPN   ; Open it
6170      STA ICCOM,X
6180      LDA #OPOUT  ; for output
6190      STA ICAUX1,X
6200      LDA # <PRN
6210      STA ICBADR,X
6220      LDA # >PRN
6230      STA ICBADR+1,X
6240      JSR CIO
6250      TYA              ; Print open
6260      BPL LPRINT1 ; error?
6270       FP  NOPRN
6280      LDA #0           ; Disable print
6290      STA PRCTL        ; on error
6300      RTS
6310 LPRINT1
6320      LDA LX           ; Open, whip
6330      STA ICBADR+1,X
6340      LDA LA           ; out!
6350      STA ICBADR,X
6360      LDA #80
6370      STA ICBLEN,X
6380      LDA #0
6390      STA ICBLEN+1,X
6400      LDA #CPTXR
```

```
6410        STA ICCOM,X
6420        JMP CIO
6430 *-----------------------------*
6440 * Close-all IOCBs we used,    *
6450 * clean house before exit,    *
6460 * since we are sloppy about   *
6470 * how we use them in the prog *
6480 *-----------------------------*
6490 CLOSEALL
6500        LDX #$10
6510        LDA #CCLOSE
6520        STA ICCOM,X
6530        JSR CIO
```

```
6540        LDX #$20
6550        LDA #CCLOSE
6560        STA ICCOM,X
6570        JSR CIO
6580        LDX #$50
6590        LDA #CCLOSE
6600        STA ICCOM,X
6610        JMP CIO
6620 *-----------------------------*
6630        *=  $02E0
6640        .WORD STARTUP
6650 *-----------------------------*
6660 *
```

(CONTINUED FROM PAGE 49)

C-MANSHIP LISTING 1: C

```
open_acct ( file )
char *file;
{
   int x, len;
   char zip[10], buf[25];

   if ( ( acctfile = fopen ( file, "br" )) == 0 )
      form_alert ( 1, "[1][Can't open the file][CONTINUE]" );
   else {
      fread ( chkname, 1, 26, acctfile );
      fread ( chkstreet, 1, 26, acctfile );
      fread ( chkcity, 1, 16, acctfile );
      fread ( buf, 1, 10, acctfile);
      strcpy ( &chkcity[strlen(chkcity)], ", " );
      fread ( &chkcity[strlen(chkcity)], 1, 3, acctfile );
      strcpy ( &chkcity[strlen(chkcity)], " " );
      fread ( zip, 1, 10, acctfile );
      len = strlen ( chkcity );
      if ( strlen ( zip ) > 5 ) {
         strncpy ( &chkcity[len], zip, 5 );
         chkcity[len+5] = 0;
         strcpy ( &chkcity[strlen(chkcity)], "-" );
         strcpy ( &chkcity[strlen(chkcity)], &zip[5] );
      }
      else
         strcpy ( &chkcity[strlen(chkcity)], zip );
      fread ( &balance, 4, 1, acctfile );
      if ( fclose ( acctfile ) != 0 )
            form_alert ( 1, "[1][File close error!][OKAY]");
      check_addr[CHKNAME].ob_spec = chkname;
      check_addr[CHKSTREE].ob_spec = chkstreet;
      check_addr[CHKCITY].ob_spec = chkcity;
      do_new_mnth ();
      if ( loaded )
         set_menu_entries ();
      else
         balance = 0;
   }
}


do_new_mnth ()
{
   int choice;

   cancdial_addr[CANCSTRG].ob_spec = newm;
   choice = get_month ();
   if ( choice == CANCOK ) {
      if ( !saved )
         save_month ( monthfile );
      open_new_month ();
   }
}


save_month ( file )
char *file;
{
   char newmfile[64];
   int x;

   strcpy ( newmfile, file );
   strcpy ( &newmfile[strlen(newmfile)-3], "BAK" );
   Fdelete ( newmfile );
   if ( Frename ( 0, file, newmfile ) == FAILED )
      form_alert ( 1, "[1][Error creating .BAK file!][OK]" );
   if ( ( mfile = fopen ( file, "bw" )) == 0 ) {
      form_alert ( 1, "[1][Disk Error!|Cannot save file.][CONTINUE]" );
      Frename ( 0, newmfile, file );
   }
   else {
      fwrite ( &num_trans, 2, 1, mfile );
      for ( x=0; x<num_trans; ++x )
         save_check ( x, mfile );
      if ( fclose ( mfile ) != 0 )
         form_alert ( 1, "[1][File close error!][OKAY]");
      else
         saved = TRUE;
      if ( ( mfile = fopen ( filename, "br+" )) == NULL )
         form_alert ( 1,
                  "[1][Error opening .MCK file!|Cannot update balance.][OK]" );
```

(PROGRAM LISTINGS CONTINUED ON PAGE 97)

LISTING 1: ASSEMBLY

(CONTINUED FROM PAGE 16)

```
0 ;LIST#D:MYUSR.ASM
10 ;ASM ,,#D:MYUSR.OBJ
11 SUM = $D4
12 NUM1 =$E0
13 NUM2 = $E2
20   *=1536        ; Assemble for PAGE 6.
30 ADDTHEM PLA     ; First off the stack is parameter count.
40   BEQ ERROR     ; Always check for no parameters ERROR.
50   CMP #2        ; Did we get exactly 2 parameters?
60   BEQ AOK
70   TAX           ; No, clean up stack and return safely.
80 CLEANUP PLA     ; Two bytes per parameter.
90   PLA
100  DEX           ; Get all the parameters off?
110  BNE CLEANUP   ; when all gone, just the valid return addr
120 ERROR RTS      ; is at the top of the stack for the RTS.
130 ; We have valid input, compute the sum.
140 ; The first parameter in the USR call (after the addr)
150 ; is the first parameter off the stack, high-byte
160 ; low-byte sequence. REMEMBER this!
170 AOK PLA        ; GetNUM1, high byte
180   STA NUM1+1
190   PLA          ; Get NUM1, low byte
200   STA NUM1
210   PLA          ; Get NUM2, high byte
220   STA NUM2+1
230   PLA          ; Get NUM2, low byte
240   STA NUM2
250 ; Now we have the data in temporary storage
260 ; and the stack is cleared of parameters.
270 ; Just the return address (to get us back to BASIC)
280 ; is at the top of the stack, which gets pulled off
290 ; into the program counter automatically by the RTS
300 ; instruction.
310   CLC          ; Must clear the carry flag first.
320   LDA NUM1     ; Low byte of first integer to add.
330   ADC NUM2     ; Add to low byte of second integer.
340   STA SUM      ; And store in low byte of their SUM.
350   LDA NUM1+1   ; Now add high bytes, leave carry alone.
360   ADC NUM2+1   ; It "carries over" from previous add.
370   STA SUM+1    ; And their summation is complete.
380   RTS          ; Back to BASIC/PX
```

CIRCLE #113 ON READER SERVICE CARD.

CIRCLE #114 ON READER SERVICE CARD.

(CONTINUED FORM PAGE 20)

```
QF 460 IF XX=3 THEN POSITION 10,4:? #6;"█
   "
EU 470 IF XX=4 THEN POSITION 10,4:? #6;"█
   ":SOUND 0,100,0,10
JJ 480 FOR R=15 TO 0 STEP -3:SOUND 0,R+70
   ,0,R:SOUND 1,R+90,0,R:NEXT R:TI=0:RETU
   RN
VK 490 GRAPHICS 18:POKE 708,14:POSITION 4
   ,5:? #6;"READING DATA":GOSUB 1270:GOTO
    1820
PT 500 GRAPHICS 17:POKE 559,0
YK 510 POKE 756,CH/256:POKE 708,137:POKE
   709,2:POKE 710,101:POKE 711,14
AQ 520 FOR C=0 TO 19:COLOR 2:PLOT C,23:DR
   AWTO C,RND(0)*5+12:NEXT C
UF 530 GOSUB 1070
FU 540 FOR I=0 TO 7:POSITION 0,I:? #6;"!!
   !!!!!!!!!!!!!!!!!!":NEXT I:POSITION 6,
   0:? #6;"skyrisers"
WS 550 GOSUB 950
FZ 560 FOR S=1 TO 29:POSITION RND(0)*19,R
   ND(0)*6+1:? #6;"V":NEXT S
MM 570 POSITION 0,22:? #6;"qqqqqqqqqqqqqq
   qqqqqq"
LT 580 FOR D=1 TO 18:POSITION RND(0)*19,R
   ND(0)*5+16:? #6;"w":NEXT D
OH 590 FOR D=1 TO 30:POSITION RND(0)*19,R
   ND(0)*5+16:? #6;"z":NEXT D
GX 600 FOR R=10 TO 22:POSITION 6,R:? #6;"
   U":POSITION 13,R:? #6;"U":NEXT R
FT 610 POSITION 9,3:? #6;"DGH":POSITION 9
   ,4:? #6;"L M":POSITION 9,5:? #6;"PTX"
VW 620 POSITION 6,22:? #6;"J":POSITION 13
   ,22:? #6;"F"
QT 630 POSITION 10,4:? #6;".":POSITION 2,
   2:? #6;"b"
QW 640 FOR G=17 TO 23:POSITION 2,G:? #6;"
   f":NEXT G:POSITION 2,16:? #6;"o"
TT 650 FOR G=19 TO 23:POSITION 3,G:? #6;"
   f":NEXT G
NI 660 FOR G=17 TO 23:POSITION 11,G:? #6;
   "f":NEXT G:POSITION 11,16:? #6;"n"
ZA 670 FOR G=18 TO 23:POSITION 8,G:? #6;"
   f":NEXT G
NW 680 FOR G=18 TO 23:POSITION 17,G:? #6;
   "f":NEXT G:POSITION 17,17:? #6;"o"
XH 690 FOR G=21 TO 23:POSITION 18,G:? #6;
   "f":NEXT G:POSITION 18,20:? #6;"n"
SV 700 POSITION 15,21:? #6;"a":POSITION 1
   5,22:? #6;"c":POKE 559,62
VH 710 GOSUB 1910
PL 720 GOTO 190
GI 730 DATA -1
UQ 740 REM ****PLANES AFTER PLAYER ONE***
   *
BV 750 POKE 704,0:POKE 707,255
GE 760 IF PEEK(53770)<30 THEN FOR D=1 TO
   4:POSITION 10,4:? #6;".":FOR G=1 TO 6:
   NEXT G:POSITION 10,4:? #6;"█":NEXT D
NK 765 IF PEEK(53770)<30 THEN RETURN
NZ 770 IF PEEK(53770)<100 THEN GOTO 820
DN 780 FOR R=0 TO 90 STEP 4:POKE 53248,R:
   SOUND 0,R,8,10:SOUND 1,100,0,4:NEXT R
SP 790 IF P1H<=9 THEN POKE 704,55:FOR D=1
    TO 5:NEXT D:POKE 53248,0:GOSUB 300:RE
   TURN
HV 800 FOR R=90 TO 224 STEP 6:POKE 53248,
   R:SOUND 0,R,0,10:NEXT R
ZG 810 RETURN
UZ 820 FOR R=0 TO 90 STEP 4:POKE 53251,R:
   SOUND 0,R,8,10:SOUND 1,100,0,4:NEXT R
JY 830 IF P1H<=15 THEN POKE 707,55:FOR D=
   1 TO 5:NEXT D:POKE 53251,0:GOSUB 300:R
   ETURN
VM 840 FOR R=90 TO 224 STEP 6:POKE 53251,
   R:SOUND 0,R,0,10:NEXT R:RETURN
AI 850 REM ****PLANES AFTER PLAYER TWO***
   *
VQ 860 IF PEEK(53770)<30 THEN FOR D=1 TO
   14:POSITION 10,4:? #6;".":FOR G=1 TO 6
   :NEXT G:POSITION 10,4:? #6;"█":NEXT D
NL 865 IF PEEK(53770)<30 THEN RETURN
NZ 870 IF PEEK(53770)<100 THEN GOTO 910
TC 880 FOR R=255 TO 157 STEP -4:POKE 5324
   9,R:SOUND 0,R,8,10:SOUND 1,100,0,4:NEX
   T R
UB 890 IF P2H<=15 THEN POKE 706,55:FOR D=
   1 TO 5:NEXT D:POKE 53249,0:GOSUB 370:R
   ETURN
VE 900 FOR R=157 TO 0 STEP -6:POKE 53249,
   R:SOUND 0,R,0,10:NEXT R:SOUND 0,0,0,0:
   RETURN
BX 910 FOR R=255 TO 160 STEP -4:POKE 5325
   0,R:SOUND 0,R,8,10:SOUND 1,100,0,4:NEX
   T R
DG 920 IF P2H<=9 THEN POKE 706,55:POKE 53
   250,0:GOSUB 370:RETURN
IS 930 FOR R=160 TO 0 STEP -6:POKE 53250,
   R:SOUND 0,R,0,10:NEXT R:RETURN
FM 940 REM ****P/M GRAPHICS****
WF 950 RESTORE 1020
CU 960 A=PEEK(106)-24:POKE 54279,A:POKE 5
   3277,3:POKE 623,1
EW 970 P0=256*A+1024:K=100:P1=256*A+1280:
   J=145:P2=256*A+1536:D=95:P3=256*A+1792
   :R=150
HW 980 FOR I=P0+K TO P0+7+K:READ F:POKE I
   ,F:NEXT I
HS 990 FOR I=P1+J TO P1+7+J:READ F:POKE I
   ,F:NEXT I
IA 1000 FOR I=P2+D TO P2+7+D:READ F:POKE
   I,F:NEXT I
EZ 1010 FOR I=P3+R TO P3+7+R:READ F:POKE
   I,F:NEXT I
OK 1020 DATA 0,128,192,254,127,56,16,0
GB 1030 DATA 0,0,1,3,126,252,60,14
MY 1040 DATA 0,1,126,255,126,1,24,0
MC 1050 DATA 0,128,255,126,56,96,0,0
AW 1060 REM ****DLI****
IC 1070 RESTORE 1100
AX 1080 FOR A=1536 TO 1536+28:READ B:POKE
    A,B:NEXT A:POKE 512,0:POKE 513,6:DL=P
   EEK(560)+256*PEEK(561):POKE 54286,192
YU 1090 POKE DL+28,128:RETURN
AV 1100 DATA 72,138,72,141,10,212,169
FZ 1110 DATA 132
QN 1120 DATA 141,26,208,162,15,141,10,212
   ,202,208,250,173,200,2,141,26,208,104,
   170,104,64
PJ 1130 REM ****GAME OVER****
XL 1140 GRAPHICS 17:POKE 756,224:POKE 711
   ,14:POKE 710,255:POKE 708,135:POKE 712
   ,0:POKE 559,0
LO 1150 FOR G=0 TO 22:POSITION 0,G:? #6;"
   ____________________":NEXT G
DE 1160 POSITION 6,5:? #6;"game over"
CW 1170 IF P1W=1 THEN POSITION 3,11:? #6;
   "PLAYER ONE WINS"
VW 1180 IF P2W=1 THEN POSITION 3,11:? #6;
   "PLAYER TWO WINS"
LG 1190 IF TIE=1 THEN POSITION 3,11:? #6;
   " game is a tie  ":FOR D=155 TO 0 STEP
    -6.4:SOUND 0,D*20,10,5:NEXT D
JF 1200 SOUND 0,0,0,0:SOUND 1,0,0,0:POKE
   559,62
CJ 1210 FOR E=1 TO 20:FOR D=15 TO 0 STEP
   -2:SETCOLOR 0,24,D:SOUND 0,D*60,10,1:N
   EXT D:NEXT E:POKE 712,0
FY 1220 FOR G=1 TO 5:FOR S=1 TO RND(0)*15
   :SOUND 0,S*9,6,10:SOUND 1,5,10,3:NEXT
   S:SOUND 0,0,0,0:SOUND 1,0,0,0
FL 1230 POKE 712,G*5:POKE 712,0:NEXT G
QC 1240 GOTO 2050
QS 1250 GOTO 1250
AL 1260 REM ****CHARACTER SET****
DE 1270 CH=(PEEK(106)-8)*256:FOR I=0 TO 7
   :POKE CH+I,5:NEXT I
CB 1280 FOR I=0 TO 512:POKE CH+I,PEEK(537
   44+I):NEXT I
LM 1290 RESTORE 1330
SU 1300 READ A:IF A<0 THEN RETURN
WX 1310 FOR J=0 TO 7:READ B:POKE CH+A*8+J
   ,B:NEXT J:GOTO 1300
EI 1320 REM ****CHARACTER DATA****
SJ 1330 DATA 33,126,66,66,66,126,194,194,
   194
RH 1340 DATA 2,255,255,255,255,255,255,25
   5,255
OP 1350 DATA 5,0,16,56,124,16,16,16,0
DI 1360 DATA 36,127,64,64,64,67,68,68,72
```

SKYRISER

PROGRAM LISTING

```
VY 1370 DATA 46,239,239,239,199,199,131,1
   71,131
CD 1380 DATA 35,254,134,134,128,128,128,1
   30,254
AD 1390 DATA 47,255,255,251,243,227,203,1
   39,131
YV 1400 DATA 48,72,68,68,67,64,64,64,127
IQ 1410 DATA 41,8,8,8,24,24,24,24,24
YM 1420 DATA 40,255,1,1,129,97,17,17,9
ZB 1430 DATA 52,0,0,0,0,193,62,0,255
DW 1440 DATA 39,254,130,128,128,128,142,1
   34,254
MA 1450 DATA 45,9,9,5,5,5,5,9,9
PQ 1460 DATA 51,254,192,192,192,254,2,2,2
   54
VE 1470 DATA 37,254,128,128,128,254,192,1
   92,254
CS 1480 DATA 38,131,171,131,171,131,171,1
   31,131
HD 1490 DATA 50,252,132,132,132,254,198,1
   98,198
RS 1500 DATA 43,66,68,72,88,224,254,194,1
   94
IV 1510 DATA 58,255,213,255,213,255,213,2
   55,255
QE 1520 DATA 49,255,255,0,255,0,255,0,255
AY 1530 DATA 55,255,147,255,255,147,255,2
   55,147
AN 1540 DATA 57,130,130,130,130,254,6,6,1
   26
VX 1550 DATA 54,0,0,0,4,0,0,0,0
XR 1560 DATA 42,255,255,255,255,255,231,2
   31,231
QV 1570 DATA 6,0,16,16,16,124,56,16,0
GF 1580 DATA 11,0,8,16,56,124,124,56,0
YY 1590 DATA 14,0,56,68,4,8,16,0,16
LZ 1600 DATA 44,72,72,80,80,80,80,72,72
MO 1610 DATA 56,129,129,129,129,147,147,2
   11,127
ZQ 1620 DATA 53,255,255,255,255,255,255,2
   55,255
YZ 1630 DATA 17,60,126,255,255,255,255,12
   6,60
OO 1640 DATA 18,0,8,33,4,0,8,5,88
NU 1650 DATA 19,128,32,66,0,68,16,128,76
NZ 1660 DATA 20,2,32,13,8,66,4,32,0
FZ 1670 DATA 21,102,13,168,0,8,96,2,128
FB 1680 DATA 22,0,0,1,0,0,9,5,0
XS 1690 DATA 23,0,0,0,0,8,64,0,64
EM 1700 DATA 25,0,0,0,5,0,16,0,1
VS 1710 DATA 26,16,128,32,0,0,0,0,0
QQ 1720 DATA 1,0,0,0,0,0,0,0,0
FI 1730 DATA 34,12,24,48,112,112,48,24,12
RY 1740 DATA 16,255,0,255,0,255,0,255,0
AU 1750 DATA 24,129,129,129,129,129,129,1
   29,129
ZM 1760 DATA 33,255,255,255,255,199,131,1
   31,199
GC 1770 DATA 35,215,215,171,147,171,85,10
   9,109
QD 1780 DATA 39,255,0,62,193,0,0,0,0
XB 1790 DATA 56,9,17,17,97,129,1,1,255
EU 1800 DATA -1
IQ 1810 REM ****TITLE SCREEN****
AN 1820 GRAPHICS 18:POKE 559,0:POKE 756,C
   H/256:POKE 711,14:POKE 710,255:POKE 70
   9,2
XS 1830 FOR R=54248 TO 53252:POKE R,0:NEX
   T R
UU 1840 FOR G=4 TO 6:POSITION 0,G:? #6;"!
   !!!!!!!!!!!!!!!!!!!!":NEXT G
IW 1850 POSITION 6,5:? #6;"skyrisers":POK
   E 559,62
KH 1860 FOR T=1 TO 13
EY 1870 FOR D=15 TO 0 STEP -1:SETCOLOR 0,
   24,D:SOUND 0,D+60,10,1:NEXT D:SOUND 0,
   0,0,0
KU 1880 NEXT T
TP 1890 GOTO 2190
XG 1900 REM ****MOVE PLANES DEMO****
XJ 1910 M=255:POKE 704,255:POKE 705,49:PO
   KE 706,0:POKE 707,212
UY 1920 FOR R=0 TO 255 STEP 5:POKE 53248,
   R:POKE 53249,M-R:SOUND 0,R,0,3:SOUND 1
   ,R,0,2:NEXT R
```

```
LU 1930 FOR R=0 TO 255 STEP 6:POKE 53250,
   M-R:POKE 53251,R:NEXT R:GOTO 110
VP 1940 REM ****FIREWORKS!****
GF 1950 QQ=9:KK=3:POKE 708,0
DZ 1960 COLOR 3:FOR C=3 TO 5:POSITION 9,C
   :? #6;"  ":NEXT C
TO 1970 FOR Q=1 TO 7:FOR D=15 TO 0 STEP -
   6:SOUND 0,D,8,D:POKE 712,D+45:NEXT D:N
   EXT Q:POKE 712,0
SX 1980 FOR FIRE=1 TO 16:POSITION QQ,KK:?
    #6;"23":POSITION QQ,KK+1:? #6;"45"
AR 1990 FOR S=1 TO RND(0)*15:SOUND 0,S*7,
   6,10:SOUND 1,S,10,3:NEXT S:SOUND 0,0,0
   ,0:SOUND 1,0,0,0
HA 2000 POSITION QQ,KK:? #6;"67":POSITION
    QQ,KK+1:? #6;"59":FOR D=1 TO 3:NEXT D
TV 2010 POSITION QQ,KK:? #6;"  ":POSITION
    QQ,KK+1:? #6;"  ":POKE 708,0
CV 2020 QQ=INT(RND(0)*2)+9:KK=INT(RND(0)*
   3)+1
LO 2030 NEXT FIRE
PA 2040 GOTO 1130
XE 2050 FOR D=1 TO 200:NEXT D:GRAPHICS 17
   :POKE 708,0:POKE 710,14:CC=0:POKE 712,
   3
KA 2060 POSITION 0,19:? #6;"_■_■_■_■_■_■_
   ■_■_■_■"
MV 2070 POKE 708,0:POKE 709,146
QY 2080 POSITION 0,2:? #6;"_■_■_■_■_■_■_■
   _■_■_■"
XV 2090 FOR R=54248 TO 53252:POKE R,0:NEX
   T R
KV 2100 POSITION 5,10:? #6;"PLAY AGAIN?":
   FOR R=15 TO 0 STEP -1:SOUND 0,20,10,R:
   NEXT R
UP 2110 POSITION 5,13:? #6;"press fire":F
   OR D=1 TO 20:NEXT D
WP 2120 POSITION 5,13:? #6;"           ":
   FOR D=1 TO 15:NEXT D
MU 2130 IF STRIG(0)=0 OR STRIG(1)=0 THEN
   GOTO 2190
WB 2140 CC=CC+1:IF CC>45 THEN CC=0:GOTO 2
   160
OQ 2150 GOTO 2110
DC 2160 GRAPHICS 18:POKE 712,64:POKE 708,
   14
KT 2170 POSITION 1,5:? #6;"OH,COME ON ALR
   EADY!"
UP 2180 FOR D=15 TO 0 STEP -3:SOUND 0,D*6
   ,10,D:NEXT D:FOR G=1 TO 300:NEXT G:GOT
   O 1810
WA 2190 GRAPHICS 17:POKE 709,130:POKE 708
   ,132:POKE 710,14:T=11:POKE 756,224:POK
   E 712,130:POKE 711,255
LS 2200 POSITION 0,6:? #6;"_______________
   _________"
DX 2210 POSITION 0,17:? #6;"______________
   __________"
PI 2220 POSITION 1,5:? #6;"select skill l
   evel":POKE 708,223
WZ 2230 POSITION 6,11:? #6;"EASY"
ZS 2240 POSITION 6,13:? #6;"INTERMEDIATE"
EM 2250 POSITION 6,15:? #6;"HARD"
QR 2260 FOR D=15 TO 0 STEP -1:SOUND 0,50,
   10,D:NEXT D
ZS 2270 FOR D=15 TO 0 STEP -2:SOUND 0,20,
   10,D:NEXT D:SOUND 0,0,0,0
AG 2280 POSITION 5,T:? #6;">":FOR D=1 TO
   10:NEXT D:POSITION 5,T:? #6;" ":FOR D=
   1 TO 8:NEXT D
GJ 2290 IF STICK(0)=14 OR STICK(1)=14 THE
   N T=T+2
GT 2300 IF STICK(0)=13 OR STICK(1)=13 THE
   N T=T-2
EL 2310 IF T>15 THEN T=11
FI 2320 IF T<11 THEN T=15
BL 2330 IF STRIG(0)=0 OR STRIG(1)=0 THEN
   2350
SQ 2340 GOTO 2280
XR 2350 IF T=11 THEN TD=8
GY 2360 IF T=13 THEN TD=4.5
VN 2370 IF T=15 THEN TD=3
OU 2380 FOR D=10 TO 0 STEP -1:SOUND 0,35,
   10,D:NEXT D:FOR R=0 TO 19:PLOT R,0:DRA
   WTO R,23:NEXT R
PS 2390 GOTO 500
```

(CONTINUED FROM PAGE 28)

LISTING 1:GFA BASIC 3.0

```
' PROGRAM: Dialog Box Tutorial
DIM medrbtn&(3),denrbtn&(2)
GOSUB my_menu
GOSUB initialize
GOSUB window_open
GOSUB ld_resource
ON MENU GOSUB respond_menu
DO
  ON MENU
LOOP
PROCEDURE respond_menu
  ' Procedure to respond to menu items
  hold$=strip$(MENU(0))
  IF hold$="Dialog Tutorial"
    ' code to handle Desk menu item here
    txt$="Dialog Box Tutorial|By David Plotkin"
    ALERT 1,txt$,1,"OK",a%
  ENDIF
  IF hold$="Add info"
    ' code for Add info
    GOSUB show_resource
  ENDIF
  IF hold$="Quit"
    ' code for Quit
    GOSUB pgm_end
  ENDIF
  MENU OFF
RETURN
PROCEDURE my_menu
  ' Menu definition procedure
  MENU KILL
  RESTORE m_data
  DIM strip$(150)
  FOR i%=0 TO 150
    READ strip$(i%)
    EXIT IF strip$(i%)="***"
  NEXT i%
  strip$(i%)=""
  strip$(i%+1)=""
  m_data:
  DATA Desk,"Dialog Tutorial"
  DATA -------------------
  DATA 1,2,3,4,5,6,""
  DATA File,Add info,--------------,Quit
  DATA ***
  MENU strip$()
RETURN
PROCEDURE initialize
  ' initializes variables
  ' These variable statements were written out by the Resource
  ' Construction Set by toggling the box to write GFA Output under OUTPUT
  ' in the GLOBAL menu item.  For this to work, you must have provided
  ' an name for each item in the dialog box while using RSC.
  LET benefits&=0 !RSC_TREE
  LET title&=1 !Dialog box title string
  LET name&=2 !Editable text field for name
  LET address&=3 !Editable text field for street address
  LET city&=4 !Editable text field for city
  LET state&=5 !Editable text field for state
  LET zip&=6 !Editable text field for zipcode
  LET box1&=7 !Box around the medical plans(groups radio buttons)
  LET box2&=12 !Box around the dental plans(groups radio buttons)
  LET mplana&=9 !Medical plan A radio button
  LET mplanb&=10 !Medical plan B radio button
  LET mplanc&=11 !Medical plan C radio button
  LET dplana&=14 !Dental plan A radio button
  LET dplanb&=15 !Dental plan B radio button
  LET depno&=17 !Title string for number of dependents
  LET depup&=18 !Button to increase number of dependents (touchexit)
  LET depdown&=16 !Button to decrease number of dependents (touchexit)
  LET depnbrsl&=26 !Button holding the number of dependents
  LET impdown&=20 !Button to decrease importance slider
  LET impup&=23 !Button to increase importance slider
  LET imparent&=21 !Box containing the importance slider
  LET impslide&=22 !The importance slider
  LET ok&=24 !OK exit button
  LET cancel&=25 !Cancel exit button
  LET mplnttle&=8 !Title string for medical plan box
  LET dplnttle&=13 !Title string for dental plan box
  ' now load the medical and dental radio button values into their arrays
  medrbtn&(1)=mplana&
  medrbtn&(2)=mplanb&
  medrbtn&(3)=mplanc&
  denrbtn&(1)=dplana&
  denrbtn&(2)=dplanb&
```

```
RETURN
PROCEDURE ld_resource
  ' load the resource file
  RESERVE FRE(0)-30000
  IF RSRC_LOAD("BENEFITS.RSC")=0
    ALERT 3,"Resource file not found",1," STOP ",a%
    RESERVE FRE(0)+30000
    EDIT
  ENDIF !RSRC_LOAD("BENEFITS.RSC")=0
  ~RSRC_GADDR(0,0,adr%) !get the address of the dialog box
  ' initialize the operating width of the importance slider
  width%=OB_W(adr%,imparent&)-OB_W(adr%,impslide&)
RETURN
PROCEDURE show_resource
  ' put the dialog box on the screen
  ~FORM_CENTER(adr%,x%,y%,w%,h%) !get the centered coordinates
  GET x%,y%,x%+w%,y%+h%,hold$ !store the screen rectangle
  ' clear the dialog box strings
  CHAR{{OB_SPEC(adr%,name&)}}=""
  CHAR{{OB_SPEC(adr%,city&)}}=""
  CHAR{{OB_SPEC(adr%,address&)}}=""
  CHAR{{OB_SPEC(adr%,state&)}}=""
  CHAR{{OB_SPEC(adr%,zip&)}}=""
  ' set the number of dependents to zero
  depnum%=0
  CHAR{OB_SPEC(adr%,depnbrsl&)}=STR$(depnum%)
  ' set the importance factor to 50
  import%=50
  CHAR{OB_SPEC(adr%,impslide&)}=STR$(import%)
  ' and the position of the slider itself to 500 (halfway)
  sliderpos%=500
  newpos%=width%/1000*sliderpos%
  OB_X(adr%,impslide&)=newpos%
  dum%=FORM_DIAL(1,0,0,0,0,x%,y%,x%+w%,y%+h%)
  ~OBJC_DRAW(adr%,0,6,x%,y%,w%,h%)   ! draw the box
  REPEAT !loop til OK or CANCEL is pressed
    exit_obj%=FORM_DO(adr%,0)
    ' handle different exit objects here
    IF exit_obj%=depup& !number of dependents goes up
      INC depnum%  ! increase by one
      IF depnum%>99 !make sure only two digits (otherwise,...crash!)
        depnum%=99
      ENDIF !depnum%>99
      CHAR{OB_SPEC(adr%,depnbrsl&)}=STR$(depnum%)
      ~OBJC_DRAW(adr%,depnbrsl&,6,x%,y%,w%,h%)
    ENDIF  ! exit_obj%=depup&
    IF exit_obj%=depdown& !number of dependents goes down
      DEC depnum% !decrease by one
      IF depnum%<0  !hmmm, not possible
        depnum%=0  ! fix it
      ENDIF !depnum%<0
      CHAR{OB_SPEC(adr%,depnbrsl&)}=STR$(depnum%)
      ~OBJC_DRAW(adr%,depnbrsl&,6,x%,y%,w%,h%)
    ENDIF !exit_obj%=depdown&
    IF exit_obj%=impslide& !clicked on the slider itself
      ' exit dialog and allow user to drag slider to new position
      sliderpos%=GRAF_SLIDEBOX(adr%,imparent&,impslide&,0)
      newpos%=width%/1000*sliderpos%
      OB_X(adr%,impslide&)=newpos%
      ' Now set value in slider
      import%=sliderpos%/10
      CHAR{OB_SPEC(adr%,impslide&)}=STR$(import%)
      ' redraw the slider
      ~OBJC_DRAW(adr%,imparent&,3,x%,y%,w%,h%)
    ENDIF ! exit_obj%=impslide&
    IF exit_obj%=impdown& ! clicked on the down button
      SUB sliderpos%,10  ! subtract five
      IF sliderpos%<0
        sliderpos%=0
      ENDIF !sliderpos%<0
      newpos%=width%/1000*sliderpos%
      OB_X(adr%,impslide&)=newpos%
      ' Now set value in slider
      import%=sliderpos%/10
      CHAR{OB_SPEC(adr%,impslide&)}=STR$(import%)
      ' redraw the slider
      ~OBJC_DRAW(adr%,imparent&,3,x%,y%,w%,h%)
    ENDIF ! exit_obj%=impdown&
    IF exit_obj%=impup& ! clicked on the up button
      ADD sliderpos%,10  ! add five
      IF sliderpos%>1000
        sliderpos%=1000
      ENDIF !sliderpos%>1000
      newpos%=width%/1000*sliderpos%
      OB_X(adr%,impslide&)=newpos%
      ' Now set value in slider
      import%=sliderpos%/10
      CHAR{OB_SPEC(adr%,impslide&)}=STR$(import%)
      ' redraw the slider
      ~OBJC_DRAW(adr%,imparent&,3,x%,y%,w%,h%)
    ENDIF ! exit_obj%=impdown&
    IF exit_obj%=ok&
      hndl_ok
    ENDIF  ! exit_obj%=ok&
    IF exit_obj%=cancel&
      dum%=FORM_DIAL(2,0,0,0,0,x%,y%,x%+w%,y%+h%)
      PUT x%,y%,hold$  ! restore the screen
      ' change the exit object back to non-selected
      ~OBJC_CHANGE(adr%,exit_obj%,0,x%,y%,w%,h%,0,0)
    ENDIF !exit_obj%=cancel&
  UNTIL (exit_obj%=ok& OR exit_obj%=cancel&)
RETURN
PROCEDURE pgm_end
  ' clean up memory, get rid of the resource and end
  ~RSRC_FREE() !toss the resource
  RESERVE FRE(0)+30000 !get the memory back
  CLOSEW #1
  EDIT !return to the editor
RETURN
PROCEDURE window_open
  OPENW 1
  FULLW 1
  CLEARW 1
  TITLEW #1,"STLOG Dialog Box Demo in GFA Basic"
RETURN
PROCEDURE hndl_ok
  dum%=FORM_DIAL(2,0,0,0,0,x%,y%,x%+w%,y%+h%)
  PUT x%,y%,hold$  ! restore the screen
  ' change the exit object back to non-selected
  ~OBJC_CHANGE(adr%,exit_obj%,0,x%,y%,w%,h%,0,0)
  ' recover the editable text fields
  nm$=CHAR{{OB_SPEC(adr%,name&)}}
  city$=CHAR{{OB_SPEC(adr%,city&)}}
  address$=CHAR{{OB_SPEC(adr%,address&)}}
  state$=CHAR{{OB_SPEC(adr%,state&)}}
  zip$=CHAR{{OB_SPEC(adr%,zip&)}}
  CLEARW 1
  PRINT AT(1,1);"Results of Dialog Box: "
  PRINT
  PRINT "Name: ";nm$
  PRINT "Address: ";address$
  PRINT "City: ";city$;" State: ";state$;" Zip: ";zip$
  ' and the radio buttons
  ' First the medical plan (3 buttons):
  select&=0 ! none selected
  FOR cnt&=1 TO 3
    IF BTST(OB_STATE(adr%,medrbtn&(cnt&)),0) ! test bit 0 (selected)
      select&=medrbtn&(cnt&)
    ENDIF ! BTST(OB_STATE(adr%,medrbtn&(cnt&)),0)
  NEXT cnt&
  IF select&<>0 ! a radio button was selected
    ' retrieve the text of the selected button
    med$=CHAR{OB_SPEC(adr%,select&)}
    PRINT "Medical Plan Selected: ";med$;" which is button ";select&-8
    ' change the radio buttons back to non-selected
    ~OBJC_CHANGE(adr%,select&,0,x%,y%,w%,h%,32,0)
  ELSE
    PRINT "No medical plan was selected!"
  ENDIF !select&<>0
  ' now the dental plan buttons
  select&=0 ! none selected
  FOR cnt&=1 TO 2 !only two dental plan buttons
    IF BTST(OB_STATE(adr%,denrbtn&(cnt&)),0) ! test bit 0 (selected)
      select&=denrbtn&(cnt&)
    ENDIF ! BTST(OB_STATE(adr%,denrbtn&(cnt&)),0)
  NEXT cnt&
  IF select&<>0 ! a radio button was selected
    ' retrieve the text of the selected button
    den$=CHAR{OB_SPEC(adr%,select&)}
    PRINT "Dental Plan Selected: ";den$;" which is button ";select&-13
    ' change the radio buttons back to non-selected
    ~OBJC_CHANGE(adr%,select&,0,x%,y%,w%,h%,32,0)
  ELSE
    PRINT "No dental plan was selected!"
  ENDIF !select&<>0
  ' get the number of dependents (read the button contents)
  nmdep$=CHAR{OB_SPEC(adr%,depnbrsl&)}
  PRINT "Number of dependents: ";nmdep$
  import$=CHAR{OB_SPEC(adr%,impslide&)}
  PRINT "Importance of plans: ";import$
RETURN
```

(CONTINUED FORM PAGE 35)

LISTING 1: BASIC

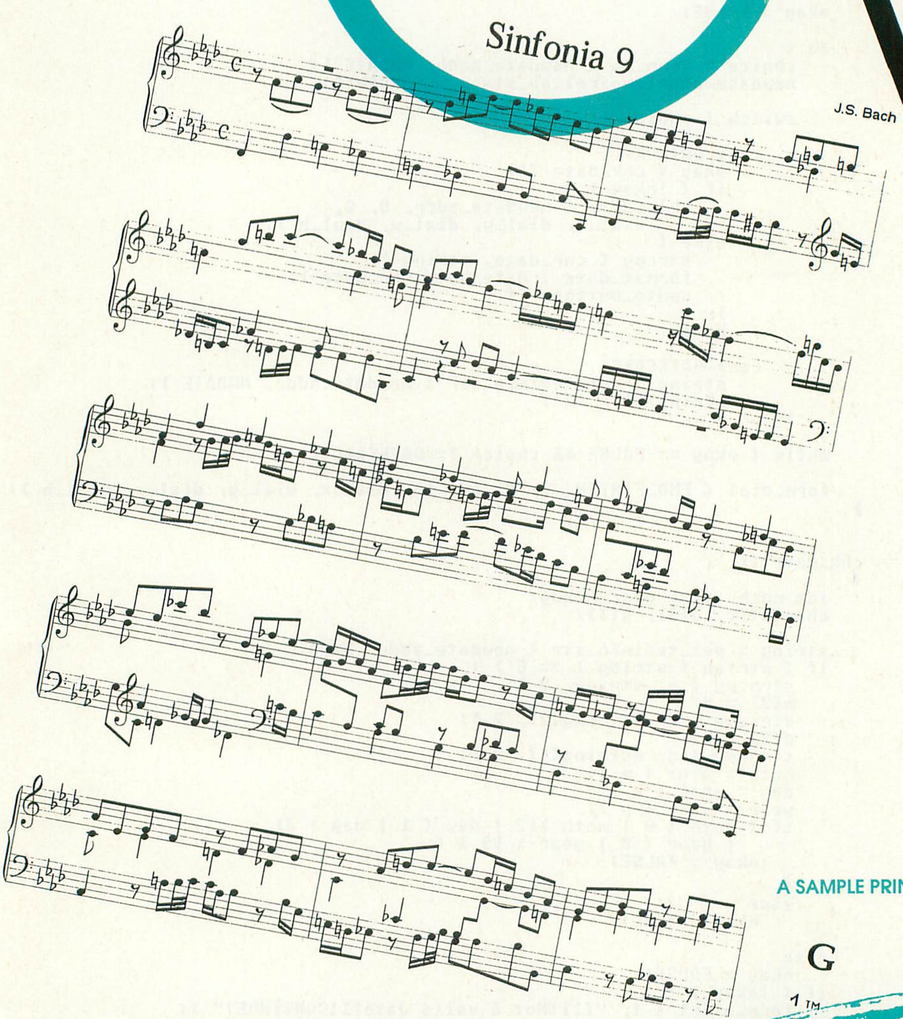
```
EI 1 REM ******************************
JG 2 REM *     ATARI TRUCHET TILES      *
XH 3 REM *        by Frank Kweder       *
TB 4 REM *                              *
TC 5 REM *                              *
VS 6 REM *        COPYRIGHT 1989        *
LW 7 REM *     BY ANALOG COMPUTING      *
EP 8 REM ******************************
NO 9 REM
LV 10 GRAPHICS 24:COLOR 1:POKE 764,255
GH 15 POKE 708,138:POKE 709,14:POKE 710,0
   :POKE 711,60:POKE 712,64:POKE 559,0
EN 20 POKE 752,1:G=1:VSCROLL=32:J=1:SPACE
   =33
BL 45 REM
RW 49 REM  DRAW PATTERN FROM DATA 
JH 50 GOSUB 300:GOSUB 700
FI 55 FOR Z=2 TO 11 STEP 2:PLOT 0,Z:DRAWT
   O 319,Z:NEXT Z
SY 60 FOR R=12 TO 154 STEP 12:FOR C=9 TO
   300 STEP 12
DA 65 IF J=5 THEN J=1:RESTORE 1000
SX 70 READ V1:GOSUB 420:NEXT C:J=J+1
MI 75 NEXT R
DF 80 FOR Z=155 TO 167 STEP 2:PLOT 0,Z:DR
   AWTO 319,Z:NEXT Z
SC 198 REM
RD 199 REM  VERTICAL SCROLL ROUTINE 
AP 200 POKE 87,0:FOR A=0 TO 7
LF 210 POKE 54277,A:Z=SIN(A):NEXT A
SK 220 Z=COS(A):POSITION 24,0:? "press_s
   pace";
QB 230 FOR A=7 TO 0 STEP -1
LL 240 POKE 54277,A:Z=SIN(A):NEXT A
LO 250 FOR Z=1 TO 200:NEXT Z
HP 260 ON PEEK(764)=SPACE GOTO 500+300*(G
   =2):GOTO 200
SD 298 REM
WR 299 REM SET UP DLI & VERTICAL SCROLL
FN 300 DL=PEEK(560)+256*PEEK(561)
FL 310 POKE DL+3,64+7:POKE DL+6,6+VSCROLL
QT 311 REM
UC 312 REM SHIFT 4K BOUNDARY DOWN 1 LINE
CQ 315 POKE DL+102,PEEK(DL+101):POKE DL+1
   01,PEEK(DL+100)
BF 316 POKE DL+100,PEEK(DL+99):POKE DL+99
   ,PEEK(DL+98)
RO 318 REM
RL 319 REM  CLEAN UP BOTTOM OF SCREEN 
XS 320 POKE DL+174,65:POKE DL+175,PEEK(56
   0):POKE DL+176,PEEK(561)
ZF 330 RETURN
SE 398 REM
RK 399 REM  PLOT RANDOM DESIGN 
OZ 400 V1=INT(PEEK(53770)/128)
GO 420 ON V1+1 GOTO 430,450
RR 428 REM
JL 429 REM  PLOT AT 430 IF V1=0 
XB 430 PLOT C+5,R:DRAWTO C+5,R+2:DRAWTO C
   +2,R+5:DRAWTO C,R+5
CP 432 PLOT C+6,R:DRAWTO C+6,R+2:DRAWTO C
   +2,R+6:DRAWTO C,R+6
ES 434 PLOT C+11,R+5:DRAWTO C+9,R+5:DRAWT
   O C+5,R+9:DRAWTO C+5,R+11
KS 436 PLOT C+11,R+6:DRAWTO C+9,R+6:DRAWT
   O C+6,R+9:DRAWTO C+6,R+11
DO 438 IF G<3 THEN PLOT C+11,R:DRAWTO C+1
   1,R+11:PLOT C+11,R+11:PLOT C+9,R+11
IQ 439 IF G<3 THEN PLOT C+7,R+11:PLOT C+5
   ,R+11:PLOT C+3,R+11:PLOT C+1,R+11
ZI 440 RETURN
RV 448 REM
MD 449 REM  PLOT AT 450 IF V1=1 
YC 450 PLOT C+5,R:DRAWTO C+5,R+2:DRAWTO C
   +9,R+6:DRAWTO C+11,R+6
VT 452 PLOT C+6,R:DRAWTO C+6,R+2:DRAWTO C
   +9,R+5:DRAWTO C+11,R+5
VJ 454 PLOT C,R+5:DRAWTO C+2,R+5:DRAWTO C
   +6,R+9:DRAWTO C+6,R+11
TP 456 PLOT C,R+6:DRAWTO C+2,R+6:DRAWTO C
   +5,R+9:DRAWTO C+5,R+11
DS 458 IF G<3 THEN PLOT C+11,R:DRAWTO C+1
   1,R+11:PLOT C+11,R+11:PLOT C+9,R+11
IU 459 IF G<3 THEN PLOT C+7,R+11:PLOT C+5
   ,R+11:PLOT C+3,R+11:PLOT C+1,R+11
ZM 460 RETURN
SF 498 REM
VK 499 REM  DRAW BY PLOTTING RANDOM # 
QR 500 PUT #6,125:COLOR 1:G=2:GOSUB 700
WC 510 FOR R=1 TO 154 STEP 12:FOR C=9 TO
   300 STEP 12
AK 520 GOSUB 400:NEXT C:NEXT R
AV 530 POKE 764,255:GOTO 80
SH 698 REM
RX 699 REM  MESSAGES AND BORDERS 
UK 700 POSITION 0,0:POKE 87,0:? "___TRUCH
   ET__TILES___";
ZI 710 IF G=1 THEN ? "________pattern_______
   _";
IZ 720 IF G=2 THEN ? "________random_______
   _";
OK 730 POKE 87,8:POKE 764,255:POKE 559,34
HU 740 FOR Z=1 TO 8 STEP 2:PLOT Z-1+Z/3,1
   :DRAWTO Z-1+Z/3,176:PLOT 319-Z-Z/3,1
IC 750 DRAWTO 319-Z-Z/3,176:NEXT Z:RETURN
SI 798 REM
JF 799 REM PLOT FULL SCREEN TO PRINT 
HG 800 GRAPHICS 8+16:G=3
VM 810 COLOR 1:POKE 710,0:POKE 712,128
QV 820 PLOT 4,0:DRAWTO 315,0:DRAWTO 315,1
   91:DRAWTO 4,191:DRAWTO 4,0
XV 830 FOR R=0 TO 191 STEP 12:FOR C=4 TO
   305 STEP 12
AR 840 GOSUB 400:NEXT C:NEXT R
RZ 848 REM
NG 849 REM  MENU 
CM 850 GRAPHICS 8+32:POKE 710,0:POKE 752,
   1:POKE 764,255
FM 860 ? "K":? " 1 PLOT AGAIN    2 PRIN
   T    3 END";
CS 870 CLOSE #1:OPEN #1,4,0,"K:":GET #1,A
   :A=A-48
VA 880 ON A GOTO 800,890,990:GOTO 870
TR 890 GRAPHICS 8+16+32:POKE 709,0
SJ 898 REM
DP 899 REM  PRINT WITH G: DEVICE 
UO 900 TRAP 989
KH 910 CLOSE #7:OPEN #7,8,0,"G3:"
BX 920 XIO 64,#7,3,0,"G3:":GOTO 850
XM 989 ? "↓↓↓    ???? G.OBJ NOT INSTALLED
   ????":TRAP 40000
LG 990 GRAPHICS 0:END
SN 999 REM
QY 1000 REM  DATA FOR PATTERN 
BC 1001 DATA 1,0,1,0,1,0,1,0,1,0,1,0,1,0,
   1,0,1,0,1,0,1,0,1,0,1
ZX 1002 DATA 0,0,1,1,1,0,0,0,1,1,1,0,0,0,
   1,1,1,0,0,0,1,1,1,0,0
BK 1003 DATA 1,1,0,0,0,1,1,1,0,0,0,1,1,1,
   0,0,0,1,1,1,0,0,0,1,1
AF 1004 DATA 0,1,0,1,0,1,0,1,0,1,0,1,0,1,
   0,1,0,1,0,1,0,1,0,1,0
```

(CONTINUED FROM PAGE 89)

```
else {
   fseek ( mfile, 91L, FROM_BEG );
   fwrite ( &balance, 4, 1, mfile );
   if ( fclose ( mfile ) != 0 )
            form_alert ( 1, "[1][File close error!][OK]");
      }
   }
}


open_new_month ()
{
   int mnth, x;

   for ( x=JAN; x<=MZERO; ++x )
      if ( cancdial_addr[x].ob_state == SELECTED )
         if ( x == MZERO )
            mnth = 0;
         else
            mnth = x-JAN+1;
   sprintf ( cancmnth, "%d", mnth );
   open_month ( acct_name, cancmnth );
}


open_month ( file, mnth )
char *file, *mnth;
{
   int x, len, button, do_it, trans_cnt, old_dep_cnt, old_chk_cnt;
   char a[20], new_mfile[64];

   saved = TRUE;
   old_dep_cnt = num_deps;
   old_chk_cnt = num_chks;
   num_chks = num_deps = 0;
   strcpy ( new_mfile, filename );
   strcpy ( &new_mfile[strlen(new_mfile)-4], mnth );
   strcpy ( &new_mfile[strlen(new_mfile)], ".DAT" );
   if ( ( mfile = fopen ( new_mfile, "br" )) == 0 )
      form_alert ( 1, "[1][Can't open the file][CONTINUE]" );
   else {
      do_it = TRUE;
      fread ( &trans_cnt, 2, 1, mfile );
      if ( trans_cnt == 0 ) {
         button = form_alert ( 1, "[2][The data file for|this month is \
empty.|Do you want to start|a new month?][YES|NO]" );
         if ( button == YES ) {
            num_trans = load_auto ();
         }
         else {
            do_it = FALSE;
            num_chks = old_chk_cnt;
            num_deps = old_dep_cnt;
         }
      }
      else
         num_trans = trans_cnt;
      if ( do_it ) {
         clear_window ();
         loaded = TRUE;
         if ( balance < 0 && balance > (-100) )
            sprintf ( bal_but, "$-%ld.%02ld", balance/100,labs(balance%100) );
         else
            sprintf ( bal_but, "$%ld.%02ld", balance/100,labs(balance%100) );
         strcpy ( monthfile, new_mfile );
         strcpy ( acct_name, file );
         month = atoi ( mnth );
         x = 0;
         while ( x < trans_cnt ) {
            read_check ( x, mfile );
            if ( strcmp ( checks[x].number, "9999" ) == MATCH )
               num_deps += 1;
            else
               num_chks += 1;
            ++x;
         }
         if ( x > 0 ) {
            strcpy ( cur_chk_num, checks[x-1].number );
            curchknum = atoi ( cur_chk_num );
            if ( strcmp ( cur_chk_num, "9999" ) != MATCH ) {
               curchknum += 1;
               sprintf ( a, "%d", curchknum );
               len = strlen ( a );
               strcpy ( &cur_chk_num[4-len], a );
            }
         }
         cur_top = edit_top = 0;
         cur_count = num_trans;
         cur_chk_strc = checks;
         strcpy ( windname, acct_name );
         strcpy ( &windname[strlen(windname)], ": Edit mode" );
         wind_set ( w_h2, WF_NAME, windname, 0, 0 );
         full_draw = TRUE;
         if ( fclose ( mfile ) != 0 )
            form_alert ( 1, "[1][File close error!][OKAY]");
      }
   }
}


load_auto ()
{
   char autoname[64];
   FILE *autofile;
   int x, count;

   count = 0;
   strcpy ( autoname, filename );
   strcpy ( &autoname[strlen(autoname)-4], ".AUT" );
   if ( ( autofile = fopen ( autoname, "br" )) != NULL ) {
      fread ( &count, 2, 1, autofile );
      x = 0;
      while ( x < count ) {
         read_check ( x, autofile );
         if ( strcmp ( checks[x].number, "9999" ) == MATCH ) {
            num_deps += 1;
            balance += checks[x].amount;
         }
         else {
            num_chks += 1;
            balance -= checks[x].amount;
         }
         ++x;
      }
      saved = FALSE;
      if ( fclose ( autofile ) == FAILED )
         form_alert ( 1, "[1][Error closing AUTO file!][CONTINUE]" );
   }
   return ( count );
}

save_check ( i, f )
int i;
FILE *f;
{
   fwrite ( checks[i].number, 1, 4, f );
   fwrite ( checks[i].payee, 1, 30, f );
   fwrite ( checks[i].memo, 1, 30, f );
   fwrite ( checks[i].date, 1, 8, f );
   fwrite ( &checks[i].amount, 4, 1, f );
   fwrite ( checks[i].cancel, 1, 1, f );
   fwrite ( "THIS SPACE FOR POSSIBLE FUTURE EXPANSION", 1, 40, f );
}

read_check ( i, f )
int i;
FILE f;
{
   fread ( checks[i].number, 1, 4, f );
   fread ( checks[i].payee, 1, 30, f );
   fread ( checks[i].memo, 1, 30, f );
   fread ( checks[i].date, 1, 8, f );
   fread ( &checks[i].amount, 4, 1, f );
   fread ( checks[i].cancel, 1, 1, f );
   fread ( future_use, 1, 40, f );
}

clear_window ()
{
   GRECT r;

   wind_get ( w_h2, WF_WORKXYWH, &r.g_x, &r.g_y, &r.g_w, &r.g_h );
   draw_rec ( r, 2, 8, WHITE );
}

get_month ()
{
   int choice;
   int dial_x, dial_y, dial_w, dial_h;
   clear_cancdial ();
   form_center ( cancdial_addr, &dial_x, &dial_y, &dial_w, &dial_h );
   form_dial ( FMD_START, 0, 0, 10, 10, dial_x, dial_y, dial_w, dial_h );
   objc_draw ( cancdial_addr, 0, 8, dial_x, dial_y, dial_w, dial_h );

   choice = form_do ( cancdial_addr, 0 );
   cancdial_addr[choice].ob_state = SHADOWED;

   form_dial ( FMD_FINISH, 0, 0, 10, 10, dial_x, dial_y, dial_w, dial_h );
   return ( choice );
}

clear_cancdial ()
{
   int x;

   for ( x=JAN; x<=MZERO; cancdial_addr[x++].ob_state = NORMAL ) ;
   if ( month != -1 )
      if ( month == 0 )
         cancdial_addr[MZERO].ob_state = SELECTED;
      else
         cancdial_addr[month+JAN-1].ob_state = SELECTED;
}

get_new_date ()
{
   int choice, okay;
   int dial_x, dial_y, dial_w, dial_h;
   string = get_tedinfo_str ( newdate_addr, NWDATE );
   string[0] = 0;
   form_center ( newdate_addr, &dial_x, &dial_y, &dial_w, &dial_h );
   form_dial ( FMD_START, 0, 0, 10, 10, dial_x, dial_y, dial_w, dial_h );
   objc_draw ( newdate_addr, 0, 8, dial_x, dial_y, dial_w, dial_h );

   okay = FALSE;

   do {
      choice = form_do ( newdate_addr, NWDATE );
      newdate_addr[choice].ob_state = SHADOWED;

      switch ( choice ) {

         case DATEOK:
            okay = chk_date ();
            if ( !okay )
               objc_draw ( newdate_addr, 0, 8,
                   dial_x, dial_y, dial_w, dial_h );
            else {
               strcpy ( cur_date, string );
               format_date ( date_but, cur_date );
               updte_buttons ();
            }
            break;

         case DATECANC:
            string = get_tedinfo_str ( newdate_addr, NWDATE );
            string[0] = '@';
      }
   }
   while ( okay == FALSE && choice != DATECANC );

   form_dial ( FMD_FINISH, 0, 0, 10, 10, dial_x, dial_y, dial_w, dial_h );
}


chk_date ()
{
   int mnth, day, year, okay;
   char m[3], d[3], y[3];

   string = get_tedinfo_str ( newdate_addr, NWDATE );
   if ( strlen ( string ) == 6 ) {
      strncpy ( m, string, 2 );
      m[2] = 0;
      strncpy ( d, &string[2], 2 );
      d[2] = 0;
      strncpy ( y, &string[4], 2 );
      mnth = atoi ( m );
      day = atoi ( d );
      year = atoi ( y );
      if ( mnth < 0 | mnth >12 | day < 1 | day > 31
           | year < 0 | year > 99 ) {
         okay = FALSE;
      }
      else
         okay = TRUE;
   }
   else
      okay = FALSE;
   if ( !okay ) {
      form_alert ( 1, "[1][Not a valid date!][CONTINUE]" );
      string[0] = 0;
   }
   return ( okay );
}

updte_buttons ()
{
   if ( !full ) {
      set_buttons ();
      center_butstring ( bal_but, 35, 194 );
      center_butstring ( trans_but, 131, 194 );
      center_butstring ( check_but, 227, 194 );
      center_butstring ( dep_but, 323, 194 );
      center_butstring ( mnth_but, 419, 194 );
      center_butstring ( date_but, 515, 194 );
   }
}
```

FROM OVE

By Marshal M. Rosenthal



A SAMPLE PRINT OUT FROM *MUSIC PUBLISHER*

This month, we'll continue to play "hooky," hanging out in England and enjoying the "fine" weather that flip-flops between warm, sunny days and hailstorms followed by unrelenting cold rain. We haven't quite picked the bones clean from the ST show yet, so let's give it one more go.

One of the perks of being in the thick of things is that you get the chance to see companies that don't plan to bring their products to the States—at least not yet. As an example, there's Take Control's *Music Publisher.* It has the distinction of being the first page-layout-oriented music desktop-publishing program designed specifically for Atari users, allowing the creation and printout of full orchestral scores.

Professional musicians, composers and copyists will find this program a godsend. Mouse and keyboard combinations let you insert notes and symbols anywhere, do deletions, cut-and-paste operations, even insert or remove spaces for adjusting appearance. Multiple-window editing enables copying between files, orchestration and part extraction. The physical size of the composition can be varied through a page-layout window, allowing A3, A4, U.S. legal and a variety of other sizes.

Note support includes cue-size notes and three different shapes for note heads. Plus you can create one, three, four or five line staves. There's a full repertoire of symbols, common and user-defined time signatures, even two sizes of clefs. Text can be placed anywhere on the page, with three fonts supplied (use any font or size with GDOS support), plus underline, bold, italic, light and outline text effects.

Printer drivers are supplied for Epson FX/compatible (9 pin), Epson LQ/compatible (24 pin) and the Atari Laser printer (a Postscript print driver is currently under development). Not for the hobbyist, *Music Publisher* costs 249 pounds (about $450) but really delivers. On the down side, there isn't any MIDI support. Take Control does promise this in a future upgrade, though, and they also note a number of other enhancements in the works as well, such as grouping notes together for mass movements and keyboard shortcuts.

Music Publisher requires a minimum of one meg, although two would seem appropriate if you load any accessories with it. (I recommend using Codehead Software's excellent *G+Plus* GDOS replacement with *Music Publisher.*) A monochrome monitor will give the cleanest image, although color can be used effectively with just a bit of the Toolbox images disappearing at the bottom left; plan on viewing the work at the lowest magnification. One double-sided drive will handle everything if necessary, but two drives are better and a hard drive is best of all.

Sticking with music for a bit, let's check

THE BIG WATER

out *The Music Matrix*. This disk-based magazine from Scotland does more than just provide interesting music to listen to; it lets you participate in the playback of other musicians' work. You can learn arrangement and composition by studying scores and progressions, and investigate bass lines and drum tracks from the data supplied on the disk. Make your own changes and then listen to what you've created (keeping in mind that the original music should be respected; I'd make a backup first).

Programs include patch editors and sequencer demos, plus public-domain materials (although the disk mostly contains original materials produced at some cost of time to the professional musicians who support it). Lou Nisbet, editor, invites your involvement and support. Feel free to drop him a line.

From Capo Software in Holland comes *That's Write*. New word processors are always regarded skeptically at first: "What features could they possible have that I'm not getting now?" In this case, many. Start with optimized pagination, auto formatting, automatic indexing and table of contents. Then add text and graphics on the same line (text being high-quality GEM-IMG format), with wide support for both built-in fonts (proportional, pica, elite, 15 dpi, enlarged, double size...) and graphic GEM fonts (Helbo, Symbol, Gothic, Times...). This is all in one "What You See Is What You Get" package that works on color as well as monochrome monitors (but color doesn't look half as good as mono).

Add paragraph and page layouts, mouse and keyboard control, even macro functions. Want more? Try a built-in spelling checker (dictionary goes into RAM if there's enough memory), file functions, font editor and resident snapshot accessory. The keyboard can be edited for international characters, and there's also a mail-merge with interface to some databases.

A wide range of 9- and 24-pin printers is supported, as well as lasers (including Atari's) and daisy wheels. Just plan on having plenty of memory for this baby—at least one megabyte.

We haven't scoured all the games from the pot this month. *Populous* fans will be happy to learn that Electronic Arts U.K. will shortly be coming out with additional "world" data disks. One will be *The Bit Plains*, featuring (did you guess?) a landscape comprising a programmer's world. The ground will consist of printout paper, with various settlements composed of such computers as ZX81s and Crays, and the landscape will be dotted with pencil stubs and cigarette butts. (They smoke much more over there.)

Many have heard of JRI's JATO accelerator board, developed here in the States. Hypercache comes from Germany and also consists of a 16MHz 68000 chip contained on a circuit card. Like JATO, the original 8MHz CPU must be unsoldered and replaced by this board. Wires must also be attached to pins of both the Shifter and sound chip. Again like JATO, only internal calculations can reach the higher processing speed.

The most striking difference, besides the price (about $250 as opposed to JATO's $99), is the built-in RAM cache. This enables information being written to RAM to go into this CPU cache and be "massaged" at the higher speed. Speed increase is rated at about 20-30% over normal in most situations as a result. (Software control over the cache can be activated through an included desk accessory.) Unlike JATO, there is no provision to tone the board down to 8MHz.

Well, every good thing must come to an end. In this case it means my returning to New York and facing tons of junk mail and bills. And maybe a few ST goodies in the post...?

***Marshal M. Rosenthal** is a writer and New York-based photographer whose work has appeared in major publications throughout the world. He specializes in children and in the electronic entertainment field.*

Products mentioned:

BIT PLAINS
Electronic Arts U.K.
Langley Business Centre/11-49 Station Road
Langley, Nr. Slough
Berkshire, England SL3 8YN

HYPERCACHE
Third Coast Technologies
Unit 8, Bradley Hall Trading Estate
Standish, Wigan
England WN6 OXQ

MUSIC MATRIX
Jongleur House
14 Main Street, East Wemyss, Fife
Scotland KY1 4RU

MUSIC PUBLISHER
Take Control Software
Jonic House
Speedwell Road
Hay Mills, Birmingham
England B25 8EU

THAT'S WRITE
Capo Software
Distributed by Cavendish Distributors, Ltd.
85 Tottenham Court Road
London, England W1

ATARI

AUTHORIZED SERVICE
CENTER FOR ALL
ATARI PRODUCTS

MICROTYME

A DIVISION OF MICRO PERIPHERALS, INC.
4049 MARSHALL RD. • KETTERING, OHIO 45429

VISA DISCOVER

ATARI

MICROTYME

The nations best source for
Atari Hardware & Software!
Nearly **2000** ST titles in stock!

MICROTYME

New Games!

3D Pool	29
Astaroth	25
Bermuda Project	25
Bio Challenge	32
Blasteroids (**Wow**)	25
Blood Money	25
Buffalo Bill's Rodeo	32
California Games	31
Chariots of Wrath	31
Cosmic Pirate	25
Devon Aire	32
Dominator	25
Dragon Ninja	25
Eliminator	22
Emmanuelle	25
F-16 Combat Pilot	37
F40 Pursuit Simulator	29
Garfield: Winter's Tail	25
Gauntlet 2	32
Giants (4 Games)	38
Gilbert	25
Guide to the Solar System	29
Hawkeye	25
Hellraiser	25
High Steel	26
Hole-In-One Mini Golf	18
Indy Jones-Last Crusade	
☛ Action Game	26
☛ Graphics Advnture	32
Joe Blade	19
Journey Centre Earth	25
Kult	31
Legend	25
Legend of DJEL	32
Licence to Kill (007)	25
Millenium	32
Murder in Venice	31
Nether World	22
Newzealand Story	26
Operation Hormuz	25
Operation Neptune	31
Pac-Land	25
Pacmania (3D Pac-Man)	26
Personal Nightmare	38
Phobia	25
Populous	32
☛ Data disk	19
Powerdrome	25
Project Neptune	26
Pro-Soccer	26
Psycho	19
Purple Saturn Day	31
RVF Honda	25
Rambo 3	19
Red Heat	26
Red Lightning	39
Rick Dangerous	32
RoboCop	25
Rocket Ranger	33
Roll-Out	23
Roy of the Rovers	25
Savage	22
S.T.A.G.	32
Skweek	26
Sleeping God's Lie	32
Stormtrooper	25
Story So Far (4 Games)	25
Thunderbirds	31
Timerunner	25
Time Scanner	25
Tom & Jerry	31
Triad (3 Games)	26
TV Sports: Football	32
Waterloo	31
Weird Dreams	31
World of Flight	99

Accounting

B.E.S.T. Business Mgmt.	194
B.E.S.T. General Ledger	52
Chartpak	31
Inventory Manager	52
LDW Power	89
Personal Finance Mgr.	32
Phasar 3.0	58
ST Accounts (GEM)	95
Swift Calc ST	48

Books

ABACUS Books	
#0 Presenting the ST	15
#2 Internals	15
#3 GEM Prog. Ref.	15
#4 Machine Language	15
#5 Tricks & Tips	15
#12 3D Graphics	19
#13 Disk Drives	18
ATARI ST Book	14
ATARI ST Companion	14
Dungeon Master Guide	9
Elementary ATARI ST	16
GFA Basic Programmers Reference Vol. 1	24
GFA Quick Reference	12
Helpful Hints for the ATARI ST or Mega User	14
MIDI for Musicians	14
Programming in C	17
Programming w/ith GFA-Basic 3.0	22
Quest for Clues 1 or 2	18
ST Artist	16
ST BASIC Source Book	14
ST Reference #2: AES	16

Database

DB Man 5	149
Data Manager ST	44
Informer	64
Informer Trial Pack	13
Regent Base 2	75
Suberbase Personal	44
Suberbase Pro. 3.0x	225

Educational

First Shapes	25
Fraction Action	24
Kidtalk	24
Kinderama	24
Magical Math 1	19
Mathblaster!	26
Mathblaster Plus!	32
Mathtalk: Fractions	24
Mavis Beacon Typing	32
Planetarium	33
PreSchool Kid Progs	26
Read & Ryme	24
Smooth Talker	29
Spelling Bee	19
Statistics 1	19

Games

10th Frame Bowling	26
Airball	24
All Aboard!	16
Artura	27
Autoduel	24
Baal	19
Balance of Power 1990	32
Balistix	23
Barbarian	19
Bards Tale	31
Battle Chess	32
Battle Tech	31
Blockbuster	25
Boulderdash Constr. Kit	17
Breach	25
Bridge 5.0	22
Bubble Ghost	21
Captain Blood	31
Captain Fizz	21
Chessmaster 2000	28
Chrono Quest	32
Circus Games	26
Colossus Chess X	23
Combat Course	25
Dark Side	26
Death Bringer	25
Death Sword	19
Deep Space	19
Defender of the Crown	31
Demon's Winter	25
Dive Bomber	18
Dungeon Master	24
Dungeon Master 2	CALL
Elite	23
Empire	32
Falcon	28
☛ Mission Disk	16
Fire and Forget	25
Flight Simulator 2	32
☛ Scenery Disks	ea 21
F.O.F.T.	31
Foundations Waste	24
Gateway	28
Gladiator	25
Gold Rush	26
Goldrunner 2	25
☛ Scenery Disks	ea 8
Gunship	31
Harrier Combat Simulator	32
Heroes of the Lance	26
High Roller	18
Hit's Disk Vol. 1 (4 Games)	33
Hostage Rescue Mission	28
Hunt for Red October	32
Hyperdome	21
I Ludicrus	22
Impossible Mission 2	27
Indy Jones-Temple Doom	32
International Soccer	25
Jet	34
Jinxter	25
Jupiter Probe	17
Karateka	21
Kings Quest 1-4	ea 31
Knickerbockers	12
Lancelot	26
Leaderboard Dual Pack	16
Leatherneck	25
Leisure Suit Larry 1	25
Leisure Suit Larry 2	32
Lock On	25
Lords of Conquest	13
Major Motion	25
ManHunter (N.Y.)	32
ManHunter 2 (S.F.)	32
Menace	19
Microleague Baseball 2	38
Midi Maze	25
Missle Command	19
Moon Patrol	19
Night on the Town	22
Ninja Mission	14
Obliterator	25
Off-Shore Warrior	26
Ogre	27
Oids	21
Orbiter	26
Platoon	25
Police Quest 1 or 2	ea 32
Pool of Radiance	CALL
Prime Time	27
Prison	25
Quink	11
R-Type	24
Rastan	19
Road Runner	29
Roadwars	19
Robotron: 2084	19
Scruples	25
Sex Vixens from Space	25
Shadowgate	31
Shared of Spring	27
Shuffleboard	19
Sinbad	24
Space Conquest:	18
Space Harrier	31
Space Quest 1 or 2	ea 31
Space Quest 3	37
STAC	44
STOS	39
STOS Compiler	25
Star Fleet 1	35
Starglider 2	27
Superman	26
Superstar Ice Hockey	31
Talespin	31
Tanglewood	25
Test Drive	24
Test Drive 2	CALL
☛ Cars Disk	CALL
☛ Scenery Disk	CALL
Tetris	25
The Games: Winter	28
Thunder Blade	32
Times of Lore	26
Total Eclipse	26
Typhoon Thompson	22
Uninvited	31
Universal Military Sim.	31
Vampires Empire	18
Vegas Craps	22
Vegas Gambler	22
Wargame Const. Set	22
Winter Challenge	10
Wizards Crown	25
World Games	12
World Karate Champ.	16
Zak McKracken	28
Zany Golf	26

Graphics

Architectural Design Disk	19
Art Gallery 1-3	ea 19
Becker-Cad	259
Cad-3D 2.0	57
Cartoon Design Disk	19
Cyber Control	39
Cyber Paint 2.0	46
Cyber Print	32
Cyber Sculpt	53
Cyber Texture	32
Cyber VCR	45
DEGAS Elite	38
Drafix CAD	114
Draw-Art Professional	46
Dyna CADD	439
Easy Draw (Supercharger)	95
Easy-Tools	32
First CADD	33
Fonts and Borders	21
Future Design Disk	19
Genesis	49
Human Design Disk	19
IB-Scan Pix #1-Religion	24
IB-Scan Pix #2-Pets	24
IB-Scan Pix #3-Holidays	24
Interior Design Disk	19
Maps & Legends	22
☛ Datamaps Vol. 1	19
Master Cad	119
Microbot Design Disk	19
Neochrome	26
Print Master Plus	24
Scan-Art	32
Spectrum 512	41
Touch-Up	109
Video Titling Disk	19

Midi & Music

All three below	109
☛ Chord Master	44
☛ Note Wizard	44
☛ Scale Master	44
Copyist 1	66
Copyist 2 (Professional)	158
Dr. Drums	19
Dr. Keys	19
EZ Score Plus	99
EZ Track Plus	43
Edit Trak	119
Guitaristics	49
KCS 1.7 (Regular)	149
KCS Level 2	219
Ludwig	99
Master Traks Jr.	86
Master Traks Pro.	269
Midimax	32
Midi Recording Studio	36
Midisoft Studio (Std)	66
Midisoft Studio (Adv)	99
Music Studio '88	39
T.I.G.E.R.	99
Take Note	52
Tune Smith	99
X-Or	199

Misc.

ANSI Term	18
Flash	18
Fontz!	22
Generation Gap (512k)	32
Generation Gap Plus	32
Great Chefs Set (1-3)	39
HyperFont	32
Interlink	24
Michtron BBS 3.0	52
PC-Ditto 3.0	59
Shadow	18

Programming

AssemPro	39
Cambridge Lisp	59
Devpac ST	66
Forth/MT	32
GFA-Basic 3.0	58
HiSoft Basic Professional	99
Laser C	119
Laser DB (Debugger)	53
Lattice C 3.04	109
Mark Williams C	114
☛ C Source Debugger	39
Personal Pascal	66
Prospero C	129
Prospero Fortran	129
Prospero Pascal	98
R.A.I.D. (Debugging)	34

Publishing

Calamus	174
Calamus Font Editor	64
Fleet Street 2.0	129
PageStream	115
☛ Font Disks	ea 25
Publisher ST	59
☛ Design Ideas	15
☛ Education Graphics	20
☛ People, Places,Things	20
☛ Symbols & Slogans	20
UltraScript ST	159
☛ Font Pack (PS35)	209

Utilities

CodeHead Utilities	19
G + Plus	21
Go Go	22
Hard Disk Accallerator	20
Hard Disk Backup	23
Hard Drive Turbo Kit	39
Juggler (New version)	32
Macro Mouse	25
Mirror Image	19
Multi Desk	21
NeoDesk 2	29
OmniRes	23
ProCopy	24
Revolver	31
Turbo Jet	26
Turbo ST	23
Universal Item Selector 2	13
Utilities Plus	39
Virus Killer	26

Word Processing

First Word Plus	64
Microsoft Write	65
Tempus 2	49
Word Perfect	139
Word-Up	47
Word Writer ST	49

Close-outs & Specials!

(Quantities are limited on close-outs!)

220 ST	7
3D Tic-Tac-Toe	8
Algebra 1-3	ea 17
ALT	15
Architecture Library I	16
Aussie Joker Poker	26
Bach Songbook	20
Ballyhoo	22
Biology 1-4	ea 15
Bizmarck	23
Black Cauldren	22
Blackjack	8
Border Pack (Easy Draw)	19
Brimstone	25
CZ Android	43
CZ Patch	54
Calendar	17
Chemistry 1 or 2	ea 15
Clip Art 6	11
Colorwriter	3
Cornerman	27
Cosmic Relief	21
Crazy Cars	21
Cross Town Crazy 8	6
Db Calc 2	3
DX Patch	22
Deadline	27
Delta Patrol	14
Diablo	10
Diamond Mike	4
Digi-Drum	4
☛ Samples Disk	4
Do-It! (Dos Shell)	14
Downhill Challenge	12
Dungeon Master Maps	4
EZ-Data	27
Easy Record	25
Farenheit 451	27
Fast	27
Fast Basic	49
Fast Basic M Compiler	5
Financial Cookbook	10
Fish	11

HOURS: M-F 9 a.m.-9 p.m. EST
SAT 9 a.m.-5 p.m.

ALL 50 STATES CALL TOLL FREE
1-800-255-5835

For Order Status or
Tech. Info, Call (513) 294-6236

TERMS AND CONDITIONS

NO EXTRA CHARGES FOR CREDIT CARDS! - Minimum order $15 - C.O.D: Yes, if all Shipping Charges are PRE-PAID - Shipping: Hardware, minimum $4; Software and most accessories, minimum $3 - Next day shipment available at extra charge - We ship to Alaska, Hawaii, Puerto Rico (UPS Blue Label Only), APO and FPO - Canadian orders, actual shipping plus 5%, minimum $5 - Ohio residents add 6% sales tax - Please allow 2 weeks for personal or company checks to clear - All defective products require a Return Authorization Number to be accepted for repair or replacement - No free trials or credit - Returns subject to 20% re-stocking charge - Due to changing market conditions, call toll free for latest price and availability of product. FOR YOUR PROTECTION, WE CHECK ALL CREDIT CARD ORDERS FOR FRAUD.

MICROTYME

Close-outs & Specials!

(CON'T)

Font Factory 1-6	ea 17
Font Pack (Easy Draw)	27
Font Writer	22
Frostbyte	12
GFA-Artist	44
GFA-Companion	27
GFA-Draft	54
GFA-Object	44
GFA-RayTrace	50
GFA-Vector	33
Gone Fishin'	24
Gridiron	9
H & D Base	56
Hacker 2	9
Hippo Backgammon	22
Hippo Concept	49
Hippo Fonts 1	30
Hippo Word	49
Home Casino Poker	18
Jewels of Darkness	7
K-Roget	38
K-Switch	17
Kermit & Remote Acc.	13
Knight Orc	25
Leather Goddesses	22
Liberator	10
M-Cache	20
Mi-Term	27
Macro Desk	3
Maniax	6
Mercenary	20
☛ 2nd City Data Disk	9
Metropolis	9
Midway Battles	6
Mindwheel	25
Moon Mist	9
Music Construction Set	11
Music Mouse	40
N-Vision	21
Navigator	28
Nervana Quest 1&2	15
Nervana Quest 3&4	15
Oo-Topos	11
Paladin	21
☛ Quest 1	14
Pawn	25
Pinball Factory	20
Pinball Wizard	19
Power Assist	17
Power Edit	17
Power Plan	44
ProShow +	15
Q-Ball	16
Quadralien	16
Quizam!	19
Regent Word 2	38
Renegade (Outcast)	9
Roadwar 2000	23
Roadwar Eurpoa	25
Santa Paravia	17
Seconds Out	12
Shanghai	20
Shuttle 2	22
Space Station/Protector	3
Spell Breaker	27
Spitfire-40	18
ST Karate	3
ST Key	3
ST Music Box	27
ST One Write	49
Star Quake	16
Star Ray	16
Sub-Battle Simulator	11
Super Huey	22
Time Link	27
Thunder	23
Time Blast	10
Trail Blazer	25
Transputer	18
Transylvania	11
Typesetter	22
Wanderer (3D Game!)	13
Word Master	27
World of Wine Set	15
☛ Champagne	9
☛ Great Vintage Years	9
Virus	13
Your Financial Future	22
Zero Gravity	16
Zork 2 or 3	ea 25
Zynapse	18

Printers

ATARI

SLM 804 Laser	1399
SLM 804 PostScript Laser	1450

OLYMPIA

NP-30	209
NP-136	525

PANASONIC

KX-P1180 *(192 cps)*	189
KX-P1191 *(240 cps)*	239
KX-P1124 *(192 cps)*	339
KX-P4450 *(8 ppm)*	1499

STAR

NX-1000 Multifont	179
NX-1000 Rainbow	239

Printer Accessories

ATARI

SLM 804-Toner	57
SLM 804-Drum	164

OLYMPIA

NP-30/80 Black	5
NP-30/80 Colors	6
NP-136 Black	9

PANASONIC

1080/91-1180/91 Black	10
1080/91-1180/91 Colors	10
1124 Black	12
4450-Toner	42
4450-Drum	149

STAR

NX-1000 Black	6
NX-1000 Rainbow	8

Modems

ATARI

SX212	75

AVATEX

1200 or 2400	CALL

SUPRA

2400 (1 yr. Warranty!)	124

ZOOM

2400 (2 yr. Warranty!)	115

Hard Drives

ATARI

MEGAFILE 30	CALL

ICD

Fa-ST 30	699
Fa-ST 50	799
Fa-ST Tape Backup	799
Tape Cartridges	35

SUPRA

20 Meg.	439
30 Meg.	599
45 Meg.	659
60 Meg.	759
FD-10	879

Joysticks

ATARI Pro-Line	9
Competition Pro 5000	18
EPYX 500XJ	17
Powerplayers	19
Star-Cursor	39
WICO Bat Handle	18
WICO Ergostick	18

Disks

(No Generic JUNK!)

3.5"

BASF SS/DD	13
BASF DS/DD	16
BASF HD	38
CENTECH DS/DD (Colors)	18
MAXELL DS/DD (Colors)	16
SONY DS/DD	16
SONY DS/DD Bulk 10 Pack (Colors)	12

5.25"

BASF DS/DD	9
FUJI HD	15
GOLDSTAR DS/DD	4
MAGMEDIA DS/DD (Colors)	15
MAXELL DS/DD	9
POLAROID DS/DD	9
SONY HD	16
VERBATIM HD	16

Floppy Drives

ATARI SF314	189
INDUS GTS-100	179
INDUS GTS-100x	169
Master Drive	145
I.B. Drive *(5.25)*	209

Scanners

CANON Flat-Bed	1650
CANON Sheet-Fed	1059
IMG Scan	64
MIGRAPH Hand Scanner w/ Touch-Up	439

Misc. Hardware

Color Computer Eyes	178
Drive Master	32
Dust Covers	
ATARI 520ST	9
ATARI 1040ST	10
ATARI SC1224	14
ATARI SM124	13
PANASONIC 1091	10
STAR NX-10	12
STAR NX-15	12
STAR NX-1000	9
DVT-VCR Backup	209
MIDI CABLES (**Heavy Duty**)	
5'	ea 5
10'	ea 9
15'	ea 13
20'	ea 16
25'	ea 19
Monitor Master	33
Mouse Master	29
Mouse Mat	7
PC-Ditto II	CALL
Printer Stand	9
Replay-4	86
Spectre GCR	CALL
Spectre 128	124
128k Roms	129
ST Mouse	54
Tweety Board	41
VideoKey	69

Paper

MAILING LABELS

500-White	4
800-Rainbow	9
Paper Pack-Bright Colors	8

HOURS: M-F 9 a.m.-9 p.m. EST
SAT 9 a.m.-5 p.m.

ALL 50 STATES CALL TOLL FREE
1-800-255-5835

For Order Status or
Tech. Info, Call (513) 294-6236

CIRCLE #115 ON READER SERVICE CARD.

American Techna-Vision

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

1050 MECHANISM

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

$47.50

VISICALC SPREADSHEET

Unleash the computing power of your 8 bit Atari with Visicalc. Compute everything from home finances to high powered financial projections. Hundreds of uses.

$19.95 DISK

XL 40 PIN LSI CHIP SET

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

$19.95

800 4 PIECE BOARD SET

Includes Main Board, Power Supply Assembly, CPU Module and 10K Revision B Operating System Module. All boards are new, tested and complete with all components. $28.50

800/400 MODULES

NEW PARTS COMPLETE WITH IC'S

• CX853 16K RAM Module	$9.50
• 800/400 CPU with GTIA	$8.75
• 800 10K "B" OS MODULE	$8.75
• 800 POWER BOARD	$14.50
• 400 POWER BOARD	$9.50

INTEGRATED CIRCUITS

• CO14795	$4.50
• CO14805	$4.50
• CO12296	$4.50
• CO10745	$4.50
• CO14377	$4.50
• CO60472	$4.50
• 6532	$4.50
• CO12399B	$4.50
• CO12499B	$4.50
• CO14599B	$4.50
• CO14806	$4.50
• CO10750	$4.50
• CO10444	$8.50
• CO60302	$9.50
• CO21697	$15.00
• CO61991	$15.00
• CO61598	$20.00
• CO61618	$24.00
• CO24947	$15.00

1050 IC'S

• U7 - 6532	$4.50
• U8 - 6810	$4.50
• U9 - 6507	$4.50
• U10 ROM	$19.50
• U13 2793	$19.50
• U5 LM2917	$8.50
• 5713	$5.25

All other 1050 IC's
$3.75 each
Specify by Socket#

850 INTERFACE

12 Piece Chip set. Includes all plug in IC's except ROM. Replacement fixes vast majority of 850 failures $19.50

POWER PACKS

Exact replacement transformer for 800/400, 1050 810, 1200XL, 850, and XF551 units. Part #CO17945.

$14.50

XL/XE SUPPLY

Power Pak for 800XL, 600XL 130XE, 65XE & XE Game.

$25.00

THE BOOKKEEPER AND CX85 KEYPAD

You get both Atari's 8 bit professional bookkeeping system and the handy CX85 numeric keypad for one low price. Packages Factory sealed.

$19.95 4 DISK SET

KEYBOARDS

New and complete subassembly.
Easy internal replacement.

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

REPAIR MANUALS

SAMS Service Manuals for the following units contain schematics, parts listings, labelled photographs showing the location of checkpoints and more! A special section gives oscilloscope and logic probe readings allowing you to narrow the malfunction down to a specific chip or transistor!

800, 800XL, 130XE, 1050 and 400 .... $24.95 each
520ST Service Manual. $37.50

600XL 64K UPGRADE

Easy to install internal modification. Run Disk Drive and all 800XL software. Kit includes parts and instructions. Soldering required to install 3 jumpers......... $29.95

MISCELLANEOUS

13 Pin Serial I/O Cable	$4.50
ACE Joystick	$7.95
1050 Track Zero Sensor	$8.50
2793 1050 Controller IC	$19.50
U.S. Doubler	$29.95
400 Board Set (W/O Kybd)	$22.95
Paddle Controllers (Pair)	$9.95
Rev. "C" Basic ROM	$15.00
Rambo XL w/o RAM IC's	$39.95
850 or PR Modem Cable	$14.50
850 or PR Printer Cable	$14.50
Printer Interface	$43.95
I/O 13 Pin PC connector	$4.50
1027 INK ROLLER	$6.50
1030 Modem Power Pack	$6.95

CALL TOLL FREE
1-800-551-9995
IN CALIF. OR OUTSIDE U.S.
CALL 415-352-3787

CARTRIDGES

For XL's/XE's 800/400

POLE POSITION Cartridge	$15.00
STAR WARS - Return of the Jedi	$15.00
PAC-MAN Cartridge	$4.00
DELUXE INVADERS Cartridge	$4.00
JOURNEY TO THE PLANETS Cart.	$4.00
GORF Cart. 800/400 Only	$4.00
DONKEY KONG Cartridge	$5.00
STAR RAIDERS Cartridge	$5.00
MILLIPEDE Cartridge	$10.00
PLATTER MANIA Cart. by Epyx	$9.95
WIZARD OF WOR Cartridge	$9.95
CAVERNS OF MARS Cartridge	$14.50
PENGO Cartridge	$15.00
EASTERN FRONT Cartridge	$15.00
ATARIWRITER Cartridge	$29.95
TURMOIL Cartridge	$5.00
SPRINGER Cartridge	$7.50
ADVENTURE CREATOR Cartridge	$12.50
ZONE RANGER Cartridge	$12.50
SILICON WARRIOR Cartridge	$12.50
MATH ENCOUNTER Cartridge	$12.50
BASIC LANGUAGE Cartridge	$15.00
PILOT LANGUAGE PACKAGE	$17.50
SPARTADOS-X Cartridge	$59.95
ACTION O.S.S Cartridge	$59.95
ACTION TOOL KIT DISK	$25.00
MAC-65 O.S.S. Cartridge	$59.95
MAC-65 TOOL KIT DISK	$25.00
R-TIME 8 Cartridge	$56.95

SPINNAKER EDUCATIONAL CARTRIDGES

KINDERCOMP Cartridge	$8.75
FRACTION FEVER Cartridge	$8.75
FACEMAKER Cartridge	$8.75
ALPHABET ZOO Cartridge	$8.75
UP FOR GRABS Cartridge	$8.75
ALF IN THE COLOR CAVES Cart.	$8.75
DELTA DRAWING Cartridge	$8.75
STORY MACHINE 800/400 only	$8.75
LINKING LOGIC (Fisher-Price) Cart.	$8.75

HOT DISK TITLES

BEST SELLERS / LOW PRICES

• GHOSTBUSTERS	$9.95
• F-15 STRIKE EAGLE	$14.95
• TOP GUNNER COLLECTION	$14.95
• SILENT SERVICE	$14.95
• KENNEDY APPROACH	$14.95
• VIDEO TITLE SHOP (64K)	$12.50
• CHESSMASTER 2000	$12.50
• ALTERNATE REALITY THE CITY	$12.50
• ALTERNATE REALITY DUNGEON	$12.50
• TOMAHAWK (64K)	$12.50
• 221-B BAKER STREET (64K)	$12.50
• MUSIC STUDIO	$12.50
• STRATOS	$7.50

• • • • INFOCOM ADVENTURES • • • • • •

• WISHBRINGER	$14.95
• MOONMIST	$14.95
• SPELLBREAKER	$14.95
• SEA STALKER	$14.95
• HOLLYWOOD HIJINX	$14.95
• STATION FALL	$14.95
• INFIDEL	$14.95
• LURKING HORROR	$14.95
• CUT THROATS	$14.95
• BALLYHOO	$14.95
• SUSPENDED	$14.95
• WITNESS	$14.95
• HITCH HIKERS GUIDE	$14.95
• ZORK 1	$14.95

COMPUTER BOOKS

ATARI PLAYGROUND WORKBOOK	$5.95
HACKERBOOK TRICKS & TIPS	$5.00
INSIDE ATARI BASIC	$5.00
ATARI BASIC REF. MANUAL	$5.00
HOW TO ATARI 6502 PROGRAM	$5.00
WRITE YOUR OWN GAMES	$5.00
PROGRAMMERS REF. GUIDE	$14.95
ASSEMBLY LANGUAGE GUIDE	$21.95
XE USERS HANDBOOK	$21.95
XL USERS HANDBOOK	$21.95
ATARI BASIC FASTER & BETTER	$22.95
YOUR ATARI COMPUTER	$17.95

SERIAL I/O CABLE

High Quality, 13Pin $4.50
Special ... 3 Cables for $10.00

MAC-65 CARTRIDGE

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

PR: CONNECTION

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

SUPRA MODEM

Supra 300 Baud Direct Connect Modem requires no separate interface. Includes cable and Disk Software. Supports X-Modem, Download & Upload...... $29.95

Disks-XL's/XE's/800

GHOSTBUSTERS Disk	$9.95
F-15 STRIKE EAGLE Disk	$14.95
TOP GUNNER COLLECTION	$14.95
SILENTSERVICE Disk	$14.95
KENNEDY APPROACH Disk	$14.95
NATO COMMANDER Disk	$14.95
S.A.T. MATH	$17.95
S.A.T. VERBAL	$17.95
VISICALC Spreadsheet	$19.95
BOOKKEEPER & CX85 KEYPAD	$19.95
DAVID'S MIDNIGHT MAGIC	$4.95
TECHNA-COLOR DREAM Graphics	$4.95
ALIEN AMBUSH	$4.95
REAR GUARD	$4.95
ROCKET REPAIRMAN Disk	$4.95
SPIDERMAN Disk	$5.00
HUMAN TORCH & THE THING	$5.00
MUSICAL PILOT Ed. Disk	$5.00
CON'PUTATION Disk	$5.00
MASTER CHESS Disk	$5.00
SPEED KING Disk	$5.00
LAST V-8 Disk	$5.00
PATHFINDER Disk	$5.00
MATCH RACER Disk	$5.00
WOMBATS 1 Adventure Disk	$5.00
FAMILY FINANCE Disk by Atari	$6.50
FORT APOCALYPSE Disk	$7.50
NINJA Disk	$7.50
ELECTRA-GLIDE Disk	$7.50
THE GAMBLER Disk	$7.50
MIND MAZES (Educational)	$7.50
VIDEO VEGAS Poker & Slots	$9.95
ACTION BIKER	$9.95
KICK START	$9.95
MONEY TOOLS (Financial Utility)	$9.95
DIG DUG Disk	$9.95
REPTON Disk	$9.95
PUZZLE PANIC Disk by Epyx	$9.95
CEST'E LA VE Disk	$9.95
WOMPER STOMPER Disk	$9.95
OLIN EMERALD (Jr., Adv.)	$9.95
MOVIE MAKER (Graphics)	$9.95
FINANCIAL COOKBOOK	$9.95
CYCLE KNIGHT	$9.95
SONG WRITER	$9.95
COUNTING & ADDING with Tink	$9.95
CHESSMASTER 2000	$12.50
BRIDGE 4.0	$12.50
PEGAMMON	$12.50
MUSIC STUDIO	$12.50
HOTEL ALIEN	$12.50
FOOBLIKSKY Disk / Board Game	$12.50
L. A. SWAT	$12.50
GRANDMA'S HOUSE (Educational)	$12.50
CASTLE WOLFENSTEIN Disk	$12.50
BEYOND CASTLE WOLFENSTEIN	$12.50
TEMPLE OF APSHAI	$12.50
ALI-ANTS Disk	$12.50
MOUSE QUEST Disk (64K)	$12.50
ELECTRONIC DRUMMER Disk	$12.50
STOCK MARKET Game	$12.50
PIRATES OF THE BARBARY COAST	$12.50
NICKERBOCKER Disk	$12.50
BOULDER DASH Construction Set	$12.50
MIND SHADOW Adventure 64K	$14.95
HACKER Adventure	$14.95
SEA STALKER Adventure	$14.95
CONFLICT IN VIETNAM.	$14.95
DECISION IN THE DESERT	$14.95
CRUSADE IN EUROPE	$14.95
SPITFIRE ACE	$14.95
SSI Rebel Charge at Chickamauga	$14.95
SSI SHILOH	$14.95
SSI THE ETERNAL DAGGER	$14.95
SSI 50 MISSION CRUSH	$14.95
SSI BATTALION COMMANDER	$14.95
SSI PANZER GRENADIER	$14.95
SSI FIELD OF FIRE	$14.95
SSI GEMSTONE WARRIOR	$14.95
SSI PHANTASIE II	$19.95
SSI SONS OF LIBERTY	$19.95
SSI U.S.A.A.F.	$19.95
SSI WARSHIP	$19.95
MASTERTYPE (Typing Tutor)	$15.00
BANK STREET WRITER	$17.50
CUT & PASTE W.P. Disk	$17.50
SPARTADOS CONST. SET	$39.50
SPARTADOS Tool-Kit	$32.95
LJK DATA PERFECT Database	$CALL

DATASOFT DISKS

VIDEO TITLE SHOP 3 DISK BONUS PACK	
Contains VTS + 2 Companion Disks	$14.95
ALTERNATE REALITY (The City)	$12.50
ALTERNATE REALITY The Dungeon	$12.50
ZORRO Disk	$9.95
SARACEN Disk	$9.95
CROSSCHECK Disk	$9.95
GUNSLINGER Disk (64K)	$9.95
TOMAHAWK Helicopter Game 64K	$12.50
THEATER EUROPE War Game	$12.50
BRUCE LEE Disk	$12.50
MERCENARY Disk	$12.50
MERCENARY 2ND CITY DISK	$12.50
THE GOONIES Disk	$12.50
CONAN Disk	$12.50
NEVER ENDING STORY (64K)	$12.50
221-B BAKER STREET (64K)	$12.50
221-B BAKER ST. Case Library 1	$9.95
221-B BAKER ST. Case Library 2	$9.95
Napolean in Russia (Borodino 1812)	$12.50
BISMARCK	$12.50
MOON SHUTTLE	$9.95

AMERICAN TECHNA-VISION

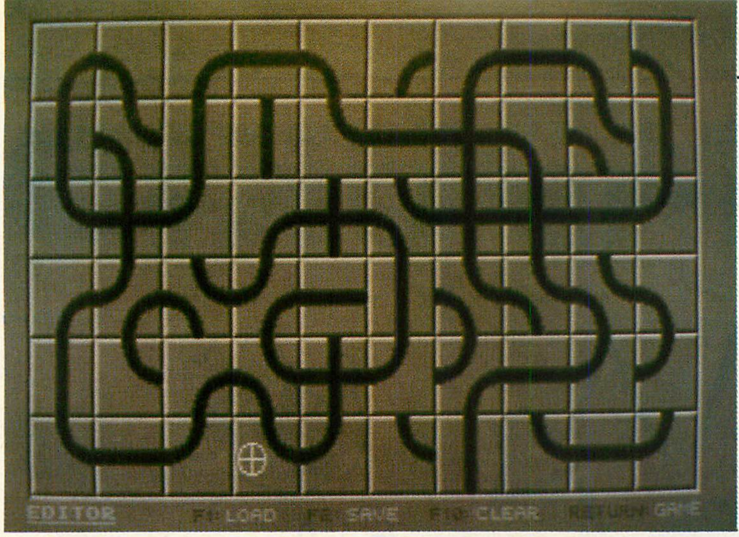
Mail Order: 15338 Inverness St., San Leandro, Ca. 94579
Sales Office: 2098 Pike Ave., San Leandro, Ca. 94577

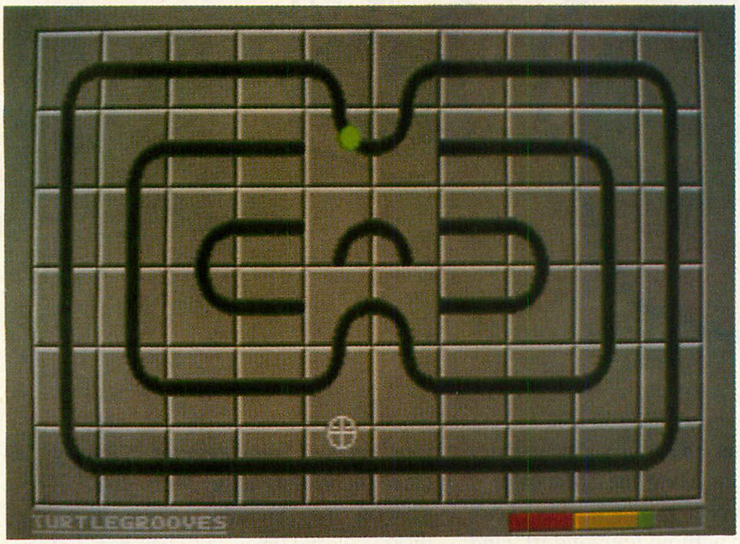
Terms: NO MINIMUM ORDER. We accept money orders, personal checks or C.O.D.s. VISA, Master/Card okay. Credit cards restricted to orders over $18.50. No personal checks on C.O.D. - Shipping: $4.00 shipping and handling on orders under $150.00. Add $2.75 for C.O.D. orders. In Canada total $6.00 for shipping and handling. Foreign shipping extra. Calif. residents include 7% sales tax. All products new and guaranteed. All sales final..

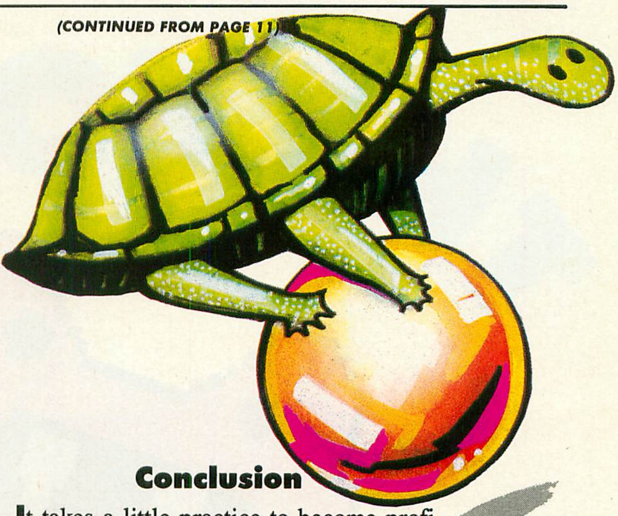
Prices subject to change without notice. Send SASE for free price list. Atari is a reg. trademark of Atari Corp.

CIRCLE #116 ON READER SERVICE CARD.

(CONTINUED FROM PAGE 11)





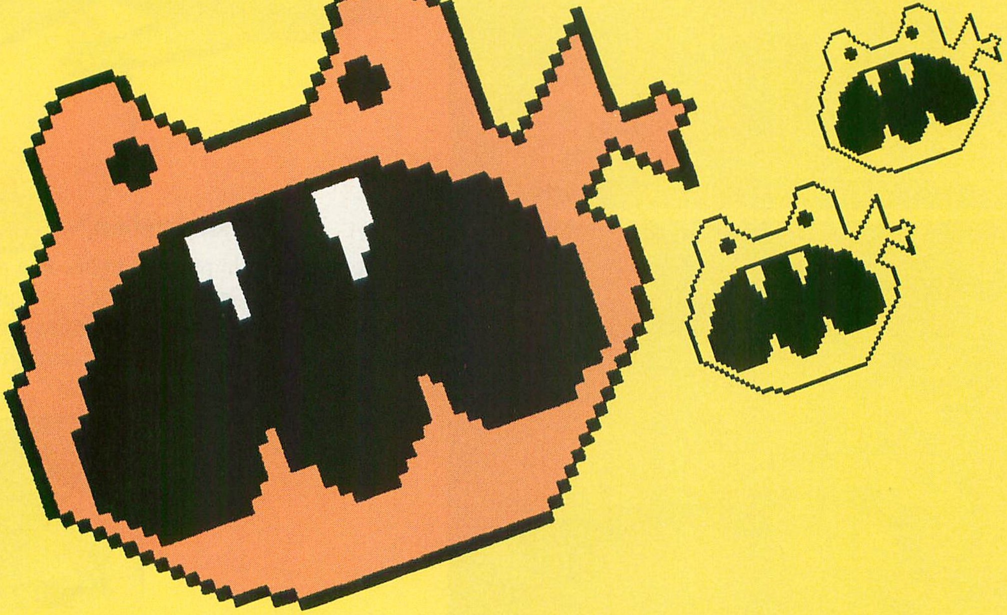


Conclusion

It takes a little practice to become proficient at *Turtle Grooves*, but once you play it a few times, you'll start to recognize patterns and learn the best tile-rotating strategies to use. Keep those balls rolling!

Greg Knauss is not 21, not attending the University of California, San Diego, and not telling the truth.

THE COMPUKID



One thing that makes the ST an ideal computer for children and preferable to many others on the market is its use of plastic-encased 3.5-inch disks. While more expensive than other floppies, these disks are very durable and can be handled by children with relatively little likelihood of damage.

Even so, there are still some hazards to disks and the data they contain, and it helps to make children aware of them. The major hazards include dust, liquid, magnetic fields and heat; but in the hands of children, additional threats to data can arise. A few simple rules and some education can do much to prevent damage.

An obvious way to protect the disk against dust accumulation is to keep the metal "window" on the disk closed at all times. I advise young children not to touch the metal parts of the disk at all. When observed, these precautions protect the disk not only from dust, but from prying fingers and probing ballpoints as well. Children can be curious about the contents of the plastic disk case, however, and sometimes this curiosity gets the better of them. If you have a disk that cannot be reused, instead of just tossing it in the trash, pry open the case and show your child what's inside.

Drinks and food are banned from the computer room at our house. (This protects the disks and the computer.) It is also a good idea to insist that children wash and dry their hands before using the computer.

My oldest son once boasted, "I bet I can pick up that disk without touching it!" and he proceeded to do so—using a magnet. Of course, the data on the disk was ruined. Children enjoy experimenting with magnets, but the computer room is not the place to do this. Although it is impossible to eliminate all hazards from magnetic fields (there are magnets in your TV set, in your vacuum cleaner, and even in your ST's monitor), making children aware of the danger will do much to reduce the risk.

There are other ways to protect data as well. The 3.5-inch disk has a little hole in the upper right-hand corner. A plastic tab can be slid up or down to open or close this hole. When the hole is open, data cannot be deleted from the disk. Nor can data be written to disk, and thus this procedure is called "write-protection." It is a good idea to write-protect all children's disks unless a program on the disk allows the child to save data. Children should also know that they must not remove a disk from the drive while the drive's "busy" light is on. The busy light is red like a stop sign and, at our house, it means "Stop!"

Adults protect their disks by keeping them in a disk box or wallet. It's a good idea for children to have a special dry, protective place to store their disks too. Adults use various kinds of filing systems to catalog, locate and differentiate their disks. These systems may include the use of colored disks, colored labels and disk-box dividers. Disks may be sorted according to the type of application or by a numbering system or both. Children can benefit from some disk organization too, and some of these methods will prove useful. Effective labeling, in particular, will help your child identify the disk he or she wants.

The single most important factor in the easy identification of disks is that each be distinctively different from the others. A group of identically colored disks with labels prepared in exactly the same way and with the same typeface, etc., makes for a handsome disk box, but not a very practical one. Using a variety of colored labels and colored disks with different handwriting or pen colors on the labels can make the disks more distinct from one another and thus make it easier to identify a particular disk quickly.

Nonreading children will especially appreciate pictures on disk labels. For example, a picture of a shark tells my daughter that the disk contains her favorite matching game, *Seaside* (by Frank M. Hundley, October '89 ST-LOG). A picture of a Christmas tree identifies the disk with *O Tannenbaum!* (by this author, November '89 ANALOG).

CONNECTION: DREIDEL

ST

BY D.A. BRUMLEVE

The simplest way to provide pictures of this kind is, of course, to draw them with a marker on a blank label. This can be effective in differentiating disks, especially when colored permanent markers are used. The resulting disk label withstands handling and, depending on the abilities of the artist, can be attractive as well as useful.

The director of my youngest's preschool had asked for picture labels with large-type titles and arrows indicating the proper position of the disk when inserting it into the drive. I used a paint program to draw icons for the pictures and then loaded them into a desktop-publishing program to add the title and arrow. It would also be possible to use a paint program such as *DEGAS* for the entire process. The resulting labels were printed out on an 8½″×11″ sheet (with several labels to a page). I then took the sheet to a copy shop where it was duplicated on sticker-backed paper. The labels were then cut out and mounted on the classroom's disks. While the ink from labels printed on a printer sometimes will smear, copier "ink" normally does not, so the resulting labels are hardy, even in the hands of children.

Instead of going to a copy shop, one could also cut the labels from the sheet of ordinary printer paper and mount them on the disks using clear Contac paper, available at very little cost from art and educational supply stores and from discount stores where shelf paper is sold. This thin plastic material, with one smooth side and one sticky side, is placed (sticky side down) over the top of the paper label so that it overhangs the edges on all four sides. The still-exposed sticky edges of the contact paper are then adhered to the disk itself. This makes for an even more durable label, one that is likely to outlast the disk itself.

The labeling method outlined above will be much easier to implement if each of the child's disks is limited to a single program. In fact, this is exactly what I'd recommend for children who are too young to use the ST's desktop. One of the best ways to promote independent use of the computer and boost your child's ego is to provide self-booting program disks. Each such disk contains only one program. When the self-booting disk is placed in Drive A and the computer is turned on or rebooted, the program on the disk is automatically loaded into memory and play can begin. The child need not open a program from the desktop, and the procedure is so simple that the child doesn't need to ask a parent for help.

A public-domain program (STARTGEM.PRG by Rick Flashman) that will allow you to provide such self-booting disks is included in the ARCed file with this month's kidprog, *Dreidel*. Setting up a self-booting disk requires a few simple steps. If you are unfamiliar with this process, please refer to the accompanying sidebar entitled "Using STARTGEM."

Using STARTGEM

(Please note that the procedure described below for preparing self-booting disks is intended for use when the computer is booted from Drive A.)

In order to use STARTGEM.PRG, you'll need to create an AUTO folder on your child's program disk. Place the disk in Drive A and double-click on the Drive A icon on your ST's desktop. The directory window will open. Then select New Folder from the File menu. A dialog box will appear, asking for the name of the folder. Type AUTO and press Return. A new folder named AUTO will appear in your disk's directory. Now copy STARTGEM.PRG to the AUTO folder.

When the disk is booted, STARTGEM.PRG is loaded immediately. The program causes the computer to look for a file named STARTGEM.INF in the root directory (not in a folder) of the boot disk. The STARTGEM.INF file, in turn, gives the computer the name and directory of the program you want to run on your child's self-booting disk.

A STARTGEM.INF file for this month's kidprog is included in the ARCed file, so if you are creating a self-booting disk for this selection, simply copy STARTGEM.INF to the root directory of your *Dreidel* disk. To create STARTGEM.INF files for other programs, you will need to use a word processor that can make ASCII-format text files. I use *1st Word* for this purpose. STARTGEM.INF must contain the pathname of the file you want automatically loaded; so if the disk is being prepared to load DRAW_IT.PRG automatically, and DRAW_IT.PRG is in the root directory of Drive A, you would type:

```
A:\DRAW_IT.PRG
```

If DRAW_IT.PRG is in the DRAWINGS folder, you would type:

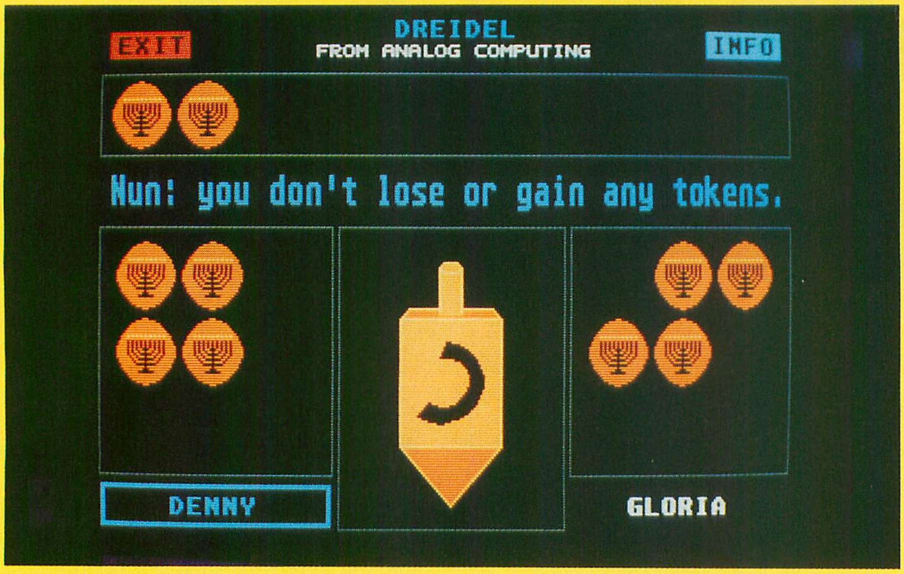
```
A:\DRAWINGS\DRAW_IT.PRG
```

Do not press Return after you have typed the line. Save the file in the root directory of your self-booting disk in ASCII format with the name STARTGEM.INF. (To save an ASCII file in *1st Word*, you need to make certain that WP mode in the Edit menu is not selected.)

The STARTGEM.INF provided in DREIDEL.ARC contains the following line:

```
A:\DREIDEL.PRG
```

STARTGEM.PRG will automatically load *Dreidel* when DREIDEL.PRG is in the root directory of the disk in Drive A and the computer is turned on or rebooted.



DREIDEL
FROM ANALOG COMPUTING

Nun: you don't lose or gain any tokens.

Once a disk has been prepared for self-booting from Drive A, it is possible to provide additional protection from unwanted damage to data. On disks used by my youngest children, I remove the drive icons from the desktop and save the DESKTOP.INF file. To do this, place the self-booting disk in Drive A. Click once on the Drive A icon on the desktop so that it is highlighted in black, and then select the Install Disk Drive option from the Options menu. A dialog box will appear. Choose "Remove." Repeat the process with the Drive B icon and any other drive icons that may be on the desktop. Then choose Save Desktop from the Options menu. If the child using this disk exits the program and returns to a desktop, there will be no directory window and no drive icons, so the child will not be able to drag data to the trash can or format the disk.

This Month's Program

Dreidel (pronounced "dray-dle") is a game of chance traditionally played during Hanukkah, the Jewish "Celebration of Candles." The dreidel itself is a spinning top with four sides. Each side is decorated with a Hebrew letter (nun, gimmel, heh and shin). These initials stand for the message "nais gadol hayah sham," which means "a great miracle happened there." The dreidel may be made from any number of materials, including cardboard, wood and clay.

A version of Dreidel for your Atari ST is on this month's disk. While this program is aimed at young readers, it can be played by nonreaders. It may also be enjoyed by adults; however, lest your expectations exceed the program's potential, I ought to point out that success (or failure) in this game requires no strategy whatsoever. The outcome is determined solely by chance.

Dreidel is a gambling game, so, as in poker, players contribute tokens (or coins, or peanuts, or whatever they are playing with) to a "pot." There are many versions of the game, but in the one presented here, the pot must always have at least two tokens. Each player contributes a token at the beginning of a round. Players take turns spinning the dreidel. When the dreidel stops spinning, it falls over onto one of its sides. The letter on the side facing up determines the outcome of the spin, as follows:

nun—the player does nothing (no tokens are gained or lost).

gimmel—the player takes all the tokens in the pot.

heh—the player takes half the tokens in the pot, or, if the pot contains an odd number of tokens, the player takes half the tokens plus one more.

shin—the player puts two tokens in the pot.

The object of the game, of course, is to win as many tokens as you can. The game ends when one player must put more tokens in the pot than he has in his own pile, or when one player has all the tokens.

Getting Started

To try your luck at *Dreidel*, first follow the disk instructions to deARC the file DREIDEL.ARC. The ARCed file contains DREIDEL.DAT, SPIN.DAT, DREIDEL.PRG and DREIDEL.LST. (DREIDEL.LST contains the GFA BASIC 2.0 source code for the program; it will be needed only if you want to examine the program's construction.)

Copy DREIDEL.PRG, DREIDEL.DAT and SPIN.DAT onto a freshly formatted disk. (For your convenience in creating an autobooting disk as described above, the files STARTGEM.PRG and STARTGEM.INF are also provided. If you want to create an autobooting *Dreidel* disk, also copy STARTGEM.INF to the disk, then create an AUTO folder as explained in the sidebar accompanying this article and copy STARTGEM.PRG into that folder.)

If autobooting is not desired, the program will run from a folder, so long as the .DAT files are in the same directory as the program. The program can be run from a hard drive. *Dreidel* requires a color monitor in low resolution.

Playing the Game

Double-click on DREIDEL.PRG to run the program. (If you have prepared the disk for autobooting, simply put the disk in Drive A and reboot your computer.) A title screen will introduce you to the dreidel and the names of each of the four letters. A little tune is played. You will hear this tune whenever a player wins a game. Children may enjoy singing the dreidel song whenever it is played. The words are as follows:

Dreidel, dreidel, dreidel,
I made it out of clay.
And when my dreidel's ready,
Oh, dreidel I will play!

Following the title screen, a white box appears with a request for the names of each of two players. Type each of the players' names, pressing Return after each.

The main screen is divided into several areas. EXIT and INFO boxes appear at the top corners of the screen. Dark grey boxes separate the areas below them. The top box represents the pot; it is empty as the game begins. Below the pot is the instructions area; it is here that you will see messages telling each player what to do next. Lower on the screen are three grey boxes in a row. The leftmost box holds the first player's pile of tokens. The box on the right holds the second player's tokens. The names of the

SAMPLE DISK LABEL FOR CHILDREN

players are written below the token boxes. The middle box holds the dreidel itself.

As the game begins, each player holds five tokens. Each of the players is advised in turn to put a token in the pot. This is done by clicking the mouse on a token in the player's pile. The token is automatically moved to the pot.

The player to begin is chosen at random by the computer. That player then spins the dreidel by clicking the mouse in the dreidel area. It will spin a random number of times and then fall over on one side. The instructions area will announce the outcome of the spin. If the spin is a gimmel or a heh, the player moves tokens from the pot to his own pile. This is done by clicking the mouse directly on the tokens in the pot; they are then automatically moved to the current player's pile. If the spin results in a nun, the player does nothing. If the spin is a shin, the player must move two tokens from his own pile to the pot.

Whenever the number of tokens in the pot drops below two, both players must add a token. If a player's pile runs out of tokens, the player may still spin the dreidel, but if the outcome of the spin requires him to put tokens in the pot, he will lose the game.

Concessions to Nonreaders

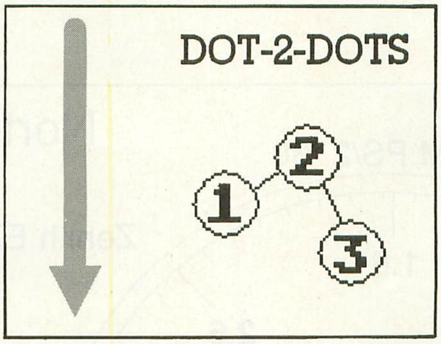
This computer version of Dreidel has many written instructions that advise players of their next move. Several techniques have been employed to provide other kinds of cues for nonreaders, making the game accessible to younger children as well.

Each player's name is associated with a color. The name of the player on the left is written in blue, and all instructions for that player are blue as well. The name of the player on the right and instructions for that player are written in white. When it is time for the player on the left to take action, his name is surrounded by a thick blue box at the bottom left-hand corner of the screen. When it is the turn of the player on the right, his name is boxed in thick white lines. Information in gold text (or multiple colors) appears in the instructions area to report the game's progress to both players at once.

Whenever action must be taken with the mouse, a box (with thinner lines) appears around the area in which the mouse must be clicked (the player's pile, the pot, or the dreidel). The box will be of the color that is associated with the current player. Thus, when the player on the right must take tokens from the pot, a white box appears around the pot. When the player on the left must spin the dreidel, a blue box is drawn around the dreidel.

All selections with the mouse can be made by clicking either mouse button or both mouse buttons simultaneously. The mouse (in the shape of a hand) is available on the screen only when it can be used.

Representations of actual tokens have been used to indicate the number of tokens in the pot and in the players' piles. Very young children do not need to recognize a "7" to know that they have seven tokens in their pile. Children may enjoy counting the tokens and moving them. Care has been taken to ensure that a child can choose any token in his or her pile when putting tokens in the pot; likewise, any token in the pot can be moved to the player's pile. There is no need to learn to click on the tokens in a particular sequence.



SAMPLE DISK LABEL FOR CHILDREN

For the benefit of young readers, and especially of nonreaders, the computer will place tokens in the pot and in the piles from left to right; that is, the first token to be placed in the pot will be placed at the far left in the grey pot box. The resulting practice with left-to-right eye movements aids the development of prereading skills in young children.

Sounds provide an additional cue that a player is out of tokens or has won the game. A low sound lets the player know he is clicking the mouse in the wrong area of the screen.

When the program asks for a name, the player may follow the instructions and type in his or her name and press Return, or may simply press Return. If the players don't type their names, the computer will name them PLAYER 1 and PLAYER 2. Many young children enjoy typing their names. Those who don't or can't can identify the Return key easily.

While intended for two players, the game can also be enjoyed by one player (who takes both roles). Two-year-olds who do not even understand the concept of a game may prefer to use the program in this way.

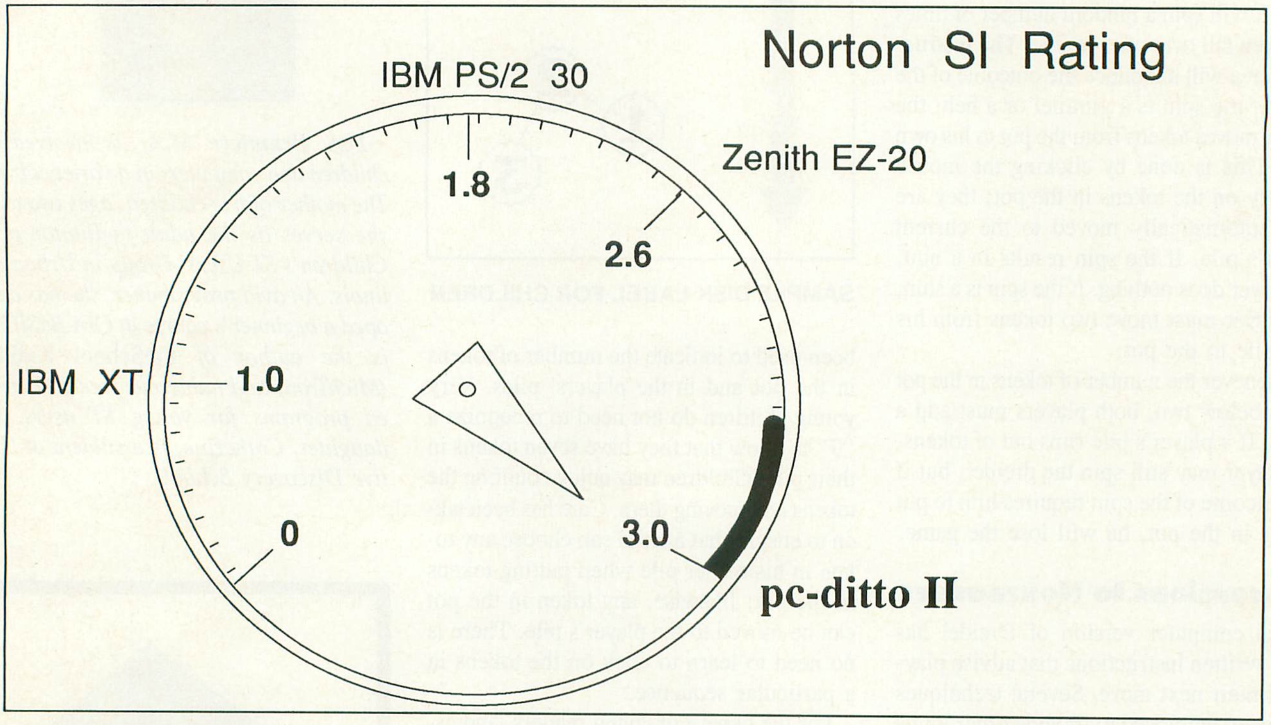
If your preschooler is made aware of these cues, he or she will be able to compete on an equal footing with older siblings or adults.

I hope you'll enjoy *Dreidel* with your children this holiday season!

D.A. Brumleve, M.A., is involved with children and computers in a variety of ways. The mother of five children, ages two to ten, she serves as the adult facilitator of the Children's ST Users' Group in Urbana, Illinois. An avid programmer, she has developed a beginner's course in GFA BASIC and is the author of PreSchool KidProgs *(MichTron) and numerous freely distributed programs for young ST users. Her daughter, Catherine, is a student at Creative Discovery School.*

pc-ditto II

By Avant-Garde Systems



IBM <u>XT</u> COMPATIBLE !

IBM <u>AT</u> PERFORMANCE !

ATARI <u>ST</u> PRICE !

$299.95

Now! Run the most popular IBM software on your Atari ST...

FAST !

See pc-ditto II at your local Atari dealer or write for free information.

Yes. Please send me more information !

Avant-Garde Systems, 381 Pablo Point Drive
Jacksonville, Florida 32225

Name____________________________

Address__________________________

City____________ **St**______ **Zip**________

CIRCLE #117 ON READER SERVICE CARD.

M/L EDITOR

> The two-letter checksum code preceding the line numbers here is *not* a part of the BASIC program. For more information, see the "BASIC Editor II" elsewhere in this issue.

M/L Editor provides an easy method with which to enter our machine-language listings. It won't allow you to skip lines or enter bad data. For convenience, you may enter listings in multiple sittings. When you're through typing a listing with M/L Editor, you'll have a complete runnable object file on your disk.

There is one hitch: It's for disk users only. My apologies to those with cassette systems.

Listing 1 is M/L Editor's BASIC listing. Type it in and, when it's free of typos, save a copy to disk, then run it.

On a first run, you'll be asked if you're starting a new listing or continuing from a previously saved point. Press S to start or C to continue.

You'll then be asked for a filename. If you're starting a new listing, type in the filename you want to save the program under, then press Return. If there's already a file by that name on the disk, you'll be asked if you wish to delete it. Press Y to delete the file or N to enter a new filename.

If you're continuing a file, type in the name you gave the file when you started it. If the program can't find the file, you'll get an error message and be prompted for another filename. Otherwise, M/L Editor will calculate where you left off, then go on to the data-entry screen.

Each machine-language program in ANALOG Computing is represented by a list of BASIC data statements. Every line contains 16 bytes, plus a checksum. Only the numbers following the word DATA need to be considered.

M/L Editor will display, at the top of the screen, the number of the line you're currently working on. As you go through the line, you'll be prompted for each entry. Simply type the number and press Return. If you press Return without a number, the default is the last value entered.

This feature provides a quick way to type in lines with repetitions of the same number. As an added convenience, the editor will not respond to the letter keys (except Q for "quit"). You must either enter a number or press Return.

When you finish a line, M/L Editor will compare the entries' checksums with the magazine's checksum. If they match, the screen will clear, and you may go on to the next line.

If the checksums *don't* match, you'll hear a buzzing sound. The screen will turn red, and the cursor will be placed back at the first byte of data. Compare the magazine listing byte by byte with your entries. If a number is correct, press Return.

If you find an error, make the correction. When all data is valid, the screen will return to gray, and you'll be allowed to begin the next line.

Make sure you leave your disk in the drive while typing. The data is saved continuously.

You may stop at any time (except when you have a red screen) by entering the letter Q for byte 1. The file will be closed, and the program will return you to BASIC. When you've completed a file, exit M/L Editor in the same way.

When you've finished typing a program, the file you've created will be ready to run. In most cases, it should be loaded from DOS via the L option. Some programs may have special loading instructions; be sure to check the program's article.

If you want the program to run automatically when you boot the disk, simply name the file AUTORUN.SYS (make sure you have DOS on the disk).

LISTING 1: BASIC LISTING

```
AZ 10 DIM BF(16),N$(4),A$(1),B$(1),F$(15)
   ,F1$(15)
LF 11 DIM MOD$(4)
BN 20 LINE=1000:RETRN=155:BACKSP=126:CHKS
   UM=0:EDIT=0
GO 30 GOSUB 450:POSITION 10,6:? "Start or
    Continue? ";:GOSUB 500:? CHR$(A)
ZG 40 POSITION 10,8:? "FILENAME";:INPUT F
   $:POKE 752,1:? " "
FE 50 IF LEN(F$)<3 THEN POSITION 20,10:?
   "  ":GOTO 40
MF 60 IF F$(1,2)<>"D:" THEN F1$="D:":F1$(
   3)=F$:GOTO 80
KL 70 F1$=F$
TN 80 IF CHR$(A)="S" THEN 120
FD 90 TRAP 430:OPEN #2,4,0,F1$:TRAP 110
HQ 100 FOR X=1 TO 16:GET #2,A:NEXT X:LINE
   =LINE+10:GOTO 100
WM 110 CLOSE #2:OPEN #2,9,0,F1$:GOTO 170
VT 120 TRAP 160:OPEN #2,4,0,F1$:GOSUB 440
   :POSITION 10,10:? "FILE ALREADY EXISTS
   !!":POKE 752,0
ZU 130 POSITION 10,12:? "ERASE IT? ";:GOS
   UB 500:POKE 752,1:? CHR$(A)
VH 140 IF CHR$(A)="N" OR CHR$(A)="n" THEN
    CLOSE #2:GOTO 30
QG 150 IF CHR$(A)<>"Y" AND CHR$(A)<>"y" T
   HEN 130
BH 160 CLOSE #2:OPEN #2,8,0,F1$
IE 170 GOSUB 450:POSITION 10,1:? "NOW ON
   LINE: ";LINE:CHKSUM=0
GH 180 L1=3:FOR X=1 TO 16:POSITION 13*(X<
   10)+12*(X>9),X+2:POKE 752,0:? "BYTE #"
   ;X;": ";:GOSUB 310
KH 190 IF EDIT AND L=0 THEN BYTE=BF(X):GO
   TO 210
FY 200 BYTE=VAL(N$)
OZ 201 MOD$=N$
BU 210 POSITION 22,X+2:? BYTE;"  "
YZ 220 BF(X)=BYTE:CHKSUM=CHKSUM+BYTE*X:IF
    CHKSUM>9999 THEN CHKSUM=CHKSUM-10000
MS 230 NEXT X:CHKSUM=CHKSUM+LINE:IF CHKSU
   M>9999 THEN CHKSUM=CHKSUM-10000
IG 240 POSITION 12,X+2:POKE 752,0:? "CHEC
   KSUM: ";:L1=4:GOSUB 310
EW 250 IF EDIT AND L=0 THEN 270
QM 260 C=VAL(N$)
SY 270 POSITION 22,X+2:? C;"   "
IL 280 IF C=CHKSUM THEN 300
DI 290 GOSUB 440:EDIT=1:CHKSUM=0:GOTO 180
LW 300 FOR X=1 TO 16:PUT #2,BF(X):NEXT X:
   LINE=LINE+10:EDIT=0:GOTO 170
FV 310 L=0
KZ 320 GOSUB 500:IF (A=ASC("Q") OR A=ASC(
   "q")) AND X=1 AND  NOT EDIT THEN 420
PO 330 IF A<>RETRN AND A<>BACKSP AND (A<4
   8 OR A>57) THEN 320
DX 331 IF A=RETRN AND N$="" THEN N$=MOD$
TD 335 IF A=RETRN AND L=0 AND X>1 THEN 35
   0
JR 340 IF ((A=RETRN AND  NOT EDIT) OR A=B
   ACKSP) AND L=0 THEN 320
DW 350 IF A=RETRN THEN POKE 752,1:? " ":R
   ETURN
GG 360 IF A<>BACKSP THEN 400
SA 370 IF L>1 THEN N$=N$(1,L-1):GOTO 390
AS 380 N$=""
RE 390 ? CHR$(BACKSP);:L=L-1:GOTO 320
BB 400 L=L+1:IF L>L1 THEN A=RETRN:GOTO 35
   0
WK 410 N$(L)=CHR$(A):? CHR$(A);:GOTO 320
KN 420 GRAPHICS 0:END
YT 430 GOSUB 440:POSITION 10,10:? "NO SUC
   H FILE!":FOR X=1 TO 1000:NEXT X:CLOSE
   #2:GOTO 30
FD 440 POKE 710,48:SOUND 0,100,12,8:FOR X
   =1 TO 50:NEXT X:SOUND 0,0,0,0:RETURN
MY 450 GRAPHICS 23:POKE 16,112:POKE 53774
   ,112:POKE 559,0:POKE 710,4
XR 460 DL=PEEK(560)+256*PEEK(561)+4:POKE
   DL-1,70:POKE DL+2,6
HW 470 FOR X=3 TO 39 STEP 2:POKE DL+X,2:N
   EXT X:FOR X=4 TO 40 STEP 2:POKE DL+X,0
   :NEXT X
ZW 480 POKE DL+41,65:POKE DL+42,PEEK(560)
   :POKE DL+43,PEEK(561):POKE 87,0
AC 490 POSITION 2,0:? "analog ml editor":
   POKE 559,34:RETURN
MZ 500 OPEN #1,4,0,"K:":GET #1,A:CLOSE #1
   :RETURN
```

(CONTINUED FROM PAGE 56)

portant niche in the DTP world. SoftLogik and Magnetic Images support *PageStream* at affordable prices, with excellent fonts for dot-matrix and laser printers. Moreover, these packages are generally available through ST dealers, though not at a great discount since their price point is low (compared to the $75 or more for one font family used with *UltraScript* or *Calamus*). Your own GDOS fonts can be created with Neocept's excellent *Fontz!*, which allows you to convert Macintosh and Amiga fonts as well as edit and create GDOS fonts.

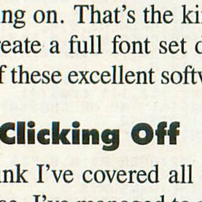
The *Calamus Font Editor* is a sophisticated editor that creates vector fonts, which are scalable. Unlike GDOS fonts, which come in specific sizes (one per file), scalable fonts can use one font file for a wide variety of sizes. If you use *Calamus*, this desk accessory is a must and includes 24 designer fonts on a second double-sided disk.

I should caution you that font editing is not for everyone, so try each of these programs out and see if they're really the kind of program you want to spend days and weeks working on. That's the kind of time it takes to create a full font set despite the ease of use of these excellent software tools.

Clicking Off

Well, I think I've covered all the bases. If nothing else, I've managed to provide an overview of what's available for DTP work on the ST. While I've given you my own preferences as to DTP software, I fully intend to provide coverage of tips and tricks for using all such programs in the months to come, so please be patient if I haven't touched on the program you've chosen to work with.

Next month I'll show you some terrific text-effects work from *PageStream*, including tips and tricks on using macros. Happy holidays and best wishes for the coming year. ■



***Donavan Vicha** has been writing about desktop publishing on the Atari ST for three years. He uses a Mega ST2 system for his freelance editorial service, transferring 12 years of book-publishing experience to this rapidly growing field. He lives in Chicago with his wife and two sons.*

Products mentioned in this article:

The FinalWord
Mark of the Unicorn
222 Third Street
Cambridge, MA 02142
(617) 576-3052

UltraScript
Imagen Corporation
P.O. Box 58101
Santa Clara, CA 95052-8101
(800) 635-3997

SyQuest 44 Hard Disk
Carter Graphics & Computers
21 North Main Street
Logan, UT 84321
(801) 753-1399

Calamus
ISD Marketing, Inc.
2561 John Street, Unit #3
Markham, Ontario L3R 2W5
(416) 479-1880

PageStream
SoftLogik Publishing Corporation
11131F South Towne Square
St. Louis, MO 63123
(314) 894-8608

Word Writer
Desktop Publisher ST
Timeworks, Inc.
444 Lake Cook Road
Deerfield, IL 60015-4919
(312) 948-9200

Fleet Street Publisher
MichTron
576 South Telegraph
Pontiac, MI 48053
(313) 334-5700

Easy Draw 3.0
Easy Tools
OSpooler
Touch Up 1.5
Hand Scanner
Migraph, Inc.
200 South 333rd, Suite 220
Federal Way, WA 98003
(800) 223-3729

ST Scan
Navarone Industries, Inc.
454 Kenneth Avenue
Campbell, CA 95008
(800) 624-6545

WordPerfect
WordPerfect Corp.
81 North State
Orem, UT 84057
(800) 321-3271

Universal Item Selector II
Application & Design Software
226 NW "F" Street
Grants Pass, OR 97526
(503) 476-0071

WordUp 2.0
Neocept, Inc.
547 Constitution #A
Camarillo, CA 93010
(805) 482-4446

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

G+Plus
CodeHead Software
P.O. Box 74090
Los Angeles, CA 90004-4090
(213) 386-5735

The Best Mouse
Best Electronics
2021 The Alameda, Suite 290
San Jose, CA 95126
(408) 243-6950

Datel Mouse
Datel Computers
3430 E. Tropicana #67
Las Vegas, NV 89121
(800) 782-9110

NeoDesk 2.0
Gribnif Software
P.O. Box 350
Hadley, MA 01035
(413) 584-7887

PictaScan
E. Arthur Brown Company
3404 Pawnee Drive
Alexandra, MN 56308
(612) 782-8847

IMG Scan
SeymorRadix
P.O. Box 166055
Irving, TX 75016
(214) 255-7490

(CONTINUED FROM PAGE 23)

al. Sometimes a nonuser may think we're a bunch of nerds. I remember the first time I went to an ACCESS meeting, even though I had a computer and knew quite a bit about it, I was still wary of walking into that room because I thought there were all these strange people in there. But eventually, I found out it was just a group of people who shared a common interest. We're no different than the car clubs or the Boy Scouts or anything like that," Guelker insisted.

In holding the computer symposium for the children, Guelker said that, along with his original objective, which was to assist the kids, his next objective was to help the club attract interested individuals, no matter what their age.

"The older folks tend to be steadfast in their ways, and in some cases they really don't understand computers. But the kids are not intimidated by them. I think they are more mystified by them rather than stupefied. At the symposium, I remember watching a little girl working with the speech synthesizer program. From the start I could tell she may have been a little introverted. However, the more she typed, the more she seemed to be saying, and the more she said, the more she brought herself out in what she said. It wasn't so much the words, but the content of those words. I think she acted that way because she could identify with the computer. She was right there, face to face with it, and it wasn't judging her. I also remember Kendrick. From the time he came in early in the morning, he was really having a ball. He played one game for about five minutes and all of a sudden, he was the expert. Then he was showing everyone else how to do it. I could tell he wasn't shy at all."

Guelker said the enthusiasm in the room inspired his members also. "The members thought the whole thing was pretty neat. They reserved that special time to devote to the kids. You know they had other things they could have done that day that may have seemed more fun, but after seeing the kids' eyes light up, they really started to have fun too."

Guelker said that with the success of the computer symposium, he already has plans for other activities designed to help his club and other people in the community.

His next project involves a similar symposium to be held at a local mall, which will be aimed more toward adults. He said getting the children interested is the easy part. Next he will try to teach the older folks about the psychological and academic benefits of computers.

"A computer, in the mind of today's adult, is a lot like 'new math' in the mind of yesterday's adult," Guelker said. "I think the kids understand how important computers are in today's society; it's their parents who need to be enlightened. Which kid do you think has the advantage in today's schoolroom? The kid with the piece of paper and the pencil, or the kid with the calculator?"

I asked Guelker what other types of projects would he undertake if he had unlimited time and financial resources?

"I would think that with all the machines that people out there have that are just sitting around doing nothing," Guelker said, "there would be so much that could be done if people would only take the time."

Guelker challenged all the Atari users' groups with broken-down computers collecting dust in their basements to think about how they could benefit someone else.

"You have all these machines that are just sitting around the house. Why don't you get your groups together and fix them for donation to a worthwhile organization in your communities? If it's money your concerned about, you can declare it as an income-tax deduction."

He added that another good idea for a group interested in doing something for their community would to be put an ad in the paper asking for inoperable computers. The group could pick them up, fix them and donate them to needy children in the group's area.

"Most of the time the repairs can be made with simple component replacements," Guelker said. "And the knowledge and experience of most computer groups, not to mention the financial capabilities, would make the procedure rather painless."

As Atari computer enthusiasts, we all have what Tom Guelker called a common interest. I think we also have another similarity: We all know the value of time. The time these children had to work with Atari computers—time generously donated by the members of ACCESS—will never be forgotten.

Tom Arterburn is 25 and lives in Belleville, Illinois. He dedicates this article to his parents, Tom Sr. and Rita Arterburn.

Massaging Your MegaFile

Upgrading Your MegaFile 20

by Gregg Anderson

BASIC TOOLS AND LAYOUT.

Remember the good old days? When having 64K of RAM and a 127K disk drive was heaven? Now we have four-megabyte Ataris with 20-megabyte hard disks, and we're not satisfied. There's not much we can do about the four-megabyte limit on the ST, but there is a cure for the MegaFile 20—a quick, painless and relatively inexpensive cure. What is it? Read on.

Twenty megabytes of storage sounds like a lot when you're first starting out, but after a while it quickly becomes obvious that even that much storage isn't going to cut it for serious graphics, drafting or desktop-publishing work. Supra, ICD and even Atari have recognized this and expanded their product lines beyond 20 megabytes. Atari joined the bandwagon by ending production of their MegaFile 20 and adding the MegaFile 30 as their bottom-line unit. As their top-end systems they've introduced the new MegaFile 60 and 44.

The original MegaFile 20 (at least, my unit) utilizes a 20-megabyte Seagate ST-225 disk mechanism with a modified Adaptec 4000 controller (MFM) and host adapter (interface) on a single PC board. This unit operates at an advertised 65Msec access rate and a 625K data-transfer rate. Using ICD's RATEHD.PRG, I found a "real world" rating of 86Msec and a 305K transfer rate (the difference between the raw data-transfer rate and the real-world rate is not unusual). Though far from being the fastest kid on the block, the unit is quite serviceable and a major improvement over the original Atari SH204 hard drive, with fewer bugs in its controller.

This is all well and good, but where does that leave the thousands of us who purchased MegaFile 20s? Are we stuck with a (relatively) slow 20-megabyte drive? Is our only solution the purchase of a brand-new (and expensive) hard drive? Fear not. There is an alternative.

First, I have to make a few statements to keep the lawyers out of my hair. Any modification you make to your hard drive will void whatever warranty existed for that unit; just the act of opening the case is enough. Also, while I had no problems with my conversion, I can't promise that yours will go as smoothly. In other words, I can't take responsibility for problems or mistakes if you try this procedure. I've every confidence that this is a sound and easily done upgrade, but the final responsibility is yours and yours alone.

Before we start, let's list the ground rules for our project.

1) Since few of us are electronic engineers, the upgrade has to be a fairly simple one involving no complex rewiring or soldering.

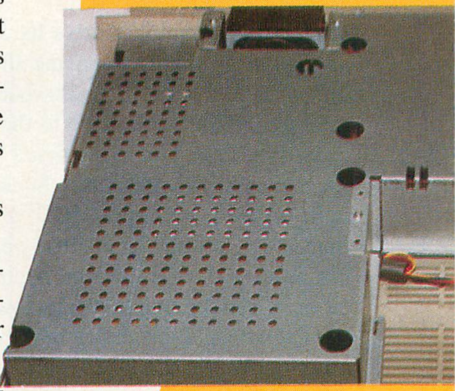
2) It has to be affordable. If it's too expensive, it would be easier to simply replace the unit with something newer and sell the MegaFile.

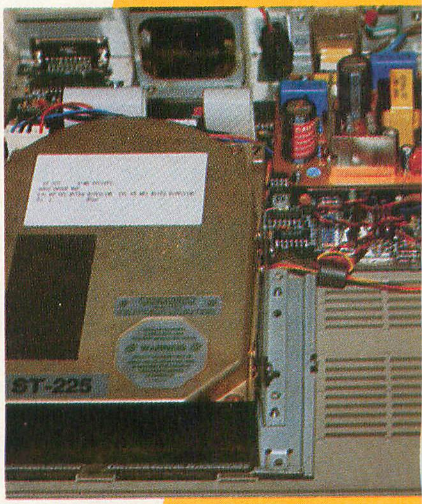
3) We want something that will fit in with our system. In other words, something that looks like it belongs there rather than like a refugee from the IBM research labs or something stolen from the Queen Mary.

4) We don't want to give up performance while increasing our storage. If possible, we want to increase our system's data-transfer speed.

I've done some research and made a few interesting discoveries about the MegaFile 20 and hard drives in general. Did you know the MFM controller in the MegaFile can accept any MFM-type drive? It doesn't seem to matter what size it is; the Adaptec 4000 will handle it. How about mounts for the hard-disk mechanism itself? No problem! Assuming it abides by standard Seagate construction, any half-height drive will fit in the MegaFile's case with no modifications whatsoever. Does this give you any ideas? It gave me a few.

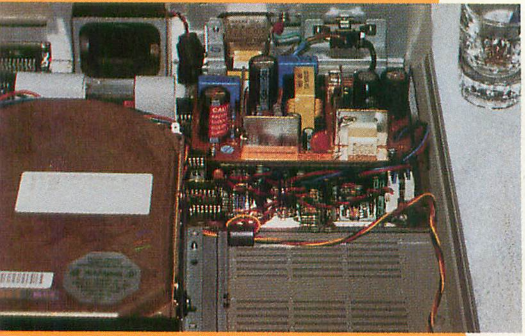
You guessed it. I started looking into up-


COVER REMOVED.



RF SHIELD REMOVED, SHOWING ORIGINAL MECHANISM, POWER SUPPLY, AND CONTROLLER CARDS.

EVERYTHING APART.



NEW MECHANISM INSTALLED.

grading the *MegaFile*'s drive hardware as an option to simply replacing the entire unit. I could find only one drawback to upgrading the MegaFile. Due to the limited size of the case, you're restricted to a single internal-drive mechanism. So unlike the units offered by Supra, ICD and Tech Specialties, it won't be internally expandable. Even so, I considered this a minor limitation, given the simplicity and low cost of the upgrade and the ability to daisy chain other drives via the DMA port.

What about the controller? Everybody's going to RLL for their new systems. Doesn't that mean RLL is better? Well, yes, there's some truth to that. RLL is becoming popular as it allows (on average) slightly higher transfer speeds and up to 50% greater storage capacity than an MFM drive of the same physical size. But there is a small problem with going RLL: cost. On average, an RLL controller can cost you up to $70 more than an MFM unit. RLL-drive mechanisms are also more expensive than MFM units, though by a smaller margin. Besides, we've got a perfectly good MFM controller in the *MegaFile* already. Changing over to an RLL unit could add as much as $250 to the cost of our upgrade since we'd also have to replace the host adapter. Then there's the board's mounting to consider. While an ICD controller and host adapter might fit in a MegaFile case, it would require modifying the mounts more than a little. By the way, it seems that Atari agrees that RLL is the way to go as well; their new MegaFile 30, 60 and 44 units are all RLL.

So while an RLL-based system is superior to an MFM system and is preferable in an all-new drive, I don't feel its advantages outweigh the additional costs when upgrading a MegaFile 20.

By the way, there has been some talk of using an RLL controller with an MFM drive to gain the advantage of greater storage without paying for the RLL drives. I have a few doubts about the long-term reliability of this combination. I'm not saying it can't be done; there are reports on the BBSs indicating that it has been done with some success. It seems that most RLL-certified drives are little more than MFM units that have passed a more stringent QC check and are thus cleared for the higher-density formatting. This appears to be the same way manufacturers determine single- and double-sided floppies. If it passes the double-sided Quality Control test, it's sold as a double-sided disk; if not, it's sold as single-sided.

Since the units in question have not passed the factory test, there is the chance of bad sectors appearing and the loss of critical data if the drive is formatted as RLL. So if you've already got an RLL controller, and you're willing to take the chance of losing data now and then, it might be worth trying. But if any bad sectors appear after your format, or if there is any question as to the quality of the drive, I strongly suggest you let it remain MFM and try a different mechanism.

Okay, so I'm keeping the controller. Great, what about the drive itself? Well, I'm not going to tell you which drive you should use. That will depend entirely on your needs and budget. You'll have to stay with an MFM drive though, unless you want to spend the money upgrading to an RLL controller and new host interface.

I strongly suggest you stay with a unit that draws less than 25 watts. The MegaFile's power supply is limited to 50 watts, and any attempt to push your drive close to that is sure to end in failure. I recommend staying as close to the power needs of the original mechanism as possible—under 15 watts. Be sure to stay with half-height drives too; full-height drives won't even come close to fitting in the MegaFile case.

You'll want a drive without the built-in SCSI interface. After all, why spend money on something you've already got? Look for the "Auto-Park" feature in anything you buy. Auto-Park is cheap insurance against power failures as it returns your heads to the park position whenever power is removed and eliminates the need for a separate park utility. Make sure whatever you order is designed with the same mount holes/brackets as a Seagate ST-225; if you tell them you want something to fit in a MegaFile 20, they're just going to look at you funny. Seagates they'll know about, but MegaFiles?

In my case, I chose the Seagate ST-251 drive (purchased from ABCO Computer Consultants). It offers 42 megabytes of storage, auto-parking, an advertised access speed of 40 ms and about the same power needs as the MegaFile's original drive. I needed to limit my spending to less than $400, and the ST-251 fit that requirement perfectly. While 40 megabytes seems to be the most common large MFM drive, there are mechanisms available from a variety of manufacturers in somewhat larger sizes. Unless you're looking for something really big, you can usually find MFM drives that offer storage and speeds that match their RLL counterparts in the 5¼-inch disk size.

So why did I choose a Seagate unit when I could have gotten a Miniscribe or Microscience of larger capacity for less? In a word: *reputation*. Seagate is famous for building low noise and long-term reliability into their units and has developed a sizable following based on that. But, as I said earlier, the decision is up to you.

So let's get on with it.

Before you do anything else, *make a complete backup of your present hard disk's contents*! You'll need this to rebuild your new disk. Constant backups are always a good idea, but for this they're vital. Use a file-copy utility, such as *Turtle*; do not use an image-copy utility. Image utilities are faster, but they will duplicate your current disk exactly as it now exists to your new drive. In other words, they'll make your new 40 Meg unit think it's a 20, and you'll lose that extra storage you wanted.

1) Find a quiet, well-lighted area with lots of clear table space. Ground yourself to avoid damaging your equipment with stray static charges. Prepare a No. 1 and a No. 2 Phillips screwdriver, a pair of needle-nosed pliers and a soft towel or work cloth (lint-free).

2) Park the heads of your MegaFile 20

before starting anything else; if you're able to sell the present mechanism, you can cut your out-of-pocket expenses. (I sold mine to a friend with a Tech Specialties unit.) Use the SHIP.PRG program that came with the MegaFile 20 for this.

3) Carefully disconnect the MegaFile 20 and place it on your work cloth upside down.

4) Remove the nine gold-colored screws from the bottom of the case. Don't touch the four silver screws yet. (Note: You've now voided your warranty. Don't you feel wicked?) Keep the screws in a separate container.

5) Holding the case together with both hands, turn the unit over and place it right side up on the work cloth.

6) Gently lift the case off the base and place it to one side.

7) With the needlenosed pliers, very gently straighten the eight tabs holding the RF shield to the motherboard. Don't damage these if at all possible.

8) Carefully lift the shielding off the base. You should now be able to see the host adapter/controller board, power supply and the disk mechanism.

9) Holding the drive mechanism securely in its mount with one hand, tilt the base so you can see the bottom and the four remaining silver screws holding the drive to the case.

10) With the screwdriver, remove the four silver screws and gently return the unit to an upright position. Make sure the drive doesn't slip out of place as you do this.

11) Without touching the board or any of its contacts or components, carefully lift the drive an inch or so from its mount. *Do not touch the bottom of the drive*! This is where the drive's circuit board is located, and it's not insulated against static.

12) Gently pull the three connectors from the back of the drive. There are two data cables and a power cable here. Don't worry about getting them confused, as they are quite different from each other.

13) Inspect the connectors on the back of the drive and make sure they're identical to the connectors on your replacement drive. If not, put everything back together and reorder a different drive. This shouldn't be a problem if you're using a Seagate or Seagate-compatible drive though. If they are identical, continue.

14) Gently lift the drive mechanism from the case and place it on your work cloth. Try not to bump or shake the unit.

15) Using the Phillips screwdriver, carefully remove the two screws holding the mounting brackets to each side of your old drive. Once these are off, you can place the original drive aside (use the padded box your new drive came in).

16) Now install the mount brackets on your new drive. Be sure to keep them aligned to the same side/direction as they were on the original drive. Don't overtorque them. Again, be careful not to touch the circuit board on the bottom of your drive.

17) By the way, don't mess with the resistor-termination pack (on the drive's motherboard). For a single-drive system, just leave it as it sits. Also, make sure the drive-select jumper is identical to the one you're replacing (it should come that way). These are used mainly to configure IBM-type systems. Your new drive will come with directions on these things if you decide to change your present configuration.

18) Holding the drive just above its mount, gently connect the three jumper cables. You may find the power connector a little tight. If so, be sure not to apply too much pressure without supporting it from below.

19) Carefully place your new drive in the case, aligning the mount holes to match those of the original mechanism. The unit should fit perfectly, with less than a ¼-inch gap between the drive's connectors and the connectors on the motherboard.

20) Again holding the drive in place with one hand, tilt the base so you can see the four screw holes that hold the drive's mount bracket in place.

21) Replace the four silver mounting screws.

22) Now we have a choice. If your drive came preformatted and partitioned, you can hook it up to your computer and test it out, or you can continue to reassemble the case. Let's go ahead and assemble the beast.

23) Gently replace the RF shielding onto the base. Make sure you've not trapped any wires or cables under the shielding's edge and that all the tabs are in their slots. With the needlenosed pliers, push the tabs back to their original positions.

24) Place the case on the base (poetic, isn't it?) and, again with both hands, turn it upside down on the work cloth.

25) Install the nine gold screws that hold the unit together, again being careful not to overtorque them.

26) Return the unit to an upright position, and you're finished. All you have to do is hook it up to your ST and start formatting.

That's all there is to the hardware end of the upgrade, but I'd be remiss in not touching on the software end of the process as well. Just like a floppy, hard disks must be formatted before use, and since the current TOS has serious problems with handling hard disks larger than 16 megabytes, you'll have to divide (or partition) your new disk into smaller "logical" drives. (The new TOS 1.4 seems to have fixed this limitation and now allows partitions of up to 32 megabytes.) Be sure to keep your FOLDERXXX.PRG handy; with more storage available, you're more likely to run into the dreaded 40-folder limit. For those of you with TOS 1.4, you'll want to use the new CASHE90.PRG that replaces FOLDERXXX.PRG for the new TOS and speeds up your disk-accessing as well.

So what are our choices for formatting software these days? Unless you're willing to try one of the older packages downloaded from a BBS, there are only three.

Atari's formatting/partitioning software is easy to use and free (you got it when you bought your MegaFile), but it was designed with Atari drives in mind. Because of that it can recognize only 20-, 30- and 60-megabyte mechanisms. If you have one of these sizes, you're off and running, but if you have a 40-megabyte (or other "odd" size), you're out of luck.

ICD has an excellent utility for formatting and partitioning a hard drive, but their most recent releases contain a rather sneaky routine that looks for ICD's own host adapter/controller boards. If it doesn't find them, it won't do anything. Unless you've replaced your host adapter and controller with ICD units, you won't be able to use their utility for this upgrade.

Supra's software is the utility of choice here. Designed with flexibility in mind, it will operate with almost any combination of host adapter/controller and drive mechanism. Best of all, at $39.95, it's available separately from Supra and is easy to operate. As you may have guessed by now, this was the utility I chose for my upgrade.

Start your system with a blank disk in Drive A. You need to do this because any hard-disk auto-boot files on Drive A will lock everything up with an unformatted hard drive attached. Once everything is up and running, insert your Supra hard-disk utilities disk and click on the SUPFMT.PRG file.

From the SUPFMT.PRG display, select "Hard Disk Type" or "Controller Type" from the primary menu. You'll be given a display showing a variety of hard disks and controllers. Scroll down this display until you find the exact name of the hard disk you've installed, then select it.

When you do this, the "Specifications" display will change to list the details of your

(CONTINUED ON PAGE 129)

BOOT CAMP

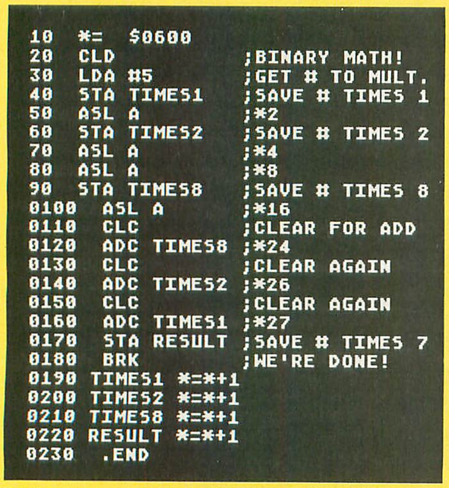
by Tom Hudson

Welcome back! As I mentioned last issue, there are only a few more 6502 instructions left for us to cover, and we'll talk about them in the next two installments. There are also a couple of instructions we're going to skip until later. They are for more advanced uses and won't make much sense until you've got more experience with assembly language.

Several people have written lately, asking if we'll get into full-scale programs using the Atari's powerful operating system. The answer: You bet! We're going to find out how to access the disk, cassette, graphics, keyboard and just about anything else you'd like to hear about. We'll study routines for high-speed math, player/missile graphics and more. *Boot Camp* is here not only to teach you what 6502 assembly instructions do, but how to apply them.

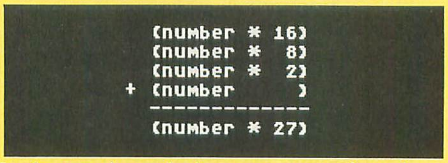
Two Solutions

Last issue, I asked you to write a program that multiplied the number 5 by 27. There are an almost infinite number of ways to do this, and I'll show you two of them now. Remember, these aren't the only possibilities, and even though your solution may not be as efficient, getting the correct answer is what counts most.



```
10   *=  $0600
20   CLD                  ;BINARY MATH!
30   LDA #5               ;GET # TO MULT.
40   STA TIMES1           ;SAVE # TIMES 1
50   ASL A                ;*2
60   STA TIMES2           ;SAVE # TIMES 2
70   ASL A                ;*4
80   ASL A                ;*8
90   STA TIMES8           ;SAVE # TIMES 8
0100   ASL A              ;*16
0110   CLC                ;CLEAR FOR ADD
0120   ADC TIMES8         ;*24
0130   CLC                ;CLEAR AGAIN
0140   ADC TIMES2         ;*26
0150   CLC                ;CLEAR AGAIN
0160   ADC TIMES1         ;*27
0170   STA RESULT         ;SAVE # TIMES 7
0180   BRK                ;WE'RE DONE!
0190 TIMES1 *=*+1
0200 TIMES2 *=*+1
0210 TIMES8 *=*+1
0220 RESULT *=*+1
0230   .END
```

The first solution I'm going to cover is shown above. This program uses the principle of breaking a multiply into bite-sized pieces, as shown last issue. In this case, I broke the multiply by 27 down into the following group of adds:



```
  (number * 16)
  (number *  8)
  (number *  2)
+ (number     )
  -------------
  (number * 27)
```

Let's step through the program and see how it works.

Line 20—clears the decimal mode. Always remember to be sure of the setting of the decimal flag before doing any arithmetic.

Line 30—loads the accumulator with the number 5. When the routine is finished, this number will be multiplied by 27 and stored in the memory location labeled RESULT.

Line 40—stores the accumulator's contents in the memory location labeled TIMES1 (5*1). We need to save this value for later, when we add the bite-sized pieces together.

Line 50—shifts the accumulator contents left one bit, multiplying it by two.

Line 60—saves the accumulator (now 5*2) in the location TIMES2. This value is also needed for our final result.

Line 70—shifts the accumulator left one bit again, leaving the accumulator with the value 5*4.

Line 80—performs another left shift on the accumulator. The accumulator now contains 5*8.

Line 90—saves the accumulator's contents in the location TIMES8.

Line 100—performs a final left shift on the accumulator, leaving the accumulator with the value 5*16. At this point, we have all the bite-sized pieces we need to get our answer and are ready to add them up.

Line 110—clears the carry flag for the first add in the group. Remember, this is a necessary instruction before any single-byte addition.

Line 120—adds the accumulator (5*16) to TIMES8 (5*8), leaving the result (5*24) in the accumulator for the next add.

Line 130—clears the carry for the next add.

Line 140—adds the accumulator (5*24) to TIMES2 (5*27), with the result (5*26) left in the accumulator.

Line 150—clears the carry again, for the final addition operation.

Line 160—adds the accumulator (5*26) to TIMES1 (5*1), leaving the accumulator holding the final value, 5 times 27!

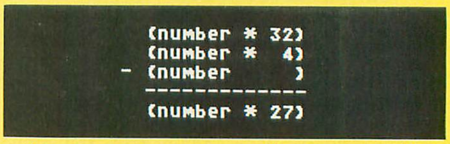
Line 170—saves the final answer in the location labeled RESULT.

Line 180—BREAKs the execution of the program. At this point, you can check the location RESULT to be sure it contains 5*27, or 135 ($87 hex).

Lines 190-220—reserve one byte for each of the four data areas used by the program.

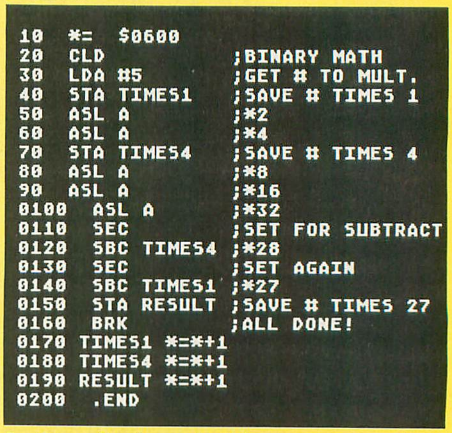
Solution -2.

The second solution is a modification of the first technique. In this program, I break the multiply down into smaller pieces again, but structure it so that subtracts are used instead of adds:



```
  (number * 32)
  (number *  4)
- (number     )
  -------------
  (number * 27)
```

As you can see, we get the same result as with adds, but with only three math operations instead of four. The figure below shows the 6502 code necessary to implement this method.



```
10   *=  $0600
20   CLD                  ;BINARY MATH
30   LDA #5               ;GET # TO MULT.
40   STA TIMES1           ;SAVE # TIMES 1
50   ASL A                ;*2
60   ASL A                ;*4
70   STA TIMES4           ;SAVE # TIMES 4
80   ASL A                ;*8
90   ASL A                ;*16
0100   ASL A              ;*32
0110   SEC                ;SET FOR SUBTRACT
0120   SBC TIMES4         ;*28
0130   SEC                ;SET AGAIN
0140   SBC TIMES1         ;*27
0150   STA RESULT         ;SAVE # TIMES 27
0160   BRK                ;ALL DONE!
0170 TIMES1 *=*+1
0180 TIMES4 *=*+1
0190 RESULT *=*+1
0200   .END
```

Let's walk through this program and see what's going on.

Line 20—clears the decimal mode for binary arithmetic. I can't overemphasize the importance of knowing the status of the decimal mode flag. If you're in doubt, set or clear it as needed.

Line 30—loads the accumulator with the number 5. When this program is finished, the number 5 will be multiplied by 27.

Line 40—saves the contents of the accumulator in the location labeled TIMES1, for later use.

Line 50—shifts the accumulator left one bit, multiplying it by 2.

Line 60—shifts the accumulator left again, leaving the accumulator with the value 5*4.

Line 70—saves the contents of the accumulator (5*4) in the memory location TIMES4.

Line 80—shifts the accumulator left again, leaving the value 5*8 in the accumulator.

Line 90—performs another left shift. At this point the accumulator contains 5*16.

Line 100—shifts the accumulator left a final time. The accumulator now contains the value 5*32. We are now ready to perform the subtract operations as shown above.

Line 110—sets the carry flag for the first subtract operation. Remember, the carry flag should always be set before a single-byte subtract to ensure correct results.

Line 120—subtracts the value TIMES4 (5*4) from the accumulator (5*32), leaving the accumulator containing the value 5*28.

Line 130—sets the carry flag for the next subtract.

Line 140—subtracts the value TIMES1 (5*1) from the accumulator (5*28), leaving the accumulator with the value 5*27!

Line 150—saves the answer in the location labeled RESULT.

Line 160—stops the program's execution with the BRK instruction. At this point, you can verify that the location RESULT (and the accumulator) contains 5*27, or 135 ($87 hex).

Lines 170-190—reserve one byte for each of the three data fields used by the program.

Obviously, these are just two of the thousands of solutions possible for this problem.

Stacking the Deck

The last topic we're going to cover before going on to bigger and better things is the 6502 *stack*. This is an important feature of the 6502, as it allows us to write subroutines. Since the stack concept is important, we're going to cover it in detail starting with this issue and finish it with assembly examples next time. Let's get started finding out what the stack is and how it works.

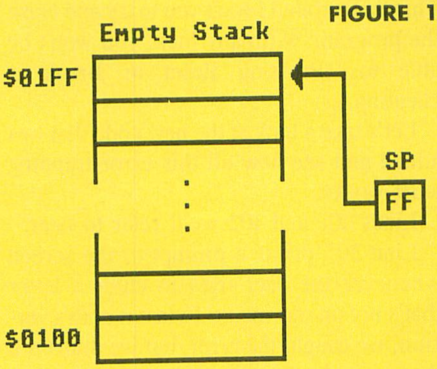
The 6502 reserves 256 bytes of memory from $0100-01FF (also called page 1) for a temporary storage area. We call this area the *stack*. This area is automatically maintained for the 6502, but we can use it for short-term storage, too.

We call the stack a "last-in, first-out" structure. The last number placed on the stack is always the first to be pulled off. A good way to remember this is to think of a stack of pancakes. When you pile them up, the last one put on the stack is on top. When you take them off one at a time, the last one you put on comes off first. Using this analogy, the computer could keep track of 256 pancakes, each with a number written on it.

The computer keeps track of the stack's contents by using the Stack Pointer register inside the 6502. This pointer ranges from $00-FF. When the stack pointer contains $00, it is pointing to the memory location $0100. When it contains $FF, the location $01FF is indicated.

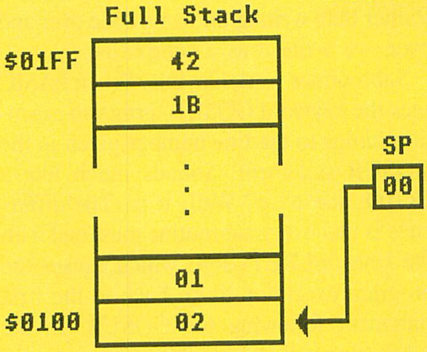
Interestingly, the stack works backwards from the way we would expect. When the stack is empty, the stack pointer is set to $FF. **Figure 1** shows an empty stack.

FIGURE 1



As the stack is filled with more and more values, the stack pointer is decremented, pointing to lower areas of page 1. When completely filled, the stack pointer will contain $00, as shown in **Figure 2**.

FIGURE 2



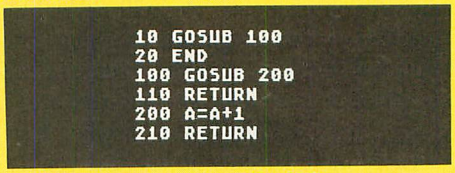
Since the computer has only reserved 256 bytes for a stack, there are obviously limitations in its use. If the stack is filled with too many values, the stack pointer will wrap around back to $FF and begin wiping out earlier stack entries! There is no error message for this, so you must be careful when working with the stack.

When entries are removed from the stack, the process is reversed. As each byte is pulled off the stack, the pointer is *incremented*, pointing to progressively higher locations of the stack.

How Subroutines Work

In BASIC, subroutines are easy to write. You simply set up the necessary BASIC code, put a RETURN instruction at the end of it, and call it with the GOSUB statement whenever you need it. The subroutine code is performed, and BASIC resumes execution at the next statement after the GOSUB. Neat, huh?

In order for a BASIC subroutine to work, the computer has to know how to get back to the instruction after the GOSUB. It does this by using a stack. Let's look at a simplified example of how a BASIC subroutine is executed.

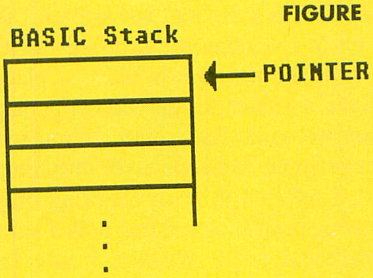


```
10 GOSUB 100
20 END
100 GOSUB 200
110 RETURN
200 A=A+1
210 RETURN
```

The above is a short BASIC program using the BASIC subroutine statements, GOSUB and RETURN. We're going to step through it and watch what happens to the BASIC stack, a special area similar to the 6502 stack.

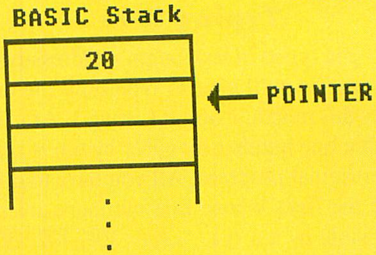
Before execution, the stack is empty, and the stack pointer is pointing to the first available position.

FIGURE 3



Line 10—GOSUB to Line 100 is executed. First, the computer finds the next statement after GOSUB. The next statement is in Line 20, so the computer *pushes* that line number onto the first location on the stack, and changes the stack pointer to point to the next available location. Execution then proceeds at Line 100. At this point, the stack looks like:

FIGURE 4



(CONTINUED ON PAGE 131)

BASIC TRAINING:
CRASH-PROOF PROGRAMMING

by Clayton Walnum

Compared to making your programs ready for the user, the actual writing of them is as easy as extinguishing a match with a water hose. Why? Because no matter how hard you try, no matter how many hours you spend trying every conceivable combination of inputs, you will never be sure that you've caught all the bugs or trapped all the possible errors. I sometimes say there's no such thing as a bug-free program. That's a little severe, maybe. It's probably more accurate to say that you'll never be able to *prove* that your program is bug-free. It might run fine for years, but one day. . . .

So let's take it as a given that we'll never know if our programs are bug-free. That doesn't mean, however, that we don't try to *make* them bug-free. Once we've got our program running, we must go through it and make sure it can "never" bomb out on the user. This is a big job. Sometimes it takes as much programming to do this as it did to write the program in the first place.

Take, for example, the address-book program we wrote last month. If you recall, that program had little error checking. In fact, I referred to error checking only in passing, promising you that in this month's installment (well, actually, I said "next month," but that was last month, so this month is next month. You see?) we would take that bare-bones program and make it solid enough for anyone to use. This month's Listing 1 is the result.

Notice how much bigger the program is? All that extra code and the program does nothing more than it did last month—except it catches every error I could imagine a user making.

Typing It In

If you typed in last month's address-book program, you'll have to type only the new lines this month. Look at Listing 1. Any line with a line number divisible by ten is from the original program. All you have to do is add the lines whose line numbers are not divisible by ten (i.e., Lines 101, 102, 305, 371, 372, etc.).

If you didn't type last month's program, you get to type all of Listing 1. Lucky you.

Bomb-Proofing

Once you have the program completely typed, save it to disk, then run it. The instructions for using the program are in last month's issue, so I won't repeat them here. This time around I want you to try and crash the program. With any luck, you won't be able to. (Pressing Reset or Break is cheating.)

Let's take a look at the new code that was added and see how all this error-trapping stuff works.

Lines 101 and 102 we'll refer to later.

Line 305 prints a prompt to the screen whenever the user tries to enter a letter that's not on our menu. In last month's version, we caught the error, but we didn't tell the user what he was doing wrong.

Line 371 calls a subroutine (GOSUB CHECKFILENAME) that checks whether the user entered a legal filename.

What's a subroutine? Many times in our programs, we have to do the same type of operations in different parts of our program. Rather than writing the same code over and over, we write it as a subroutine.

Subroutines work like this: When BASIC sees the keyword GOSUB, program execution jumps to the line number given in the GOSUB statement. Sounds much like a GOTO, doesn't it? Well, it is. The difference is that every subroutine must end with the keyword RETURN, which causes the program execution to go back to the first statement following the GOSUB. In the subroutine call in Line 371, the number of the line to which we want to jump is contained in the variable CHECKFILENAME. We could have written GOSUB 801, but the other way is more descriptive. If you look at Line 102, you'll see where CHECKFILENAME gets its value.

So, in Line 371, we first jump to Line 801 and execute the program statements there, continuing until we find a RETURN, after which we go back to Line 371 and continue with the IF. . .THEN statement on that line. There are three places in the program where the user enters a filename, and we can use the same subroutine to check them all.

In the subroutine CHECKFILENAME, we set FILENAME$ to an empty string if what the user entered was not acceptable. That's the reason for the IF. . .THEN statement following the GOSUB. If, after returning from the subroutine, we find that the string is empty, we know to go back and have the user enter a different filename. We'll look in detail at the subroutine itself later.

Because this section of code creates a new file, once the user enters a valid filename, we must check whether that file already exists. Our user won't be very happy if we erase an important file simply because he made a mistake when typing in the filename (or maybe he meant to select the "Load" option rather than "Create").

Line 372 first sets a TRAP. Should we get an error, program execution will jump to Line 379. After setting the TRAP, we try to open for read operations the file the user requested. This is how we can check whether the file already exists, without damaging it if it does. If the file isn't on the disk, we'll get a "file not found" error, and the TRAP statement will send us to Line 379 where we go ahead and open the file for write operations.

Notice that, in this case, we want the error. If we don't get an error, it's because the file the user wants to create already exists (otherwise, we wouldn't have been able to open it). If the file does exist, the TRAP is not activated and we drop down to Line 373.

Line 373 tells the user that the file he's chosen to create already exists and asks if he wants to erase it.

Lines 374-376 retrieve the user's answer and respond accordingly. If the user answers with a "Y" or a "y" (remember: people think of upper- and lowercase letters as equivalent), we go ahead and erase the file with the user's blessing. (If the user made a mistake this time—that is, really wanted to say, "N," that's just too bad. As programmers, we have the right to expect our users to have at least rudimentary intelligence. However, if the results of the "Y" answer could be truly disastrous, it's not a bad idea to give the user one more chance and ask, "Are you sure?")

Line 379 closes the file we opened in Line 372. We can't reuse an open channel; we have to close it first, even if our first OPEN came back with an error.

Line 451 does the same filename checking for the Load function as we did for the Create function. It works exactly the same.

Line 452 sets a TRAP for Line 468. In the Load function, we don't have to worry about erasing the user's file because we're going to open it for append. But we can't open a file that isn't on the disk. So, if when we try to open the file in Line 460, we get an error, the TRAP we've set will send us to Line 468.

Line 468 closes the file, then checks whether the error returned was a "file not found" error. If it was, we tell the user the file doesn't exist and send him back to try again. But wait a minute! What's this PEEK(195) stuff?

Your computer's memory is like a long string of little boxes, each of which will hold one byte of information. These little boxes, just like a row of houses, have "addresses" that we can use to reference them. The first byte in memory is numbered 0, the second byte is 1, the third byte is 2 and so on, all the way through to the last byte. (There are over 65,000 addresses!)

There are many times when we need to look at one of these locations in memory and retrieve what's there. To do that, we need to know its address and then use that address with the PEEK statement.

Many of the locations in your Atari's memory are reserved for special purposes, purposes that never change. For example, location 195 holds the number of the last error that occurred. Hey! What a coincidence! That's the address we just used in our PEEK statement.

The statement PEEK(195) causes BASIC to look into location 195 and tell us what value it finds there. That value will be the number of the last error that occurred. The "file not found" error is number 170. So if, after our TRAP sends us to Line 468, we find that PEEK(195)=170, we'll know that the file we just tried to open didn't exist. Neat, huh? Notice that rather than using the number 170 in our PEEK statement, we've used a variable called FILENOTFOUND, which was initialized in Line 102. By using names like this, rather than cryptic numbers, we'll more easily remember what our code is doing.

If the error we get is not 170, we drop down to Line 469. A "file not found" error is only one of many we could get. It's the most obvious one, so we handle it specifically. Any other error will not be identified by our program. We'll just tell the user that an error occurred and make him try again.

Line 469 tells the user that an error has occurred (an error other than "file not found," which we've already checked for) and goes back to get another filename from him.

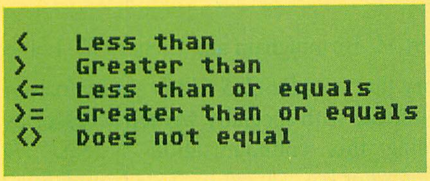
Line 681 checks the filename for the View function.

Lines 683, 772 and 775 check to make sure the file the user wants to view exists. If it doesn't, we warn him and make him try again.

Line 801 is the beginning of our subroutine to check the filenames input by the user. In this line, we check that the filename begins with an uppercase letter. Anything else would be illegal. We need the TRAP 809 in case the user just pressed Return without entering a filename.

Notice how we can compare the letters "A" and "Z" as if they were numerical values, checking whether the value in the string is less than or greater than some other value. How do we assign a value to a character? By the order in which they are arranged. Therefore, "A" is less than "Z", and "Z" is greater than "A."

In BASIC we have several operators we can use to compare values:



```
<   Less than
>   Greater than
<=  Less than or equals
>=  Greater than or equals
<>  Does not equal
```

Line 802 checks whether the filename includes the device (D:, D2:, C:, etc.). If the device is there, there has to be a colon in the second or third character of the filename (D: or Dn:, where n is the drive number). If we find the colon in its proper place, we jump to Line 805. The TRAP 808 is there in case the filename is less than three characters long. For example, if the user typed FI as the filename, we will get an error when we try to look at the value of FILENAME$(3,3); there is no character in that position. In this case, we just assume that the device is not present in the filename.

Line 804 adds the device "D:" to the user's filename if it didn't already contain its own device. If this section of the program doesn't make sense to you, go back and review the *BASIC Training* on string handling (August '89).

Line 805 finds the location of the colon in the filename. Because Atari filenames (and this time I'm referring to the portion of the filename after the device) must begin with a letter from "A" to "Z," we need to know where the first letter is. It is, of course, the first character after the colon.

Line 806 checks whether the first letter of the actual filename is a letter from "A" to "Z." If it is, the filename is okay, and we return from the subroutine. If it's not, we warn the user and set FILENAME$ to an empty string, signalling that the subroutine did not end up with a valid filename.

LET'S TAKE A LOOK AT THE NEW CODE THAT WAS ADDED AND SEE HOW ALL THIS ERROR-TRAPPING STUFF WORKS.

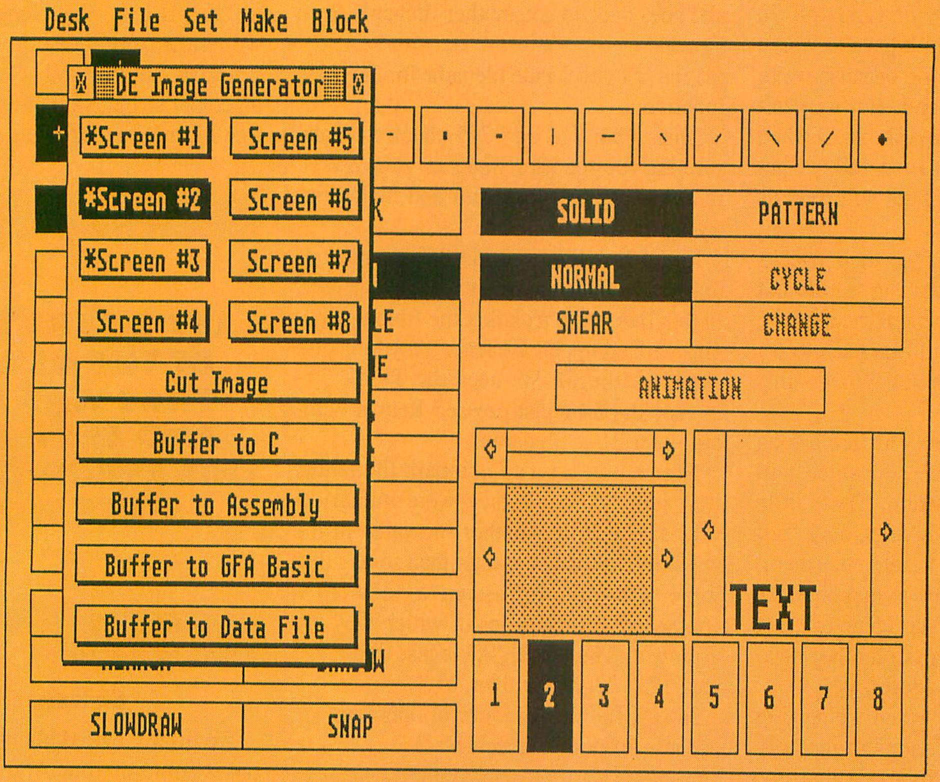
Conclusion

We briefly talked about the special purposes some of your Atari's memory locations have. If you're interested in finding out more about your computer's memory, you should refer to "The Master Memory Map," which was published in many parts in ANALOG Computing last year, or any good 8-bit Atari memory map. However, we will be covering many of these locations in future *BASIC Training* columns. Bet you can't wait.

Clayton Walnum is the Executive Editor of ANALOG Computing, *as well as the Associate Editor of* VIDEOGAMES & COMPUTER ENTERTAINMENT.

(CONTINUED ON PAGE 127)

THE DEGAS ELITE IMAGE GENERATOR



by Robert M. Birmingham and Richard C. Leinecker

If you're a programmer who has experimented with computer animation, you know that one of the least enjoyable parts of the process is defining the bit-mapped images you wish to animate. This is usually done by drawing the image on a piece of graph paper, then translating the image into numbers that the computer can understand.

Besides being time-consuming, one mistake may mean that you have to start over from scratch. If this sounds familiar, we can sympathize, since we've done the same thing more times than we'd like to admit! Faced with doing this "one more time" and with our patience worn thin, we figured there had to be a better way. So we locked ourselves away and set out to find the solution. Several days later we emerged with *The DEGAS Elite Image Generator.*

The DEGAS Elite Image Generator, or *The Generator* for short, is a GEM desk accessory that uses *DEGAS Elite*'s ability to communicate with specially written accessories. Using this communication system, *The Generator* lets you cut sections from *DEGAS Elite* pictures, in any resolution, and save them to a disk file or send them to your printer.

The program can output your images as source code for C, assembly and GFA BASIC, as well as output your images as a special data file.

Getting Started

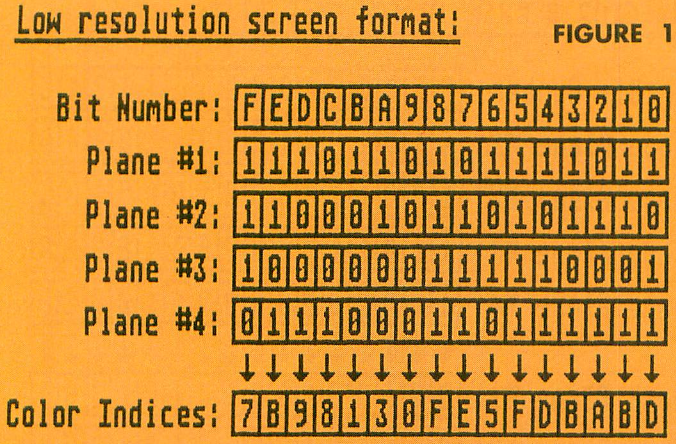
Before you can use *The Generator*, you need to copy the GENERATR.ACC file to the drive you normally boot from (this is usually Drive A: for floppy-based systems or Drive C: for hard-disk-based systems). Now, reboot your computer so it loads the accessory; then run *DEGAS Elite*. You won't be able to use *The Generator* from the GEM desktop, or from any other program, because it was written specifically for *DEGAS Elite*.

Once you have *DEGAS* up and running, load a picture from disk so you can work through the following example. From the main menu screen of *DEGAS*, select the "The Generator. . ." item from the Desk drop-down menu. *The Generator*'s window will then pop up on the screen.

The Screen Buttons

At the top of the window you will see eight buttons labeled SCREEN #1 through SCREEN #8. These buttons correspond to the eight screens that *DEGAS* supports. Clicking the mouse on one of these buttons allows you to select the screen from which you will be cutting an image. If you don't have enough memory to support the maximum of eight screens, *The Generator* will show the unavail-

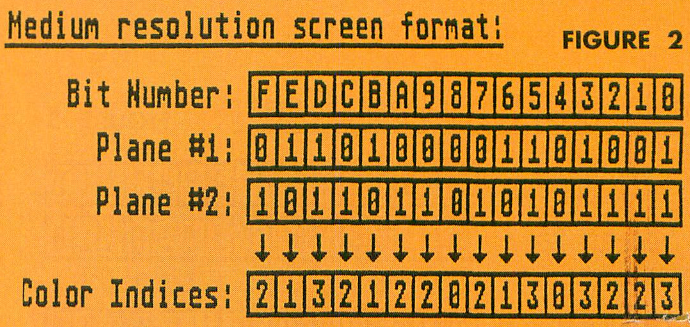
FIGURE 1



Low resolution screen format:

Bit Number:	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
Plane #1:	1	1	1	0	1	1	0	1	0	1	1	1	1	0	1	1
Plane #2:	1	1	0	0	0	1	0	1	1	0	1	0	1	1	1	0
Plane #3:	1	0	0	0	0	0	0	1	1	1	1	1	0	0	0	1
Plane #4:	0	1	1	1	0	0	0	1	1	0	1	1	1	1	1	1
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Color Indices:	7	B	9	8	1	3	0	F	E	5	F	D	B	A	B	D

FIGURE 2



Medium resolution screen format:

Bit Number:	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
Plane #1:	0	1	1	0	1	0	0	0	0	1	1	0	1	0	0	1
Plane #2:	1	0	1	1	0	1	1	0	1	0	1	0	1	1	1	1
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Color Indices:	2	1	3	2	1	2	2	0	2	1	3	0	3	2	2	3

able screens as disabled (lightened text), and you won't be able to select them.

You may have noticed that when you first selected the "The Generator. . ." menu item, there was a short pause before the window actually appeared. This is because the accessory is checking each *DEGAS* screen for picture information. If a screen has data in it, an asterisk will be put in front of the corresponding SCREEN # button. Otherwise, a blank space will precede the SCREEN # button.

The "Cut Image" Button

Once you have a picture in memory, you are ready to cut an image. First, make sure the SCREEN # button selected (the one that is highlighted) indicates the screen containing your picture. Next, click the mouse on the "Cut Image" button. Your picture will be displayed, along with a pair of rubber-banding lines. If you hit a key while these lines are displayed, you will be returned to the main screen.

Move from where the two lines meet to the top-left corner of the section you want to cut, then press the left mouse button. The rubber-banding lines will now change to a rubber-banding box. You may cancel the rubber-band box and go back to the rubber-band lines by pressing any key. Adjust the box's size by moving the mouse, then click the left mouse button to select the lower-right corner of the section you want to cut.

To help you select the exact position and size of the box you want, its coordinates, width and height are printed on the screen. Note that when you are specifying the box, the area under the rubber-banding box is included as part of your image. Once you've selected an image, you will automatically be returned to the main *DEGAS* screen.

The Buffer Buttons

If you haven't cut a section from a *DEGAS* picture, these four buttons will be disabled and you won't be able to select them. However, if you followed the instructions for the "Cut Image" button (see above) and have cut an image, these buttons will be selectable. You can now save the image as source code for C, assembly and GFA BASIC, or as a specially formatted data file that you can load into your program at runtime. Select the button corresponding to the type of data you want to save.

The buttons labeled "Buffer to C," "Buffer to Assembly" and "Buffer to GFA BASIC" let you save the image rectangle you've selected as source code for the respective language. Since the main difference between these buttons is in the output file they generate, the following description applies to all of the buttons. However, any differences will be noted.

When a button is selected, you will be asked if you want to save a "mask" image along with the image. A mask is a special image that is used to clear out the screen area where the main image is going to be placed. This is similar to using a cookie cutter to remove a shape from a sheet of dough. If the mask isn't used, there will be a conflict between the image and whatever is at the same screen location when the regular image is placed on the screen. This results in your image having the wrong colors. We will cover masks in more detail later on in the article.

You will next be asked to enter the name you want to give the source data. For the C selection, this will be used as the name of an array, pre-initialized to contain your image data. For the assembly selection, the name will used as a label marking the beginning of a series of ".dc.w" directives containing the image data. For the GFA BASIC selection, the name will be used as a label marking a series of DATA statements. These DATA statements not only contain your image data, but also contain a special header block which makes it easy to use GFA BASIC's built-in PUT image command to manipulate the image (see the GFA BASIC example program on this month's disk).

If you chose to save the mask image data, *The Generator* will append the string "__mask" to the name you entered, then save the mask image source data in the same manner as the regular image data. For example, if you entered the string "data" as the name for the image data, then the program will save the mask data with the name "data__mask."

If you use *The Generator* to save several images, you might lose track of the names of the images you've already saved. Because of this, a list of the names you've already entered is displayed whenever you're asked to enter a new name.

Last, you will be asked if you want to send *The Generator*'s output to the printer or to a disk file. If you choose the printer for output, the data will be sent immediately. However, if you opted to save the source data to disk, a file selector will be displayed to let you choose the name for the file.

If you try to save the source output to a file that already exists, you will have the option to replace it or append the source output to the existing file. This feature allows you to easily build a file that contains multiple source images to be included in your program.

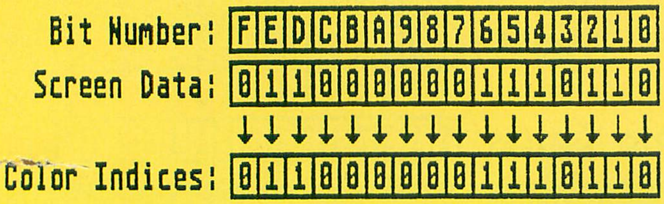
The "Buffer to Data" Button

This button makes *The Generator* even more flexible, especially if the set of languages supported doesn't include the one you normally use to write your programs. Basically, this option saves the image data as a special file that can be loaded into your program when you run it.

When saving your data, *The Generator* produces a file with a specific format. The first word of the file indicates the number of images it contains. The image count does not reflect the presence of any masks in the file;

High resolution screen format:

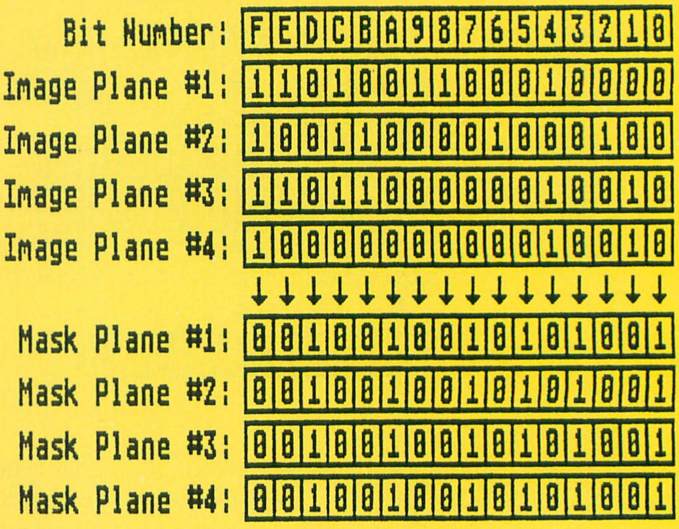
FIGURE 3



Bit Number:	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
Screen Data:	0	1	1	0	0	0	0	0	0	1	1	1	0	1	1	0
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Color Indices:	0	1	1	0	0	0	0	0	0	1	1	1	0	1	1	0

Building a mask in low resolution:

FIGURE 4



Bit Number:	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
Image Plane #1:	1	1	0	1	0	0	1	1	0	0	0	1	0	0	0	0
Image Plane #2:	1	0	0	1	1	0	0	0	1	0	0	0	1	0	0	0
Image Plane #3:	1	1	0	1	1	0	0	0	0	0	1	0	0	1	0	0
Image Plane #4:	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Mask Plane #1:	0	0	1	0	0	1	0	0	1	0	1	0	1	0	0	1
Mask Plane #2:	0	0	1	0	0	1	0	0	1	0	1	0	1	0	0	1
Mask Plane #3:	0	0	1	0	0	1	0	0	1	0	1	0	1	0	0	1
Mask Plane #4:	0	0	1	0	0	1	0	0	1	0	1	0	1	0	0	1

this is contained in a block of information called a header, which precedes every image saved to the file.

The first word in the header contains the image resolution in the low byte, and a flag in the upper byte ($0100) indicates the presence (set) or absence (not set) of a mask image. The next two words are the image width and height, respectively. The last piece of information in the header is a long integer indicating the number of bytes in the image. This long integer only represents the size of the image; if a mask is present, it has the same number of bytes and immediately follows the data for the image. The mask does not have its own header since the image and mask have the same attributes.

Using Images in Programs

Now that you know how to use *The Generator* to output the source code for your image, you may want to write a program that makes use of this data. We will get to this momentarily, but before you can delve into the programming, you need to know how the Atari ST handles graphics.

The ST supports three different formats for the 32,000 bytes it uses for screen memory: low resolution, which supports 320 pixels × 200 pixels with 16 colors; medium resolution, which supports 640 pixels × 200 pixels with four colors; and high resolution which supports 640 pixels × 400 pixels with two colors. While knowing the dimensions of the screen is important, you also need to know how the ST's screen-memory information is arranged. Unfortunately, this arrangement can be confusing because it varies depending on the current resolution.

In the low-resolution mode, the memory used for the screen display is separated into four sections, commonly called bit-planes. To determine the color of any pixel, a bit is taken from the same position in each bit-plane and combined to form a color index for that pixel. The screen-memory information is arranged such that the first four words (a word is 16 bits) represent planes one through four and thus denote the first 16 pixels on the screen (located in the upper-left corner). The next four words denote the bit-planes for the next 16 pixels on the screen, and so on, for the remainder of screen memory. This scheme is known as an interleaved bit-plane format because all the data is mixed together.

Since four bit-planes are used, 16 color-index values are possible. It is important to note that the color-index value doesn't represent a fixed color, such as red or yellow; instead it is used to access a color-palette table of 16 values. The values in the color-palette table hold the red, green and blue (RGB) content for the pixel's color.

In the medium-resolution mode the screen memory is still separated into bit-planes. However, this time there are only two bit-planes. Therefore, a pixel is made up of two bits, which gives four different color-index values. The color-palette table is accessed in the same manner as it is for low resolution, but only the first four entries in the table are used.

The high-resolution mode is the easiest to understand because only one bit-plane is used. This results in only two possible color-index values for any pixel. Also, now only the first two palette-table entries are used.

Refer to Figures 1, 2 and 3 for examples of how the color indices are formed in each resolution.

Now that we've covered the basics of the Atari ST's screen format, we can now put an image on the screen. For the following example we'll be using the C language to describe the process.

The first step is to define the data you want to display. For example, assume we have created a 16×16-pixel image and given it the name "shape." The Atari ST's operating system supports many different types of functions to aid the programmer. Fortunately, one of these functions enables us to move blocks of memory from one location to another in several ways. The function we will use to copy our image is *vro__cpyfm()*. Here's an example of what it looks like in C:

```
vro_cpyfm( handle, mode,
pxyarray, &image, &screen );
```

As you can see, *vro__cpyfm()* requires several rather cryptic-looking arguments. Among these arguments are the addresses of two data structures called Memory Form Definition Blocks, or MFDBs for short. The MFDBs are used to hold information about the memory areas that we will be manipulating. The *&image* argument is the address of the MFDB for the image we want to copy (called the "source"), and the *&screen* argument is the MFDB for the area of memory to which we want to copy the image (called the "destination"). Although the MFDB structure is defined in the GEMDEFS.H header file, we have included it here for clarity:

```
typedef struct(
    long fd_addr;    /* Address of image          */
    int  fd_w;       /* Width of image in pixels */
    int  fd_h;       /* Height of image           */
    int  fd_wdwidth; /* Width of image in words   */
    int  fd_stand;   /* Bit-plane format          */
    int  fd_nplanes; /* Number of bit planes      */
    int  fd_r1;      /* Reserved                  */
    int  fd_r2;      /* Reserved                  */
    int  fd_r3;      /* Reserved                  */
) FDB;
```

To declare our source and destination MFDBs in C, we simply do the following:

```
FDB image, screen;
```

Now we need to initialize both of the data structures with the necessary information:

```
image.fd_addr = shape;          /* shape data address         */
image.fd_w = 16;                /* 16 pixels wide             */
image.fd_h = 16;                /* 16 pixels high             */
image.fd_wdwidth = 1;           /* Width of image in words    */
image.fd_stand = 0;             /* Interleaved bit-planes     */
image.fd_nplanes = 4;           /* Low Res has 4 bit planes   */
image.fd_r1 = 0;                /* Reserved                   */
image.fd_r2 = 0;                /* Reserved                   */
image.fd_r3 = 0;                /* Reserved                   */

screen.fd_addr = Physbase(); /* Screen memory address         */
screen.fd_w = 320;              /* 320 pixels wide            */
screen.fd_h = 200;              /* 200 pixels high            */
screen.fd_wdwidth = 20;         /* Width of image in words    */
screen.fd_stand = 0;            /* Interleaved bit-planes     */
screen.fd_nplanes = 4;          /* Low Res has 4 bit planes   */
screen.fd_r1 = 0;               /* Reserved                   */
screen.fd_r2 = 0;               /* Reserved                   */
screen.fd_r3 = 0;               /* Reserved                   */
```

Next, we need to specify the rectangular region within the source MFDB from which we want to copy and the rectangular region within the destination MFDB that we want to copy it to. This is done by setting up an array that contains the coordinates for the upper-left and lower-right corners for both the source and destination rectangles. For our purposes, we will be copying the entire 16×16-pixel area defined in the source MFDB to a 16×16 area somewhere in the destination MFDB. To do this, we first need to declare the array for the coordinates:

```
int pxyarray[8];
```

Then, somewhere before we actually perform the *vro__cpyfm()* call, we need to initialize the array:

```
pxyarray[0] = 0;
pxyarray[1] = 0;
pxyarray[2] = 15;
pxyarray[3] = 15;
pxyarray[4] = 100;
pxyarray[5] = 100;
pxyarray[6] = 115;
pxyarray[7] = 115;
```

Finally, we have several options as far as the way the source image is combined with the destination image. One option is to have the source rectangle completely replace the destination rectangle. Or, we may perform logic operations between the two rectangles, such as ANDs, ORs, Exclusive ORs, etc. For our example, we will use mode 7, which tells *vro__cpyfm()* to logically OR the source rectangle onto the destination rectangle. As we will see in a moment, this will cause us some problems.

Assuming we have opened a graphics workstation and have our "handle" for the virtual workstation, we can now move the source rectangle to its location within the destination rectangle:

```
vro_cpyfm( handle, 7, pxyarray,
&source, &dest );
```

The problem with using the OR mode is that if a nonzero pixel in the source rectangle is ORed with a nonzero pixel in the destination rectangle, the result is an undesired pixel value. For example, if a pixel in the source image is color 3 and the pixel in the destination image that it is ORed with is color eight, the resulting pixel value will be color 11. To alleviate this problem, we must have some way of clearing all the pixels in the destination rectangle that will be ORed with a nonzero pixel in the source rectangle before we copy the main image to the screen. This technique is known as bit-masking, or masking for short.

A mask is an image whose contents are derived from the primary image that you want to place on the screen. Fortunately, *The Generator* will output the mask information for you, but just so you know what's going on, the mask is made according to the following rules (also, refer to Figure 4):

(1) If the color of a pixel in the main image is not zero, then the color of corresponding pixel in the mask will be set to zero.

(2) If the color of a pixel in the main image is zero, then the color of the corresponding pixel in the mask is set to color 15. In other words, the corresponding bits in each of the four planes are set to ones.

Now that you know what a mask is, you may be wondering how to put it on the screen. In effect, you can do this in almost exactly the same manner in which you put the main image up in the previous explanation of *vro__cpyfm()*. The only difference is that you must AND it to the screen. Since the mask contains a color zero pixel for every nonzero pixel in the main image, the exact area that will be taken up by the main image will be erased. Also, since the mask contains ones for every zero pixel in the main image, those pixels will be left undisturbed!

To use the mask, first declare an MFDB for the mask:

```
FDB mask;
```

Then, before copying the mask to the screen, initialize the MFDB:

```
mask.fd_addr = mask_image;  /* mask data address          */
mask.fd_w = 16;             /* 16 pixels wide             */
mask.fd_h = 16;             /* 16 pixels high             */
mask.fd_wdwidth = 1;        /* Width of image in words    */
mask.fd_stand = 0;          /* Interleaved bit-planes     */
mask.fd_nplanes = 4;        /* Low Res has 4 bit planes   */
mask.fd_r1 = 0;             /* Reserved                   */
mask.fd_r2 = 0;             /* Reserved                   */
mask.fd_r3 = 0;             /* Reserved                   */
```

Notice that the mask's MFDB and the MFDB for the main image are almost identical. Also, since the source and destination rectangles we've already defined in the *pxyarray* (see above) are the coordinates we want, we can use the array for copying both the mask and the main image:

```
vro_cpyfm( handle, AND, pxyarray, &mask, &screen );
vro_cpyfm( handle, OR, pxyarray, &source, &screen );
```

We now have copied the mask and the image to the screen without the ORing problem.

About the Examples

As we mentioned earlier, we have written some example programs that show you how to use the image data saved by each of *The Generator*'s output selections. The C language and the data-file demonstration programs were written with Megamax C and use many of the same techniques described in the *vro__cpyfm()* tutorial above.

The 68000 assembly demonstration was written with the Mad Mac assembler from Atari and uses a low-level Line-A operation to copy the image to the screen. While the Line-A example might be harder to follow than the other examples, the program is thoroughly commented, so you should be able to understand it if you're familiar with assembly language.

The GFA BASIC demonstration uses the header information at the beginning of the DATA statements to set up the information needed by the PUT command. Note that this header is not saved in the source code files for C and assembly; it is only available in GFA BASIC.

The DEGAS Elite Image Generator was written to save us the time and trouble of defining our bit-mapped images by hand. In fact, writing *The Generator* has already turned out to be time well spent, since we've used it to create images for several other programs we've developed. If you're a graphics-oriented programmer, we feel that it will prove to be a useful utility for you as well.

Robert Birmingham *is 26 and lives in Miami. Although he has been programming for nine years, he remains thoroughly fascinated with computers. In his spare time he enjoys listening to music, building model rockets, juggling and stargazing.*

Richard Leinecker *is a geometry teacher at South Miami Senior High School. He's also a senior programmer/analyst at IntraCorp, Inc. He has programmed numerous pieces of entertainment software and has written several hardware-project books for the ST.*

DISK LISTING

8 BIT

THE DECEMBER 1989 ANALOG COMPUTING 8-BIT DISK CONTAINS 8 MAGAZINE FILES. THEY ARE LISTED BELOW:

SIDE 1:

FILENAME.EXT	LANG.	LOAD	ARTICLE NAME
ASMED .SRC	ASM/ED	ENTER	ASM/ED REFERENCE
DIR3 .OBJ	ML	(#3)	DIR3
DIR3 .M65	MAC/65	LOAD	DIR3 SOURCE CODE
SKYRISER.BAS	BASIC	LOAD	SKYRISER
TRUCHET .BAS	BASIC	LOAD	TRUCHET TILES
ADDRESS2.BAS	BASIC	LOAD	BASIC TRAINING
MLEDITOR.BAS	BASIC	LOAD	M/L EDITOR
EDITORII.LST	BASIC	ENTER	BASIC EDITOR II

TO LOAD YOUR ANALOG DISK

1) INSERT BASIC CARTRIDGE (NOT REQUIRED FOR XE OR XL COMPUTERS).
2) TURN ON DISK DRIVE AND MONITOR.
3) INSERT DISK IN DRIVE.
4) TURN ON COMPUTER. (XL AND XE OWNERS: DO NOT HOLD DOWN OPTION KEY!)

WARNING: BEFORE YOU RUN A PROGRAM, READ THE APPROPRIATE ARTICLE IN THE MAGAZINE. FAILURE TO DO SO MAY YIELD CONFUSING RESULTS.

NOTE: ONLY PROGRAMS WITH THE .BAS, .COM OR .OBJ EXTENSION MAY BE RUN FROM THE MENU. OTHER PROGRAMS SHOULD BE LOADED AS INSTRUCTED IN THE LOADING NOTES AND MAY REQUIRE ADDITIONAL SOFTWARE AS LISTED BELOW. HOWEVER, YOU SHOULD NOT ASSUME THAT EVERY FILE WITH THE PROPER FILE EXTENSION WILL RUN FROM THE MENU. YOU MAY HAVE TO MOVE CERTAIN PROGRAMS TO A DIFFERENT DISK TO OBTAIN CORRECT RESULTS.

EXT	DESCRIPTION
.M65	REQUIRES THE MAC/65 ASSEMBLER
.AMA	REQUIRES THE ATARI MACRO ASSEMBLER
.ASM	REQUIRES THE ATARI ASSEMBLER/EDITOR
.ACT	REQUIRES THE ACTION! CARTRIDGE
.LGO	REQUIRES THE ATARI LOGO CARTRIDGE
.SYN	REQUIRES THE SYNAPSE SYN ASSEMBLER

LOADING NOTES

LOAD BASIC PROGRAM:	LOAD "D:FILENAME.EXT"
ENTER BASIC PROGRAM:	ENTER "D:FILENAME.EXT"
LOAD MAC/65 PROGRAM:	LOAD #D:FILENAME.EXT
ENTER ASM/ED PROGRAM:	ENTER #D:FILENAME.EXT
LOAD LOGO PROGRAM:	LOAD "D:FILENAME.EXT"
LOAD SYN/AS PROGRAM:	LOAD "D:FILENAME.EXT"

#1: SEE ACTION! MANUAL.
#2: SEE ATARI MACRO ASSEMBLER MANUAL.
#3: MAY ALSO BE LOADED FROM DOS USING THE "L" OPTION OF THE DOS MENU.
#4: THIS FILE SHOULD BE TRANSFERRED TO ANOTHER DISK AND RENAMED "AUTORUN.SYS".
#5: READ THE APPROPRIATE ARTICLE FOR INSTRUCTIONS ON USING THIS FILE.

ST

The DECEMBER 1989 ANALOG COMPUTING ST disk contains 27 magazine files. They are listed below.

FILENAME.EXT	FILE TYPE	COMMENTS
CMANSHIP.ARC contains:		
C79 .C	C	C-MANSHIP
COMB079 .PRG	RUN FILE	THE PROGRAM SO FAR
MICROCHK.RSC	RESOURCE	
MICROCH2.C	C	MICROCHECK SOURCE
COMPUKID.ARC contains:		
DREIDEL .PRG	RUN FILE	DREIDEL
DREIDEL .DAT	DATA	
SPIN .DAT	DATA	
DREIDEL .LST	GFA BASIC 2.0	SOURCE CODE
STARTGEM.PRG	RUN FILE	START GEM
STARTGEM.INF	DATA	
DIALOG.ARC contains:		
DIALOGTT.LST	GFA BASIC 3.0	DIALOG DEMO SOURCE
BENEFITS.RSC	RESOURCE	
BENEFITS.LST	GFA BASIC 3.0	HEADER FILE
BENEFITS.DFN	DATA	RCS DATA FILE
GENERATR.ARC contains:		
GENERATR.ACC	ACCESSORY	IMAGE GENERATOR
GENERATR.C	C	SOURCE CODE
CUT .C	C	SOURCE CODE
SAVE .C	C	SOURCE CODE
BITMAP .H	C	HEADER FILE
EXTDEFS .H	C	HEADER FILE
GROOVES.ARC contains:		
GROOVES .PRG	RUN FILE	TURTLE GROOVES
GROOVES .GFA	GFA BASIC 3.0	SOURCE CODE
TURT_SET.1	DATA	LEVEL 1 DATA
TURT_SET.2	DATA	LEVEL 2 DATA
TURT_SET.3	DATA	LEVEL 3 DATA
TURT_SET.4	DATA	LEVEL 4 DATA
TURT_SET.5	DATA	LEVEL 5 DATA
ARCX .TTP	RUN FILE	UNARCHIVING PROGRAM
README .DOC	TEXT	Disk instructions
UNARCHIV.DOC	TEXT	Unarchiving instructio

Disk instructions:
Only those files with .PRG, .TOS, or .TTP extensions may be run from the GEM desktop. Other programs may require additional software as shown below. The files on this disk have been archived (compressed) into .ARC files. To restore the programs to their runnable form, follow the instructions found in the UNARCHIV.DOC file.

WARNING: Be sure to read the appropriate magazine articles before attempting to run the program on this disk. Failure to do so may yield confusing results.

.EXT	DESCRIPTION
.GFA	Requires GFA BASIC 3.0
.LST	Requires GFA BASIC
.C	Requires C compiler
.S	Requires 68000 assembler
.PAS	Requires Pascal compiler

```
PM 10 REM ***************************
MI 20 REM *      BASIC TRAINING       *
AX 30 REM *      ADDRESS BOOK 2       *
PZ 40 REM *     by Clayton Walnum     *
VS 50 REM *                           *
FG 60 REM *      Copyright 1989       *
RT 70 REM *    by ANALOG Computing    *
PT 80 REM ***************************
BG 90 REM
XR 100 DIM NAME$(30),ADDRESS$(30),CITY$(3
   0),PHONE$(15),A$(1),FILENAME$(15)
RA 101 DIM TEMP$(15)
QP 102 CHECKFILENAME=801:STRINGERROR=5:FI
   LENOTFOUND=170
QO 110 REM
YA 120 REM ****************************
BC 130 REM *        PRINT MENU         *
YE 140 REM ****************************
QW 150 REM
BE 160 ? CHR$(125)
JA 170 ? "┌──────────────────────────┐"
IE 180 ? "│      CREATE FILE         │"
EG 190 ? "│                          │"
QC 200 ? "│      LOAD FILE           │"
DR 210 ? "│                          │"
GE 220 ? "│      VEIW FILE           │"
DV 230 ? "│                          │"
UP 240 ? "│      QUIT                │"
JJ 250 ? "└──────────────────────────┘"
ZU 260 INPUT A$
GO 270 IF A$="C" OR A$="c" THEN 370
RE 280 IF A$="L" OR A$="l" THEN 450
TV 290 IF A$="Q" OR A$="q" THEN END
JO 300 IF A$="V" OR A$="v" THEN 680
IP 305 ? "PLEASE CHOOSE C, L, V, OR Q."
OI 310 GOTO 260
QS 320 REM
YE 330 REM ****************************
WW 340 REM *       CREATE FILE         *
YI 350 REM ****************************
RA 360 REM
BR 370 ? "FILENAME";:INPUT FILENAME$
ZF 371 GOSUB CHECKFILENAME:IF FILENAME$="
   " THEN 370
ZO 372 TRAP 379:OPEN #1,4,0,FILENAME$
WV 373 CLOSE #1:? :? "FILE ALREADY EXISTS
   ! ERASE IT (Y/N)";:INPUT A$
LA 374 IF A$="Y" OR A$="y" THEN 380
VO 375 IF A$="N" OR A$="n" THEN 370
GS 376 ? :GOTO 373
MO 379 CLOSE #1
AJ 380 OPEN #1,8,0,FILENAME$
OK 390 GOTO 520
QP 400 REM
YB 410 REM ****************************
SI 420 REM *        LOAD FILE          *
YF 430 REM ****************************
QX 440 REM
BO 450 ? "FILENAME";:INPUT FILENAME$
XG 451 GOSUB CHECKFILENAME:IF FILENAME$="
   " THEN 450
RT 452 TRAP 468
AT 460 OPEN #1,9,0,FILENAME$
OL 462 GOTO 520
DE 468 CLOSE #1:IF PEEK(195)=FILENOTFOUND
    THEN ? "NO SUCH FILE! TRY AGAIN.":GOT
   O 450
FM 469 ? "UNDETERMINED ERROR! TRY AGAIN."
   :GOTO 450
RD 470 REM
YP 480 REM ****************************
UX 490 REM *       GET ADDRESS         *
YA 500 REM ****************************
QS 510 REM
VR 520 ? :? "NAME:":INPUT NAME$:IF NAME$=
   "" THEN 620
FY 530 ? :? "ADDRESS:":INPUT ADDRESS$:IF
   ADDRESS$="" THEN 620
MT 540 ? :? "CITY, STATE & ZIP:":INPUT CI
   TY$:IF CITY$="" THEN 620
ON 550 ? :? "PHONE:":INPUT PHONE$:IF PHON
   E$="" THEN 620
WI 560 ? :? NAME$:? ADDRESS$:? CITY$:? PH
   ONE$
EK 570 ? :? "IS THE ABOVE ENTRY OKAY (Y/N
   )";:INPUT A$
FT 580 IF A$="N" OR A$="n" THEN ? :? "PLE
   ASE REENTER THE ADDRESS.":GOTO 520
ST 590 IF A$<>"Y" AND A$<>"y" THEN ? :? "
   PLEASE ANSWER Y OR N.":GOTO 570
IG 600 ? #1;NAME$:? #1;ADDRESS$:? #1;CITY
   $:? #1;PHONE$
NX 610 GOTO 520
BR 620 CLOSE #1:GOTO 160
QX 630 REM
YJ 640 REM ***************************
MO 650 REM *        VIEW FILE         *
YN 660 REM ***************************
RF 670 REM
BW 680 ? "FILENAME";:INPUT FILENAME$
GN 681 GOSUB CHECKFILENAME:IF FILENAME$="
   " THEN 680
QZ 683 TRAP 772
YO 690 OPEN #1,4,0,FILENAME$
OS 700 TRAP 760
UQ 710 COUNT=0:? CHR$(125)
ET 720 INPUT #1;NAME$:INPUT #1;ADDRESS$:I
   NPUT #1;CITY$:INPUT #1;PHONE$
TR 730 ? NAME$:? ADDRESS$:? CITY$:? PHONE
   $
GN 740 COUNT=COUNT+1:IF COUNT=4 THEN ? :?
    "PRESS [RETURN] FOR MORE":INPUT A$:GO
   TO 710
DP 750 ? :GOTO 720
WC 760 ? :? "END OF FILE.":? "PRESS [RETU
   RN] FOR MENU."
LS 770 INPUT A$:CLOSE #1:GOTO 160
RZ 772 CLOSE #1:IF PEEK(195)=FILENOTFOUND
    THEN ? "NO SUCH FILE! TRY AGAIN.":GOT
   O 680
NZ 775 ? "UNDETERMINED ERROR! TRY AGAIN."
   :GOTO 680
RZ 795 REM
ZM 796 REM ***************************
GZ 797 REM *     CHECK FILENAME       *
ZS 798 REM ***************************
SL 799 REM
OM 801 TRAP 809:IF FILENAME$(1,1)<"A" OR
   FILENAME$(1,1)>"Z" THEN 809
OO 802 TRAP 808:IF FILENAME$(2,2)=":" OR
   FILENAME$(3,3)=":" THEN 805
LY 804 TEMP$="D:":TEMP$(3)=FILENAME$:FILE
   NAME$=TEMP$
RA 805 X=2:IF FILENAME$(X,X)<>":" THEN X=
   3
FM 806 IF FILENAME$(X+1,X+1)>="A" AND FIL
   ENAME$(X+1,X+1)<="Z" THEN RETURN
TE 807 GOTO 809
OV 808 IF PEEK(195)=STRINGERROR THEN 804
CE 809 ? "FILENAME ERROR! PLEASE TRY AGAI
   N.":? :FILENAME$="":RETURN
```

ONCE WE'VE GOT OUR PROGRAM RUNNING, WE MUST GO THROUGH IT AND MAKE SURE IT CAN "NEVER" BOMB OUT ON THE USER. THIS IS A BIG JOB. SOMETIMES IT TAKES AS MUCH PROGRAMMING TO DO THIS AS IT DID TO WRITE THE PROGRAM IN THE FIRST PLACE.

FOOTNOTES

GREAT COMPUTING MYTHS, PART I

by Karl E. Wiegers

"The Paperless Office." Remember that phrase? It was popular some years back when the soothsayers predicted that the mounds of paperwork filling the average business office would be replaced by the efficient and pollution-free flow of electrons among coworkers. Nice idea. But I'm here today to tell you that it ain't gonna happen. In fact, quite the opposite has taken place. The ubiquitous presence of the computer in America's offices has caused the demise of perfectly good trees at an unprecedented rate. Let me explain some of the reasons why the paperless office hasn't become a reality, using my own workplace as an example.

One problem is that, when wildly predicting the all-electronic office, no one anticipated the need for computer manuals. In the olden days, you had room in your office for a few mementos, family pictures and humorous parting gifts from subordinates. No more: Every square inch is likely to be occupied by obtuse publications purporting to make your working hours more productive. Even the most computer-phobic manager has a tidy row of manuals in his office, albeit still in the shrink-wrap.

It's worse for us professional software types. Being a scientist by training, I decided to quantify the extent to which printed manuals have filled my office. As of today, my office (about eight feet square) contains some 175 computer manuals and books: 76 user guides for different applications programs (counting the ones I wrote myself); 40 reference manuals for eight different programming languages; 38 treatises on various aspects of software development; 17 operating systems manuals for three different computer systems; four hardware-reference manuals; zero partridges in pear trees.

Now, these publications occupy quite a bit of space. More precisely, they use up 18 linear feet of shelf space, 14 square feet of shelf space or 12 cubic feet of the office. And my best estimate of the combined weight of the paper contained in all this literature is 715 pounds. I can't even guess at the number of pages of computer knowledge contained in my office. It appears that the notion of on line computer help has a ways to go before I can toss out all these tomes.

Manuals aren't the only scourge acting to oppose the paperless office. Consider electronic mail. The early enthusiasts anticipated that electronic mail would greatly reduce the volume of paper being circulated and stored. But it didn't work out that way, for a number of reasons.

To be sure, electronic mail is fast and efficient. But it's still easier to read things on paper than on the screen, so most people still print out the notes they exchange. It's hard to do the electronic equivalent of flipping back and forth among several pieces of paper using your standard 80-column × 24-line display. And it's darned near impossible to proofread something effectively on the tube, although electronic spelling-checkers keep a lot of people from making silly mistakes.

Electronic mail is so much fun to use that people send notes when a phone call would suffice, thereby giving them something else to print out. The electronic-mail system my employer uses has the additional quirk of printing a full header page for each note, which doubles the paper consumed when printing your average one-page note.

Here's another problem: Before the days of electronic mail, a secretary might have typed up some meeting notes or a memo, run off 30 copies and mailed them to the recipients. Now you send the notes electronically to 60 people (which is just as easy as to 30), each of whom uses valuable printer and computer time printing out his own copy, complete with header page. Sounds like a step away from the paperless office to me.

Take word processing (please). Pre-computer days, a typed report or letter would usually go out after no more than one or two drafts. But now it might take four or five printouts to get it looking just the way you want, particularly if you aren't using a WYSIWYG (what you see is what you get) computer system that allows accurate screen previews. Paper, paper, everywhere. The good news is that the quality of the final product is usually higher these days, because all documents benefit from editing and revision.

For security, space or legal reasons, many companies are concerned about excessive retention of documents. Electronic files now fall under the same kinds of records-management rules. How do you deal with this? Someone tells you to erase an obsolete file, so you print out a copy just to be sure it isn't gone forever in case you need it again some day. You stuff it in your desk, along with the other archival records. Basically, we're just like squirrels, stashing away paper instead of acorns.

How about electronic meeting notices? Great idea: Everyone gets the same notice at the same time. It's easy to send out revisions, too. But do you carry your computer terminal around everywhere in case you want to check your calendar? Of course not. You print out the meeting notices and attach them to the side of your file cabinet with magnets. Our computer system has an electronic calendar and scheduling facility, fully equipped with a utility to print out your daily calendar in a handy pocket-size format. That's what I call facing reality.

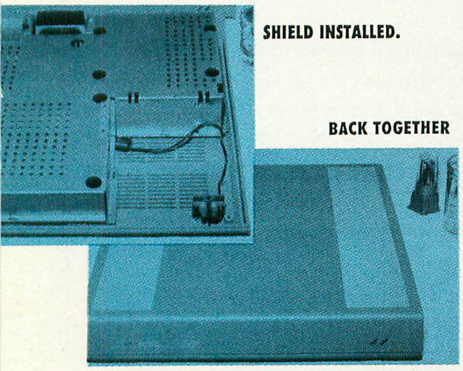
Software developers aren't any closer to the paperless office than anyone else. The advent of computer-aided software engineering (CASE) tools lets us do a lot of our design work on the screen, rather than on the backs of envelopes or whatever. But most systems can only show part of one diagram at a time, so we have to print the things out to study the overall picture or to show our design to someone else. And CASE tools make it easy to redraw the same diagram over and over, making it better each time (I hope). Naturally, I have to print the whole thing out each time so I can see if I'm done yet. I actually had a guy tell me once that he supported CASE because it would help people get the paper off their desks! This fellow had clearly never seen the mountains of printouts in an office like mine.

Don't get me wrong; I like computers. I like the ways computers can make my life easier and I like wrestling with them (to a point) when they try to make my life more difficult. But I think the notion that computers will eliminate, or even reduce, paperwork is naive. I hate to see all that virgin timber crashing down so that I might feed a laser printer. But I do what I can to compensate. All of my scratch paper consists of the backs of rejected printouts, not nice new legal pads from the stockroom. I do my best work on the backs of someone else's header pages.

***Karl Wiegers**, Ph.D., spent the '70s learning how to be an organic chemist, then spent the '80s wrestling with computers. He is now a software engineer in the Eastman Kodak Company photographic research labs. He hasn't selected a career for the '90s yet. It may be interesting to note that, although this article was submitted and edited on disk, a hard copy accompanied the original submission. After editing, a final manuscript was printed and sent to the production department (along with the disk), where the manuscript was photocopied numerous times for distribution to several other departments. Once the file was transferred from disk to the typesetting equipment, at least six versions of the galleys (the typeset article) were generated before the final page was shipped to the printer. Somewhere, a forest is screaming.*

ANALOG Computing *invites all authors to submit essays for possible use in the* Footnotes *column. Submissions should be no longer than 1,500 words and may be on any aspect of Atari computing. Any style or type of essay is acceptable—opinion, humor, personal experience—but creativity is a plus. Submissions should be sent to:* Footnotes, *c/o* ANALOG Computing, *P.O. Box 1413-M.O., Manchester, CT 06040-1413.*

(CONTINUED FROM PAGE 117) MEGAFILE



SHIELD INSTALLED.

BACK TOGETHER

new drive. If needed, you can alter these settings. However, unless you know exactly what you're doing, I strongly urge you not to mess with them unless you enjoy courting disaster. On the "Controller Type," you'll need to click on Adaptec 4000. This is the controller installed in your MegaFile 20. If you can't find your hard disk listed in the display, Call Supra customer service for assistance.

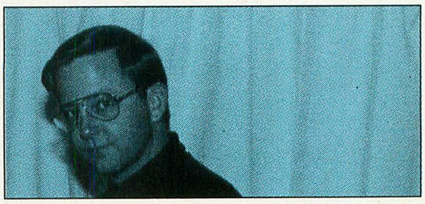
Once you've selected the appropriate hard disk and controller, click on OK and follow the directions on the screen as they appear. The process is painless as you select the number and size of partitions you want and start formatting. Be forewarned that the process will take close to 40 minutes from start to finish.

After formatting, select the SUPUTL-.PRG to map out any bad sectors and install your auto-boot. I suggest that you take the time to run the "map" utility; it can save you much future pain. With these tasks completed, you've finished your upgrade and are ready to start reloading your data.

Congratulations! You've doubled your storage and noticeably increased your data-transmission rate. I ran RATE.HD on my upgrade and found the seek rate was reduced to 42Msec and the transmission rate increased to 382K/sec.

A warning: While the Seagate ST251 is generally a quiet drive, it makes a sound like a diesel throwing a rod when you power it down. This is the auto-park engaging, so you don't need to worry about it. For the same reason, the unit's startup is a little noisier than usual. I wasn't expecting it and almost had a heart attack when I powered down for the first time.

So there you have it: 42 megabytes of high-speed hard disk for a cash outlay of only $405 ($350 for the drive, $15 for shipping and $40 for Supra's software). Deduct the $125 I made selling the original mechanism, and my total cost was $280. Purchasing a new 40-megabyte hard-disk system from Supra, ICD or any of the "build it yourself" folks could easily run $800 and up. So how does it feel to have just saved $520? I don't know about you, but this little miser feels just great!



***Gregg Anderson**, a captain in the United States Air Force with a background in electronics repair, has been an avid Atari user since 1982.*

Companies Mentioned in This Article:

ABCO Computers
P.O. Box 6672
Jacksonville, FL 32236
(904) 783-3319

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

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

Supra Corp.
1133 Commercial Way
Albany, OR 97321
(503) 967-9075

Seagate
920 Disc Drive
Scotts Valley, CA 95066
(800) 468-3475

Tech Specialties
909 Crosstimbers
Houston, TX 77022
(713) 691-4527

ST GOSSIP FROM Hollywood, USA

by TG

The "New" ST

During a recent phone conversation with a friend who has some inside lines to Atari Corp., I learned that Atari has a new ST model waiting in the wings. My first reaction to this news was to wonder how Atari could possibly think of adding yet another model to a line that already contains so many machines (the 520, 1040, Megas, Stacy). The word from the top seems to be that this new ST model will replace several if not all of the current machines (except the Stacy). This machine has been called the EST and ST+, but my sources tell me the real name Atari uses is the "STE" (E for "Extended" or "Enhanced").

Although the exterior form of this machine hasn't been made final yet, the internals have. A number of changes make the STE a better machine than any of the current ST models. First, the mouse/joystick ports can now be used for output, something the old 8-bit Ataris could do that the ST couldn't. Second, Atari has managed to crunch down the number of parts again. Word is that the MMU and GLU circuits are now contained in one chip.

Of more importance to the average user, the video chip has been modified to support 16 levels for each of the RGB settings, rather than the eight levels of older STs. This means the total palette has been expanded from 512 colors to a more desirable 4,096 colors. This should encourage software developers to provide support for this larger palette, which may also help people who have installed color-upgrade boards into their current STs.

Sound has long been the Achilles' heel of the ST computers. The ST's sound chip is a relatively old one that pales in comparison to the sound hardware of computers like the Amiga and Apple IIGS. The STE addresses this problem through its inclusion of a new sound chip that provides much better audio via stereo output. This chip is not the AMY sound chip that Atari had announced four years ago. Apparently, AMY is still bogged down in a technical and legal quagmire.

Word from sources close to Atari indicate that the current plan is to release a single STE model. This machine will be configured much like a 1040ST and contain one double-sided microfloppy drive and one megabyte of RAM. For those who need or want more memory, the STE allows this by providing plug-in connectors for SIM (Single Inline Memory) modules (such as are used in some Macintosh models). In this way, users can easily add RAM just by plugging it in, eliminating the need for hardware hacking.

And, in spite of all rumors to the contrary, the STE will not feature a 68020 microprocessor. It will contain the same 8-MHz MC68000 processor that all other STs have.

In light of this, it seems reasonable to assume that Atari will discontinue production of the current 520ST and 1040ST models. Not to worry, though. Atari is working hard to make sure the STE is completely compatible with the current ST. When running normal ST software, the STE will act like any of the current models. It will only perform differently when STE-specific instructions are sent to it.

TOS 1.6 and Up?

At this writing, the ROM version of the long-awaited TOS 1.4 is only available to licensed ST developers, and the general ST public is still waiting for this upgrade. But, even as users impatiently await the release of TOS 1.4, Atari is working on further revisions.

Many ST users who have gotten wind of this rumored TOS 1.6 have become upset, feeling that by the time they can get 1.4, Atari will be ready to replace it. Well, the word I have gotten is that these further TOS upgrades aren't specifically for the ST. The upgrading is part of an ongoing development process aimed at new machines coming down the pike, such as the 68030 TT machine. Sources I spoke to at Atari indicated that they have never been happy with certain aspects of TOS, and not particularly with the version of the GEM Desktop contained in the ST's operating system. In light of user comments and Atari's own feelings, it was decided to improve things for future machines.

The source code for these future versions of TOS is being written in a non-compiler-specific form, which will allow TOS to be ported and compiled in C compilers other than Atari's own Alycon C, which all current versions of TOS are written specifically for. This will allow TOS to be ported to new machines and processors more easily (without first having to await Alycon C to be converted).

As to the GEM Desktop, taking the cue from users and from third-party software developers who have "added" features to GEM, Atari is looking to incorporate some of these new and improved ideas to make the Desktop easier to work with. One source at Atari has hinted that a mock-up of the new Desktop is in the works, and that while current GEM users will find it instantly familiar, they will also discover it is easier to use and more intuitive.

The 68030 TT

My sources tell me that, yes, the TT does exist. The TT is the next-generation machine beyond the current ST models, and as such is much faster and more powerful, capable and *expandable* than any ST or even the STE.

The TT's heart is a 16-MHz 68030 microprocessor. This pure 32-bit chip is many times faster and can directly address more memory than the 68000 microprocessor used in the ST. With a caching program installed, the TT is rumored to be approximately four to five times as fast as any ST.

Some facts about this machine have become known: It will contain the same sound chip as the STE and also feature a 4,096-color palette. As to graphics, there will be six resolutions listed under the "Set Preferences" menu on the Desktop. They are:

- Low, 320×200 with 16 colors (as on the ST).
- Medium, 640×200 with four colors (as on the ST).
- High, 640×400, monochrome using an SM124 monitor (as on the ST), or using two colors if using a multisynch monitor.
- 320×480, 256 colors, requires a multisynch monitor.
- 640×480, 16 colors, requires a multisynch monitor.
- 1280×960, monochrome, requires a special high-resolution monitor.

Atari is apparently looking into the possibility of TOS/GEM support for plug-in 24-bit graphics cards, which would allow very high resolutions capable of using hundreds or thousands of colors out of a 16 million-color palette, such as the high-end graphics board used in the ATW (Atari Transputer Workstation).

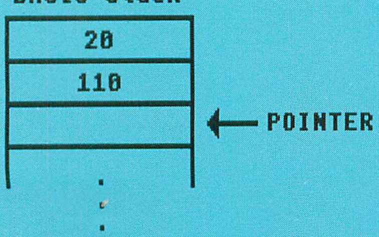
As to expandability, the TT will feature all of the current ST ports and add an SCSI (Small Computer Systems Interface) port (for connecting non-Atari hard drives and other devices), two RAM expansion connectors, two VME bus interfaces (the same type used in Sun workstations) and possibly even sockets for high-speed RAM chips.

As with the STE, ST compatibility is a serious concern of Atari's, even though the TT can be set up to run Unix System V. One major problem crops up because of a few instruction incompatibilities between the 68000 and the 68030 (as reported a few months ago), but as I reported then, Atari is making an effort to provide programmers with a system call that will allow the software to check if it is running on an ST or a TT, allowing the program to compensate for any differences in the machines. This will require authors to update their programs, but it should mean that almost any ST software that won't normally run on a TT can be easily fixed to do so. ▣

After a long and relaxing stay at the Institute for the Potentially Nervous, **TG** *has decided to give up his favorite vice, police chiefs' daughters, and his favorite sport, van dodging. Because he's found that fresh air stimulates his creativity, clears his complexion and prevents nosebleeds, he now writes this column while hang-gliding over the Pacific Ocean.*

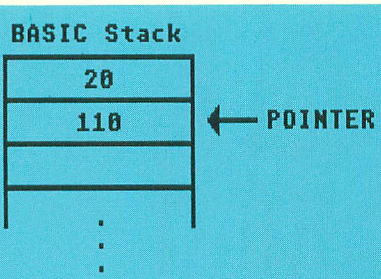
(CONTINUED FROM PAGE 119)

Line 100—This line executes a GOSUB to Line 200. The next statement after this GOSUB is Line 110, so this number is placed on the stack, and the stack pointer is advanced to the next available position. Execution continues at Line 200. The stack now looks like:



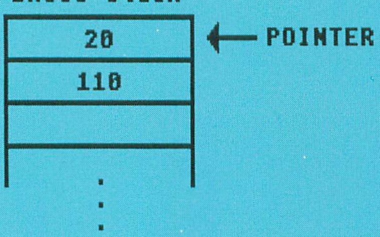
Line 200—The computer adds one to the variable A. The stack is not affected.

Line 210—The computer encounters a RETURN statement. At this point, the computer *increments* the stack pointer, like so:



Now the computer takes the line number 110 from the stack. As you can see, the computer can now go back to the instruction after the last GOSUB. Execution continues at Line 110.

Line 110—Another RETURN is encountered, and the stack pointer is incremented again. Now the stack looks like this:



The computer gets the line number from the stack and completes the RETURN by resuming execution at Line 20.

Line 20—This line terminates execution with the END statement. The stack is back to its original condition, with the pointer indicating the first stack location. The line numbers are still in the stack itself, but since the stack pointer no longer points to them, they are no longer active. They will be wiped out by new stack entries.

Now do you see how the stack works? It's a great way to handle subroutines, where the computer must be able to find its way back to the code which called up the subroutine.

Until Next Time

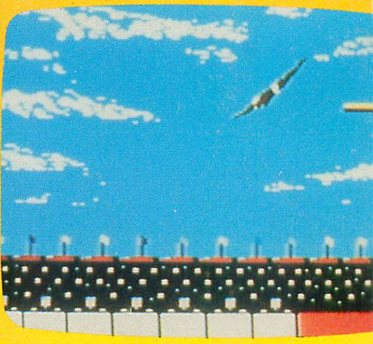
If you think *Boot Camp* looks more like *BASIC Training* this issue, hold on! I wanted to explain the subroutine process in a language you're familiar with, like BASIC. Next issue we'll examine the operation of the 6502 subroutine process and learn how to use the stack for our own programs.

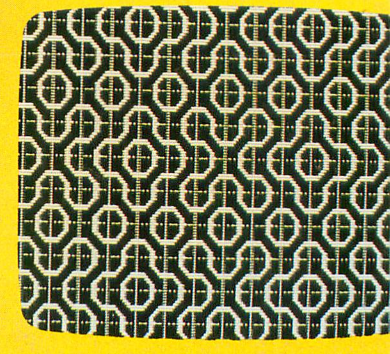
```
10 GOSUB 10
20 END
```

Until we meet again, the above is a little program to get you thinking. Type in the BASIC program and run it. It may take a while, but something will happen, and I want you to see if you can find the cause. Use the stack illustration method I used in the BASIC example to get the answer.

Also, if you haven't already, try to find more alternate methods for multiplying five by 27! ▣

iNSIDE this ISSUE





MORE

END USER • IAN'S QUEST • DATABASE DELPHI