

THE #1 MAGAZINE FOR ATARI COMPUTER OWNERS

ANALOG

COMPUTING

T.M.

FDC 50075

AUGUST 1988
ISSUE 63

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

P
LAD

HOPPING
MADNESS
TRAIN
CRAZY

Crisis Center

BCALC

A full-featured
spreadsheet

REVIEWS: *Shiloh* • *Bridge 5.0* • *Barnyard Blaster*



American Techna-Vision

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

"Providing 8 Bit support with one of the Worlds largest inventories of Atari replacement parts"

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

800.4 PIECE BOARD SET

Includes Main Board, Power Supply Assembly, CPU Module and 10K Revision B Operating System Module. All boards are new, tested and complete with all components.

\$28.50

1050 MECHANISM

Factory fresh TANDON mechs. make difficult repairs a snap. Units are complete with Head, Stepper, Spindle motor etc. Just plug in, no difficult alignments or adjustments required.

\$47.50

SUPRA

2400 BAUD MODEM
Fully Hayes Compatible. High quality. Works with all Atari's. 8 bit requires interface.

\$157.95

POWER PACKS

Exact replacement transformer for 800/400, 1050, 810, 1200XL, 850 and 1020 units. Replaces older "weaker" units. Atari part #CO17945.

\$14.50

400 3 PIECE BOARD SET

Includes Main Board, Power Supply Assembly and CPU Module. All boards are new, tested and complete with all components.

\$19.50

400 Membrane Keyboard \$12.50

810 DISK DRIVE PARTS

Sideboard with Data Sep. \$44.95
Analog Board \$19.50
Rear Power Board (New Style) . . . \$35.00
Rear/Analog Set (Upgrade) . . . \$47.50
Complete Set (New Style) . . . \$67.50
Board set includes instructions for using a Tandon TM-100-1 or MPI B-51 mechanics to build your own disk drive. (Mechanics NOT included or avail. from us)

ATARIWRITER CARTRIDGE

Popular cartridge version turns any 8 bit Atari into a powerful word processor. Written by Atari. Disk drive supported but not required.

For all Atari's except ST **\$29.95**

600XL 64K UPGRADE

Easy to install internal modification allows you to hook up a disk drive and run all 800XL software. Kit includes all parts and detailed instructions. Soldering required to install 3 jumpers.

\$29.95

800 10K "B" O.S. Module

Older 800 units need the revision "B" Operating system to run newer software. Type the following peek in BASIC to see which revision you have. PRINT PEEK(58383).

\$9.50

If the result is 56 order now!

PILOT PROGRAMMING LANGUAGE PACKAGE

Includes PILOT cart. with "Turtle Graphics", Pilot Primer and Student Pilot manuals. PILOT is an excellent learning or teaching tool.

Works with all Atari's except ST. **\$17.50**

PADDLE CONTROLLERS

(Pair). Required for numerous 8 bit programs and applications. Use these to add two changeable variables to your BASIC or machine language programs.

\$6.50

SERIAL I/O CABLE

High quality 13 pin cable used to connect 8 bit Atari's to disk drives, interfaces, etc.

New low price **\$5.95**

800/400 MODULES

NEW PARTS COMPLETE WITH IC'S

\$9.50 EACH

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

CX853 16K Ram Module \$14.50

800 Power Supply Board \$14.50

INTEGRATED CIRCUITS

\$4.50 EACH

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

1050 O.S. ROM \$13.50
2793 1050 FDC \$19.50
1050 5713 STEP DRIVER . . \$5.25
CO21697 - Use CO12296 - Except 130XE
810 O.S. ROM CO11299C . . \$10.00

REPAIR MANUALS

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

MISC. HARDWARE

1050 Track 0 Sensor . . . \$6.50
1030 Power Pack \$9.50
Fastchip for 800/400 . . . \$15.50
800 256K Upgrade w/o Ram \$89.95
Supra 2400 Baud Modem \$159.95
Atari Joystick \$7.00
850 or PR Modem Cable \$14.50
850 or PR Printer Cable \$12.50
P-R: Connection \$65.00
Printer Interface \$39.95
I/O 13 Pin PC mount . . . \$4.50
I/O 13 Pin Plug Kit . . . \$4.50
ST 6" Drive Cable . . . \$14.00
820 Printer Mechanics. . \$9.50
Joystick Extension Cable \$5.00
30 Pin Cartridge Socket . \$4.50
810 Door Latch Assy. . . \$15.00
Serial I/O Cable \$5.95
1027 Transformer CALL
U.S. Doubler \$29.95
ST Modem Cable \$14.50
ST Drive connector PC mount \$6.50
ST Drive connector plug \$6.50
25 Pin null modem connector \$10.95

COMPUTER BOOKS

Hackerbook \$5.00
Inside Atari Basic \$5.00
Atari Basic Ref. manual. . \$5.00
Basic-Faster & Better . . . \$22.95
Assembly Language Guide \$19.95
XE Users Handbook . . . \$17.95
XL Users Handbook . . . \$17.95
Advanced Programming \$19.50
Astrology with type in prog. \$5.00
Atari Games/type in prog. \$5.00
Forth on the Atari \$5.00
Mapping the Atari \$18.50

ATARI 850 INTERFACE

Bare PC Board with parts list and crystal allows you to build your own serial/parallel interface for attaching modems and printers to all 8 bit Atari computers. . . \$7.50
Board & all plug in IC's . . . \$39.50

ATARI XM301 MODEM

Direct connect 300 Baud modem works with all 8 bit Atari's. No separate interface required. \$44.95

BASIC CARTRIDGE

Basic Rev. "A" Cart. works with all Atari Computers except ST.
Exact replacement for 800/400/1200XL \$15.00

EPROM CARTRIDGES

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

SOFTWARE

Q*bert cartridge \$10.00
Choplifter Cart. \$10.00
Silicon Warrior Cart. . . \$12.50
Springer Cart. \$5.00
Fun with Art Cart. \$10.00
Donkey Kong Cart. \$5.00
Eastern Front Cart. . . . \$5.00
Star Raiders Cart. \$10.00
DropZone Disk \$9.95
Ed/Asm Cart. w/o man. \$10.00
Hard Hat Mack Disk . . . \$5.00
D-Bug childware Disk. \$5.00
Word Flyer childware. \$10.00
Home filing manager . . \$7.50
Musical Pilot Ed. Disk . . \$5.00
Big Math Attack Disk . . \$5.00
Chambers/Zorp Disk . . . \$5.00
Pathfinder Disk. \$5.00
Match Racer Disk \$5.00
Encounter/Quester Disk \$5.00
Baja Buggies Disk \$5.00
Com*putation Disk \$5.00
Cyborg Adventure Disk \$5.00
Fractions Tutorial Disk \$5.00
Decimals Tutorial Disk \$5.00

SERVICE RATES

Flat Service Rates below include Parts & Labor, 60 Day Warranty.
800 Computer \$39.50
850 Interface \$39.50
810 Disk Drive \$69.50
1050 Disk Drive \$75.00
800 Keyboard only. \$25.00
Include \$7.00 return shipping and insurance. Include \$4.00 shipping for 800 keyboard repair only.

CALL TOLL FREE

1-800-551-9995

IN CALIF. OR OUTSIDE U.S.

CALL 415-352-3787

AMERICAN TECHNA-VISION

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

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

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

Editorial

It has always been ANALOG Computing's policy to bring its readers the highest quality, type-in software found in any magazine. Frequently, however, commercial-quality programs are so huge that the printing of the listings in the magazine is prohibitive. Rather than resort to a "disk only" format for those programs, we sadly pass them by.

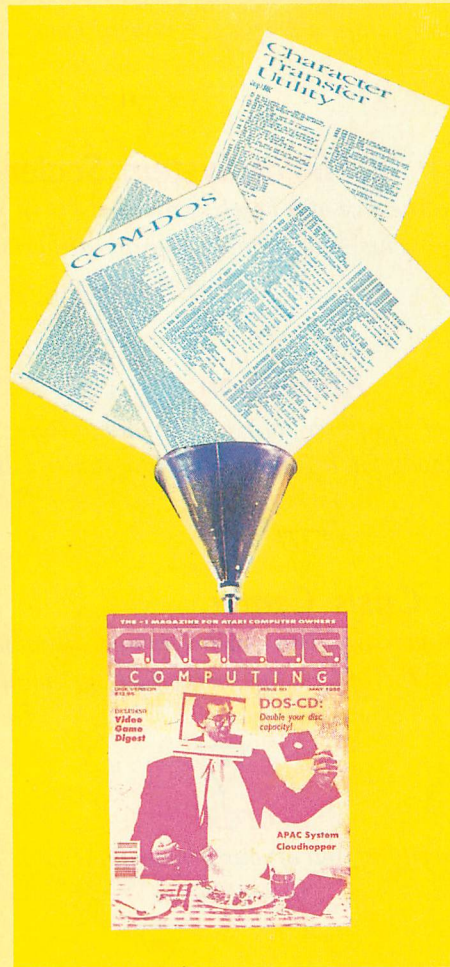
Once in a while, though, a program comes in that simply can't be ignored—even if it is on the large size—and we are faced with a decision: Do we offer the program over the course of two or three months, breaking it up into smaller pieces? Or do we dedicate a larger-than-normal portion of the magazine to the program, hoping that the majority of our readers will want to get the whole thing in one large article?

The last time this came up was when the program *Troll War II* came across my desk. In that instance, we chose to break the program up into two smaller parts, published in two succeeding months. Well, the letters came in, and the bulk of them requested that, the next time we had a program of this size to publish, we print it all in one month rather than make people wait for the conclusion. "We're not afraid of all that typing!" they insisted.

Also in those letters was another frequently asked question: "Why doesn't ANALOG Computing still run full-length assembly-language listings?" And to be honest, we really didn't have a good answer to that question. It seems that over the years we slowly got into the habit of not publishing those listings in order to find space for other material—and before we realized it, the full-length assembly listing had become a thing of the past.

In recent months we've tried to remedy that. We've made a greater effort to get our authors to supply nicely formatted assembly code for use in the magazine—and when they supply it, we print it. Looking back over the years, it strikes me as odd that we stopped publishing those listings regularly, especially considering that we've never had anyone complain about their inclusion in the magazine. *Au contraire!* It was this extra attention given to the advanced programmer that set ANA-

by Clayton Walnum



LOG Computing apart from its competition.

Of course, you've undoubtedly realized by now that there's a reason for this little chat we're having. To put it in a nutshell, it's happened again. A program has fallen into our hands that we absolutely cannot ignore.

By now, all ANALOG readers are familiar with the names Barry Kolbe and Bryan Schappel. In the past these two gentlemen have supplied some of the finest machine-language programs ever to appear in the pages of this magazine. Those programs include *BBK Artist*, *The*

Robox Incident, *TEDIT*, *The ANALOG Database* and *The Clash of Kings*, to mention only a few. This issue we're proud to present, complete with its assembly-language listings, *BCALC*, a full-featured spreadsheet for your 8-bit Atari computer—written, of course, by those prolific machine-language wizards, Kolbe and Schappel. There's no need now for you to run out and spend \$50 for that spreadsheet program you've been needing. *BCALC* will fit the bill quite nicely, thank you.

And once you've finished using *BCALC* to set up your home's or business' finances, don't forget to check out Colin Faller's zany *Train Crazy*, a truly arcade-quality game. And there's more! Joe McManus will get your nerves jangling with his fast-moving simulation, *Crisis Center*, where you get a chance to see how it feels to have hundreds of lives depending on your quick thinking and careful decision making. Carey Furlong brings us *Solar System Scaler*, an Atari BASIC program that'll bring the universe right into your living room, scaled down to a size that even the feeble (galactically speaking) human mind can understand.

Also in this issue, we start two new columns for those of you who want to learn more about your computers and how to program them. First on the agenda is Robin Sherer's *Master Memory Map*, a complete tour of your Atari's innards brought to you over the course of the next few months, including complete documentation of even the most esoteric memory locations in your machines. And for those of you who've always wanted to write your own arcade games, but didn't know where to start, we've got *Game Design Workshop*, a column by Craig Patchett that'll take you, month by month, through the entire process of designing and writing a video game.

We think this is an exciting issue, and we're sure you will too. As usual, we'd like to hear from you. Let us know how we're doing, and what we can do to better serve you. This is *your* magazine, and it's your input that'll keep it moving in the direction it should go.

And as always, thank you.

C O N T E

F E A T U R E S



10

Train Crazy

Join Oscar the Ostrich on the wild railroad romp on top of a speeding train. Look out for the tunnels!

by Colin Faller

12

Solar System Scaler

If you could reduce the earth to the size of a baseball and hold it in your hand, how far away would the sun be? How about the moon? This educational program answers these and many other questions.

by Carey M. Furlong

25

Animation

The art of computer animation made simple, including two methods for bringing your programs to life.

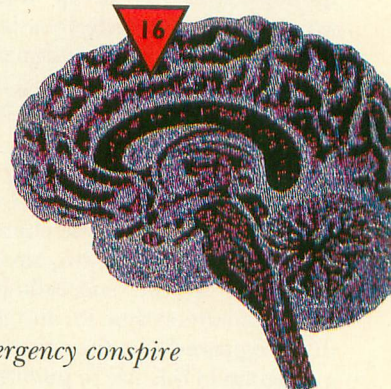
by Ron Goodman

34

Crisis Center

Car crashes, hotel fires, snipers, airplane hi-jackings, bank robberies and all manners of emergency conspire to put your abilities to the test in this nerve-racking simulation.

by Joe McManus



41

Wordlock

Having trouble keeping the riff-raff out of your private files? Lock 'em up tight with this machine-language utility.

by Andy A. Lee

46

PrintScreen

A Graphics O screen dump utility that'll print the contents of your screen from within most any program.

by Justin E. Wilder

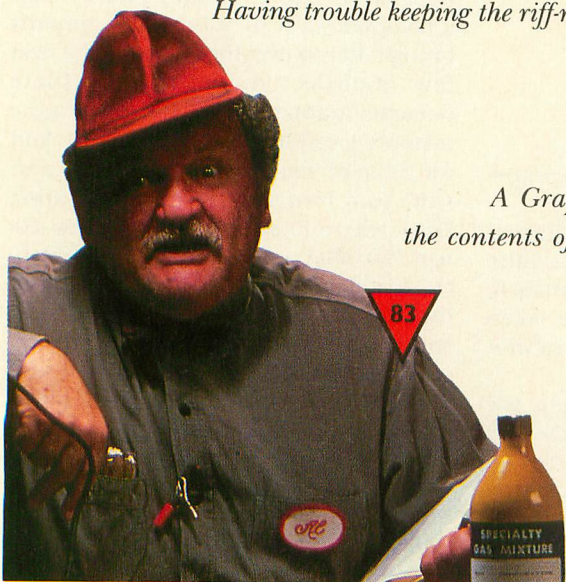
49

BCALC

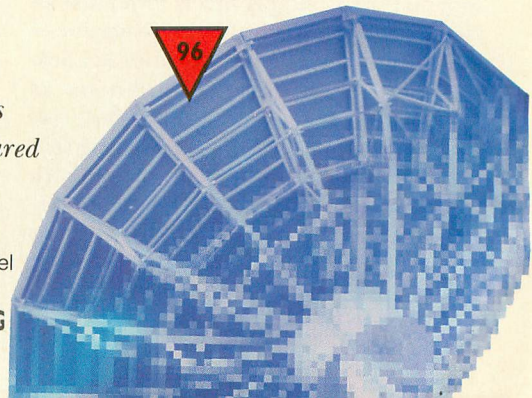
ANALOG Computing is proud to present a full-featured spreadsheet program for 8-bit Atari computers.

by Barry Kolbe and Bryan Schappel

96



83



T S

AUGUST 1988
ISSUE 63

R E V I E W S

9
Barnyard Blaster (Atari Corp.)

by Matthew J.W. Ratcliff

92
Panak Strikes

Steve gives the old thumbs-up/thumbs-down test to
Shiloh: Grant's Trial in the West (SSI)
and *Bridge 5.0*
(Artworx).
by Steve Panak

C O L U M N S

3
Editorial

by Clayton Walnum

6
8-Bit Notes

8
Reader Comment

16
Master Memory Map

by Robin Sherer

32
BASIC Editor II

by Clayton Walnum

40
ST Notes

83
Game Design Workshop

by Craig Patchett

96
Database Delphi

by Michael A. Banks

Front Cover Photography

Dean Brierly

S T A F F

Publisher
LEE H. PAPPAS

Executive Editor
CLAYTON WALNUM

Art Director
EDDY G. HERCH

Managing Editor
DEAN BRIERLY

East Coast Editor
ARTHUR LEYENBERGER

West Coast Editor
CHARLES F. JOHNSON

Contributing Editors
LEE S. BRILLIANT, M.D.,
MICHAEL BANKS, FRANK COHEN,
ANDY EDDY, MAURICE MOLYNEAUX,
STEVE PANAK, CRAIG PATCHETT,
MATTHEW J.W. RATCLIFF,
ROBIN SHERER, KARL E. WIEGERS

Entertainment Editors
ANDY EDDY, DAVID PLOTKIN

Illustrations
BRENT WATTS

Copy Chief
KATRINA VEIT

Copy Editors
ANNÉ DENBOK
SARA BELLUM, PAT ROMERO

Typographers
KLARISSA CURTIS,
JUDY VILLANUEVA,
DAVID BUCHANAN

Contributors
COLIN FALLER,
CAREY M. FURLONG,
RON GOODMAN,
BARRY KOLBE, ANDY A. LEE,
JOE MCMANUS, BRIAN SCHAPPEL,
JUSTIN E. WILDER

Production Director
DONNA HAHNER

National Advertising Director
JE PUBLISHERS REPRESENTATIVE
(213) 467-2266
(For regional numbers, see map)

Advertising Production Director
JANICE ROSENBLUM

Advertising Manager
PAULA THORNTON

Subscriptions Director
IRENE GRADSTEIN

Vice President, Sales
JAMES GUSTAFSON

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

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

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 \$7 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.

ANALOG Computing (ISSN 0744-9917) is published monthly by L.F.P., Inc., 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210. © 1988 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 \$7 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.

8-Bit News

The Newsroom is
entirely menu driven
and, as the box
indicates, well made for
"journalists of all ages."

Desktop Publishing for Your XE/XL

Before you spend \$4,000 on a Macintosh desktop publishing system, you might want to find out if desktop publishing is what you really need. For less than \$500 you could be writing your own newsletters using an Atari 130XE, Floppy Disk and *The Newsroom* from Springboard Software. This new package for the XE/XL lets you create banners, newspapers and fairly sophisticated layouts.

The Newsroom comes in a sturdy plastic box with two diskettes crammed with nifty drawings, diagrams and pictures. Additional *Clip Art Collection* disks are available for under \$20 to expand the library of drawings that can be used in your publications. *The Newsroom* is entirely menu driven, and as the box indicates is well made for "journalists of all ages."

Documentation for *The Newsroom* comes in a nicely illustrated 100-page users manual. Selections for printing text with serifs, sans serif, bold, underlined and more are easily chosen with a couple of key strokes. The program is well-designed for mixing text and graphics.

Springboard Software, Inc.
7808 CreekrIDGE Circle
Minneapolis, MN 55435
(612) 944-3912

AtariWriter 80 Rounds the Finish Line

The official line from Atari regarding the release date of *AtariWriter 80*, the new Atari word processor that works with the XEP80, 80 column-board, has been...

"AtariWriter 80 will be out next month!"

ANALOG contacted Microfantasy, the development group that wrote the original *AtariWriter Plus* and is now developing *AtariWriter 80*. Ron Rosen, the *AtariWriter 80* product manager, told us the *AtariWriter 80* program was completed in November 1987 and delivered to Atari for testing. After several debugging rounds, the program was finally accepted in April 1988 and is now being prepared for release.

AtariWriter 80 is an unusually powerful word processor. Using the XEP80 board, *AtariWriter* users have access to a professional word processor that offers print-preview, multiple margins settings, tab settings, cut, copy and paste functions and an advanced mail-merge database, proofreader and spelling checker for approximately \$100. The XEP80 board also gives immediate access to a parallel printer port, avoiding the use of the hard-

to-find Atari 850 interface board. On the rear of the XEP80 board, a DB25 parallel printer connector lets you use a standard printer connector to hook up your parallel printer.

Atari Corp.
1196 Borregas Avenue
Sunnyvale, CA 94086

Speed Talking

Years ago, Atari announced a 1200 baud, Hayes compatible modem that would be under \$100 with a SIO port connector for XE and XL users, and now the SX212 is finally out. ANALOG has been using the SX212 with a 130XE and 1040ST for the past month to communicate with Delphi, and has found it to be a very usable modem. The SX212 has jacks on the back for hookup to both the standard Atari SIO port and an RS-232 (DB22) connector for everyone else.

All of the modem's commands are set by typing on your XE/XL with a terminal program, while a real Hayes modem uses DIP switches inside the modem to set user options. The Atari modem is much easier to use. Inside the modem is a speaker, so you can hear the modem dial, and possibly receive a busy signal.

You will need a terminal program to use the SX212. *Express!* is a popular modem program that has been modified to work properly with the SX212. Another program, *R-Verter* is available from Atari dealers to allow your 850 interface to work with the SX212.

The SX212 has a suggested retail price of \$119.95 and is available now.

Atari Corp.
1196 Borregas Avenue
Sunnyvale, CA 94086

Farmer's Almanac

CompuServe has released the third edition of their Almanac for online services. The Almanac contains concise descriptions of hundreds of helpful products and services on CompuServe. In addition to the ATARI8 forum which provides an immediate source of XE and XL information, utilities and programs, there are forums for home banking, electronic mail, news and other information. If you have ever tried to navigate your way through CompuServe's hundreds of options, you will find the Almanac a carefully written assistant.

CompuServe
2180 Wilson Road
Columbus, OH 43228

SOUTHERN SOFTWARE

205-956-0986

24 HOUR PHONE

CALL OR WRITE FOR FREE CATALOG
ALL 8 BIT AND ST SOFTWARE IN STOCK
CALL FOR PRICES AND LATEST TITLES
SOFTWARE IN STOCK FOR OTHER COMPUTERS
PRICES LISTED FOR MAIL ORDER ONLY - ADD 10% ON PHONE ORDERS

ST COMPUTERS	CALL	29.95	80 COL CARD	79.95	
1050 DISK DRIVE	239.95	ATARI WRITER PLUS	39.95	COMPULSIVE COPIER	29.95
1050 W/HAPPY INST	399.95	BASIC CARTRIDGE	19.95	COPY II ST	29.95
130 KE	139.95	BASIC XE	49.95	AM-301 MODEM	49.95
XF551 DISK DRIVE	189.95	BASIC XL	39.95	SX-212 MODEM	49.95
520 DISK DRIVE	CALL	SUPER ARCHIVER	59.95	SM804 PRINTER	189.95
850 INTERFACE	109.95	BIT WRITER	69.95	1027 PRINTER	124.95
8K 0MINIMUM ACCESSORIES	59.95	DOS 2.5 W/MANUAL	9.95	ADM121 LQ PRINTER	199.95
APE FACE	49.95	XL/XE POWER SUPPLY	27.95	ATARI LAB ST KIT	49.95
AVATEX 1200	79.95	1027 INK ROLLER	4.97	JOY STICK	8.95
AVATEX 2400	179.95	HAPPY ARCHIVER	34.95	LOGIKHORN CLOCK	39.95
CHIP/ARCHIVER 810	69.95	I/O CARD	13.95	MARK WILLIAMS C	124.95
PRO BURNER	179.95	I/O CARD 10 FT	19.95	MEGAMAX C	139.95
EZ RAM 520	169.95	HAC/65	64.95	MODULA-2	69.97
HAPPY 1050 ENHANC	119.95	HAC/65 TOOL KIT	27.95	P.R. CONNECTION	59.95
HAPPY 810 ENHANCE	104.95	ACTION	64.95	WARP SPEED DOS XL	24.95
HAPPY CONTROLLER	39.95	ACTION TOOL KIT	27.95	PC BOARD DESIGNER	169.95
HARD DISK DRIVES	CALL	BASIC XL TOOL KIT	27.95	PROLOG	69.95
ICD M10 1 MEG	299.95	BASIC XE TOOL KIT	27.95	PUBLISH PARTNER	119.95
ICD M10 256K	199.95	PERSONAL PASCAL	74.95	SOUND DIGITIZER	119.95
XE ADAPTER FOR M10	19.95	SPARTADOS TOOL KIT	24.95	* WE HAVE COLOR RIBBONS *	
POWER SUPPLIES	CALL	ST HOST ADAPTER	99.95	FOR ALL PRINTERS	CALL
PRINTERS	CALL	SPARTS DOS CART	69.95		
PRINTER CONNECT.	39.95	OMNIVIEW XL/XE	36.95	* ATARI REPAIR PRICES *	
R-TIME CARTRIDGE	49.95	HEWELL 256K	34.95	ITEMS NOT LISTED	CALL
RAMBO XL UPGRADE	29.95	0MINIMUM 400/800	44.95	1050 DISK DRIVE	89.95
256K CHIP SET	49.95	ST COPY	29.95	130 XE	69.95
RAMROD XL	39.95	NUMERIC KEYPAD	39.95	65 XE	49.95
LIGHT PEN	69.95	TOP DOS 1.5 PLUS	29.95	520 DISK DRIVE	89.95
U.S. DOUBLER	29.95	PRINT/MODEM CABLE	13.95	520 ST	139.95
UPRINT INTERFACE	59.95	RAMCHARGER	139.95	850 INTERFACE	49.95
XL/XE B05	49.95	RAMCARD FOR 800	129.95	ATARI PRINTER	69.95
		SMART LINK MODEM	189.95	XL/XE/150 POW SUP	12.95

PRICES SUBJECT TO CHANGE WITHOUT NOTICE
ADD \$5 FOR SHIPPING AND INSURANCE. MOST ORDERS SHIPPED SAME DAY. FOREIGN ORDERS WELCOME WITH SUFFICIENT POSTAGE INCLUDED. ALABAMA RESIDENTS ADD 7% SALES TAX. ADD 6% FOR VISA. ADD \$5 FOR AIRMAIL. ADD \$15 FOR OVERNIGHT SHIPMENT. ALLOW THREE WEEKS FOR PERSONAL CHECKS.

SOUTHERN SOFTWARE

1879 RUFFNER ROAD BIRMINGHAM, AL 35210

CIRCLE #102 ON READER SERVICE CARD.

MegaByte

Computers and Electronics

Call TOLL FREE 1-800-255-5786 ORDERS

For any inquiries 1(713)338-2231 ONLY

109 W. Bay Area Webster, Texas - Please Call For Low Prices on Many Other Items!!
77598 Printer-Modem-Computer-Software

GAMES DARK CASTLE 27.97 DUNGEON MASTER (IN STOCK) 27.97 F-15 STRIKE EAGLE 24.87 FLIGHT SIM II 33.87 GAUNTLET 28.97 GONE FISHIN 27.97 MOEBIUS 38.97 OIDS 23.97 Police Quest 29.97 President Elect 28.97 RoadWar Europa 25.87 Ultima IV 38.97 Wargame Const. Set 33.87	Applications Cad-3D 2.02 59.87 Cyber Control 43.97 Cyber Paint 47.97 Data Manager Swift Calc 39.95 Word Writer 32.97 PC-Ditto 3.01 74.95 Publishing Partner 62.97 SuperCharged EasyDraw 94.97	GradePlus (A complete teachers gradebook system) \$39.95
	Kuma Transputers Now Only \$995 and up!!	Turbo ST ENHANCEMENT SUPERCHARGE Your ST to 16 MHZ!!! \$249.95
		CUSTOMIZED Hard Disks \$499 and up... EXTERNAL Floppies \$199.95 1

ORDERING INFORMATION:

Prices are subject to change without notice. All goods include FULL factory warranty. Texas residents please add 8.25% sales tax. Returns only if product is defective. Prices reflect a cash discount, please add 3.99 for Visa or MasterCard. Blue label shipping is available upon request.

Dealer Inquiries Welcome!

CIRCLE #103 ON READER SERVICE CARD.



COMPUTER SOFTWARE SERVICES

P.O. BOX 17660, ROCHESTER, N.Y. 14617

ATARI PHONE (716) 467-9326

JUST RELEASED!

\$99.95 "SUPER ARCHIVER II"!

(for ATARI 1050 drives)

NOW! COPS all ENHANCED DENSITY programs plus retains all of the features of our World Famous SUPER ARCHIVER! (see below). Allows you to COPY or CREATE single or ENHANCED density protection schemes (including PHANTOM SECTORS!). Completely automatic; compatible with the BIT-WRITER!; the ULTIMATE BACKUP/PROGRAMMING device! Only \$99.95 plus \$4 S/H/I. NOTICE! If you already own a SUPER ARCHIVER!, you can upgrade to a SUPER ARCHIVER II for only \$29.95 plus \$4 S/H/I (disk only - no additional hardware required!).

THE "SUPER ARCHIVER"![®] \$69.95 (for ATARI 1050 drives)

The new SUPER ARCHIVER, obsoletes all copying devices currently available for the ATARI 1050! It eliminates the need for Patches, PDB files, Computer Hardware, etc. Copies are exact duplicates of originals and will run on any drive; without exaggeration, the SUPER ARCHIVER is the most powerful PROGRAMMING/COPYING device available for the 1050! Installation consists of a plug-in chip and 6 simple solder connections. Softwares included. Features are:

- TRUE DOUBLE DENSITY
- ULTRA-SPEED read/write
- FULLY AUTOMATIC COPYING
- SUPPORTS EXTRA MEMORY
- SCREEN DUMP to printer
- TOGGLE HEX/DEC DISPLAY
- SECTOR or TRACK TRACING
- AUTOMATIC DIAGNOSTICS
- DISPLAYS HIDDEN PROTECTION
- ADJUSTABLE/CUSTOM SKEWING
- AUTOMATIC SPEED COMPENSATION
- AUTOMATIC/PROGRAMMABLE PHANTOM SECTOR MAKER
- ARCHIVER/HAPPY ARCHIVER COMPATIBLE
- BUILT-IN EDITOR-reads, writes, displays upto 35 sectors/track (short)
- BUILT-IN CUSTOM FORMATTER - upto 40 sectors/track
- BUILT-IN DISASSEMBLER
- BUILT-IN MAPPER - upto 42 sectors/track
- DISPLAYS/COPIES Double Density HEADERS
- AUTOMATIC FORMAT LENGTH CORRECTION
- SIMPLE INSTALLATION

The SUPER ARCHIVER is so POWERFUL that the only programs we know that can't be copied are the newer ELECTRONIC ARTS and SYNFILE/SYNCALC (34 FULL sectors/track). If you want it ALL... buy the "BIT-WRITER"! also... then you'll be able to copy even these programs! Only \$69.95 plus \$4 S/H/I.

DEALER/DISTRIBUTOR/USER GROUP Discounts available call for info. PHONE Orders - MASTER CARD, VISA MAIL - Money Orders, Checks.



\$79.95

THE SUPER ARCHIVER "BIT-WRITER"!

\$79.95

The Super Archiver "BIT-WRITER" is capable of duplicating even the "uncopyable" EA and SYN series which employ 34 FULL sectors/tracks. "BIT-WRITER" is capable of reproducing these and FUTURE protection schemes of non physically damaged disks. PLUG-IN circuit boards and 4 simple solder connections. The SUPER ARCHIVER with "BIT-WRITER" is the ultimate PROGRAMMING/COPYING device for Atari 1050's EXACT DUPLICATES of originals are made! Copies run on ANY drive. Must be used with Super Archiver. Only \$79.95 plus \$4 S/H/I.

\$69.95

"ULTRA SPEED PLUS"

\$69.95

Imagine a universal XL/XE Operating System so easy to use that anyone can operate it instantly, yet so versatile and powerful that every Hacker, Programmer and Ramdisk owner will wonder how they ever got along without it! Ultra Speed Plus puts unbelievable speed and convenience at your fingertips. Use ANY DOS to place an ULTRA SPEED format on your disks, boot any drive (1-9) upon power-up, format your RAMDISK in Double Density, activate a built-in 400/800 OS for software compatibility, plus dozens of other features to numerous to mention! Below are just a FEW features you'll find in the amazing OS:

- ULTRA Speed S10 for most modified drives
- ULTRA Speed is toggleable
- Boot directly from RAMDISK
- Special timer circuits not required for 1 or 2 Meg upgrades
- Background colors adjustable
- Reverse use of OPTION key
- Cold-start without memory loss
- Built in floppy disk configuration editor (1-9)
- Built in RAMDISK configuration editor (1-9)
- RAMDISK exactly duplicates floppy drive so sector copying and sector editing are now possible
- Built in MINI Sector Copier
- Toggle SCREEN OFF for up to 40% increase of processing speed
- Toggle internal BASIC
- Rom resident disk loader program (MACH 40 menu)
- DOUBLE DENSITY RAMDISK capable
- Entire MEMORY test that pinpoints defective RAM chip
- Boot any drive (1-9) upon power-up or cold-start
- Supports memory upgrades up to TWO MEGABYTES
- THREE Operating Systems in one (XL/XE, 400/800, ULTRA SPEED PLUS)

\$29.95

"XF551 ENHANCER"!

\$29.95

The XF551 Atari drive is a fine product with one major flaw... it writes to side TWO of your floppy disks BACKWARDS. This causes read/write incompatibility problems with all other single sided drives made for Atari such as Indus, Trak, Rana, Percom, Astra, Atari 1050, Atari 810, etc. Add the XF551 ENHANCER to the new XF551 drive and your problems are over! This device will restore 100% compatibility while retaining original design qualities of Atari's super new drive. The XF551 ENHANCER is a MUST for all XF551 Owners. Installation is simple. Only \$29.95 plus \$4 S/H/I.

CIRCLE #104 ON READER SERVICE CARD.

Reader Comment

The Missing Touch

I just received the May issue of ANALOG Computing, and I love the new cover design. The new logo is really sharp and the photo is hilarious. I couldn't wait to flip open the magazine to see what other changes had been made. Imagine my confusion when I was greeted by the same old ANALOG. Now don't get me wrong. I don't mean to say there was anything very wrong with ANALOG's old style. It's just that after reading the editorial in the April issue I was expecting some exciting new changes. So where are they? Did somebody quit right after coming up with the cover? Is this a case of beauty being only cover deep? Are you getting sick of all these silly questions?

—Todd Rapherty
Dormont, PA

Silly questions? What silly questions? We can understand your confusion, and so are glad to take this space to tell you—and everyone else who is interested—something about the magazine's production.

It takes a lot of time to design a magazine from the ground up. When ANALOG Computing went back into full production, there was enough work to keep us busy just getting the magazines out the door on time, without having to worry about a new design as well. To expedite the process of getting the magazines back into your mailbox, we used the material that had already been completed at the old offices. That's why the style looks so familiar. The only thing that was missing for those first issues (April and May) was the cover for May. Because we had to come up with a new cover anyway, we took the time to have it redesigned, taking us one step closer to that new look we promised you.

Kangaroos and Computing

I have never seen my kids so captivated as they are by the game Money Pouch in the May issue of ANALOG Computing. They play it for hours at a time. It has gotten to the point where I have to fight to get some time on the computer myself! It's so rare to find an educational program that can teach while still allowing the children to have fun. I've always been an advocate of painless learning

As you can tell by
flipping through this
magazine, ANALOG
has indeed been
redesigned. Over the
next few months, as we
get comfortable with
the new publishers and
they with us, the rough
edges will be smoothed.

(maybe I'm just basically lazy), and as a teacher, I've always tried to keep the interest level of my classes as high as possible. Money Pouch really hits the mark as far as I'm concerned, sneaking the learning in underneath the fun. Thanks for a great magazine! —Louis Patten
Ridgewood, NJ

Our pleasure. Maybe it's time for you to consider a second computer, so the kids can have one of their own. Most families who own computers find that one just isn't enough—and the older your children get, the more that'll be true.

Keyed for Help

I just bought an Atari 130XE computer, and there's something I just can't figure

out. One of the keys across the top of the computer is labeled HELP. Yet, whenever I push it, it does absolutely nothing. It doesn't matter what program I'm running or what I'm doing, the help key does nothing. All it adds to the computer is an extra key. Is there something wrong with my 130XE?

—Deron Smith
Russel, KS

No, there's probably nothing wrong with your computer. The HELP key on the 130XE must be accessed from within a program to make it useful—just like the START, SELECT and OPTION keys. If the programs you're using don't take advantage of it, the HELP key won't do anything. Unfortunately, very few companies have decided to incorporate the use of the HELP key into their programs; probably because the older Atari computers don't include the key. That really isn't so much of an issue now, since very few people still use the old Atari 400s and 800s. But when most of the programs that are available today for the Atari were being developed, using the HELP key would have made the program somewhat incompatible with the older machines. There are ways around that incompatibility, of course, but it seems that most developers took the easy way out and just ignored it.

Seeing is Believing

I can't believe it! The other day I actually saw a copy of ANALOG Computing in a supermarket drugstore! When you folks said distribution under the new owners would improve, you weren't kidding. I've always had a hard time finding the magazine, and had to keep checking the local B. Dalton's to get a copy before they were sold out. It's going to be great to be able to buy my favorite Atari magazine so close to home. By the way, they also had ST-Log. —

Richard Hall
Watertown, NY

Yep, we're very serious about improving our distribution, as well as improving subscription and customer support. As time marches on ANALOG Computing will be available in more and more locations across the country.

One quick question: Why didn't you just subscribe to the magazine instead of trying to hunt it down every month?

BARNYARD BLASTER

Atari Corp.

P.O. Box 61657

Sunnyvale, CA 94088

Game Cartridge—for all Atari 8-bit computers equipped with the XG-1 light gun

\$29.95

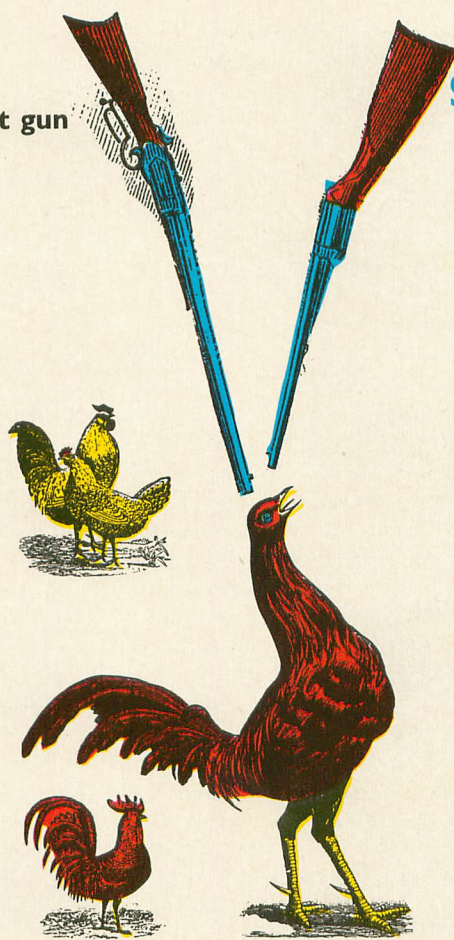
by Matthew J.W. Ratcliff

You and Gramps have just purchased a small farm that is being overrun by varmints. With your trusty XG-1 Atari light gun (similar to a .357 Magnum, but with a little less kick), you must blast all the critters in sight.

Grandpa first sets up some bottles and cans on the back fence to get you warmed up. As they are shot, the bottles explode in a shower of glass, while the cans ricochet high into the air—both excellent effects. Keep a sharp eye out for the gopher. He can pop up in the background at the end of any shooting round. He's been tearing up Grandpa's property; a tough critter to zap for bonus points.

At the end of each round, if your shooting percentage is high enough, you proceed to the bonus round. Here Grandpa is brave (stupid?) enough to toss spinning bottles into the air, for additional shooting practice. You score the most for each bottle (100 points) by exploding them over Grandpa's head. Don't accidentally hit him or you'll lose all your bonus points.

Out in the cornfield—the next round—are some watermelons and pumpkins that you must blast. Grandpa doesn't even mind that you shoot them! Once they're splatted all over the field, you must fend off blackbirds and fluffy white rabbits. A good shootist will get to



shoot bottles over Gramps' head and then move on to the barn scene.

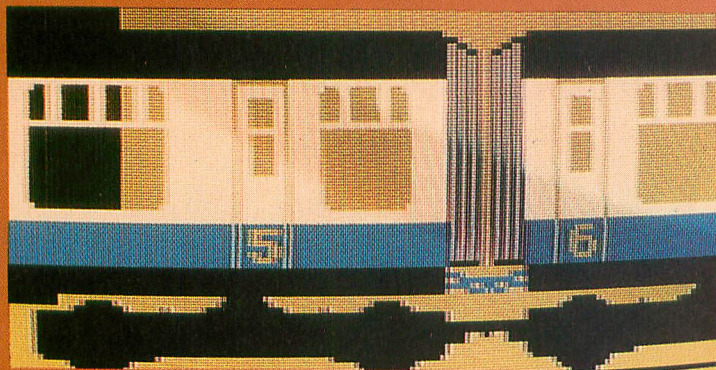
There are a whole mess of mice and sparrows loose in the barn. Those little critters are tough to zap, being small and fast. You'll have to contend with more rabbits and hoot owls too. Apparently Grandpa isn't much interested in his farmyard animals, since he doesn't mind if you blast the chickens and geese that run through the barn as well.

Barnyard Blaster presents these three

I love the
sound effects—especially
for splatting the
melons.

scenes—the barnyard fence, cornfield, and barn—alternating with Grandpa's bonus round. If you're a good sniper, you may make it through all 36 screens and finish the game. As you progress through the levels, you have fewer bullets per round and must maintain a progressively higher shooting accuracy. At the end of the game you're given a final rating from "Total Dud" through "Terminator" to the ultimate *Blaster*.

I have found *Barnyard Blaster* to be far more sophisticated and entertaining than the *Bug Hunt* game which comes with the Atari XEGS. The screens are bright and colorful, with well-executed graphics. I love the sound effects too, especially for splatting the melons. The author, Jim Zalewski, has paid great attention to detail. On any of the game screens, take aim at objects that you're *not* intended to shoot, blast and listen. Your bullets will ping off the weather vane or the tin washtub inside the barn. They will splash in the water bucket or thud into the scarecrow out in the cornfield. If you're looking for a good light-gun game for your system, *Barnyard Blaster* is the one. It's the only one actually, but it is quite good. I think we can expect more great light-gun applications from Jim Zalewski and K-Byte, the people who brought us such classics as *K-Razy Shootout*. **A**



by Colin Faller

W

elcome to *Train Crazy*, a game that will have you hopping mad. Your character is Oscar the Ostrich. Oscar is not only crazy, but *train* crazy, as you will soon find out. Oscar prefers the fresh air and likes to travel on top of the train. No problem, of course, until the train comes to a tunnel and Oscar has to run for his life! What's worse is that the London railway company doesn't like birds traveling on top of their trains, and so have hung several chains from the overhead wires to deter the "roof riders." They have also hung up some tempting diamonds in the hopes that Oscar will jump up and crack his head. But Oscar is smarter than that and can often grab the diamonds for himself.

To help Oscar survive you must duck under the chains and jump up to collect the diamonds, as well as jump between carriages and avoid the tunnel. The higher the level, the more obstacles there are. You will find it harder to get the diamonds as you progress. At the end of each level, Oscar will jump off the train, and your score will be displayed. You will then start the next level or, if you have lost a life, restart the existing level. Oscar has five lives each game.

There are four ways to lose a life: 1) by getting knocked off the train in the tunnel, 2) by hitting one of the obstacles, 3) by running out of energy and 4) by jumping too soon for the diamonds and knocking yourself out.

Typing it in

First type Listing 1, using Basic Editor II to check your typing, and then save the file to disk as TRAIN1.BAS. Now type Listing 2 (checking it with Basic Editor II, of course), and then save the file to

disk as TRAIN2.BAS.

Playing the game

The game is loaded in two parts. Part 1 loads the character set and scrolling routine and draws the train. It'll take about 33 seconds. When Part 1 is finished, it'll load Part 2 automatically.

After a few more seconds of initialization, you will hear the train whistle and be ready to play. Use a joystick in Port 1. Press either the fire button or the START key to begin.

Moving the joystick up will cause Oscar to jump up to catch the diamonds, jump over obstacles or jump from carriage to carriage. You must be careful not to jump too soon or you will hit the objects above the diamonds and lose a life. Moving the joystick down makes Oscar duck to avoid the hanging chains.

Each time you press the fire button, Oscar will move forward. If you take your finger off, he will move back with the train. If you don't keep him moving, he will go into the tunnel and lose a life.

When you first start the game, you are given 28 energy units, but these decrease as the game progresses. Each time you get a diamond, you get 200 points and an extra three energy units; so you must collect diamonds to complete the game. Each diamond collected will be displayed at the top of the screen.

When the game is over, you'll be rated according to your score as follows: TRY AGAIN — less than 4,000 points, GOOD — 4,000 to 9,999 points, GREAT — 10,000 to 14,999 points, EXCELLENT — over 15,000 points.

To play again press either the START key or SYSTEM RESET.

My highest score to date is 16,455 in ten levels. Can you do better?

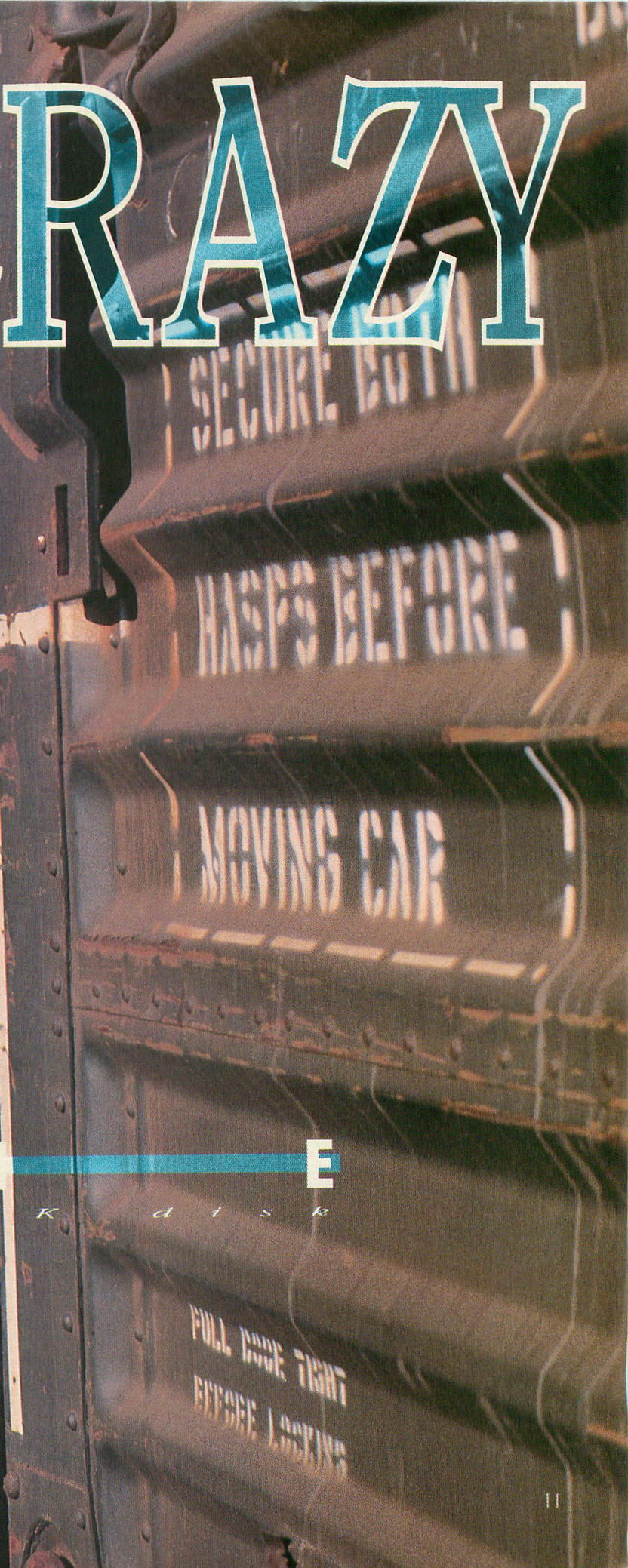
continued on page 74

TR

6

PHOTOGRAPHY BY DEAN BRIERLY

AIN CRAZY



A M E

A S K d i s k

FULL FACE MASK
BEFORE WORKING



Solar System
Sca



E D U C A T I O N A L

Have you ever been sitting in your living room watching the evening news and heard something like, "Voyager II just passed by the planet Uranus, almost two billion miles from Earth," and looked up at your 12-inch globe on the bookshelf and wondered just what that meant? Do you have a handle on what two billion miles from Earth really means?

ler

Or how about this: "Comet Iras-Aracki, a two-mile wide glacier of frozen ice-dust, will pass very close to the Earth tonight, only 400,000 miles away?" Did you know that this passage is comparable in size to a microbe passing 50 feet from a basketball (actually pretty close, astronomically speaking), and that on this scale, the moon is about the size of a baseball and is still 20 feet closer to the basketball than the microbe? Could you have gotten that from "only 400,000 miles away"?

Have you ever wondered about the relative sizes and distances of the sun and planets in the solar system and what the whole structure really looks like as a unit? Or have you ever tried to imagine a light-year? Most of us have heard that the nearest star, Alpha Centauri, is around 4.3 light-years away, which calculates to over 25 trillion miles. Now, unless you're really into this stuff, 25 trillion miles isn't going to mean much beyond a doggone incomprehensible whale of a long ways out there!

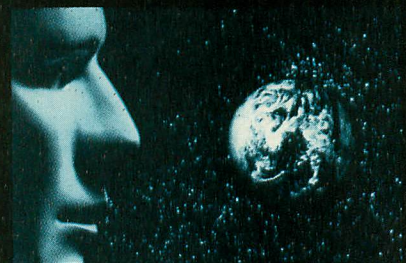
What I'm getting at through all this is that if you're like me, sizes and distances given in thousands, millions, billions and trillions of miles don't go very far in providing a palpable feel for the solar system or its relation to its nearest neighbor. But if you could turn a shrinking ray on the solar system and reduce it down to an understandable size (e.g., to a diameter of one mile, 100 feet or even the size of your living room), and then measure how far apart and how big everything is, you could gain a more realistic understanding of its true dimensions.

The menu-driven BASIC program

described in this article enables you to easily do just that. With it, mathematically scaled models of the solar system are constructed and tables generated that show the relative sizes and distances of the sun, planets and nearest star system. All values are determined by user defined constants input through prompts found on the menus. Units of Measure (UOM) are also determined by the user and can be millimeters, inches, feet, miles or anything you want to type in.

The program is based on ratio formulas and is simple in operation. Mathematical tables with size or distance

Now we know the diameter of the sun in feet when the Earth is 12 inches in diameter.



components for scaled models of the solar system are generated and printed to the screen or printer.

Type in the program listing (checking your work with Basic Editor II) and save a copy before running it. There are five options available from the main menu:

- "S" SUN DIAMETER BASED TABLES
- "E" EARTH DIAMETER BASED TABLES
- "SS" SOLAR SYSTEM DIAMETER BASED TABLES
- "12" 12-INCH EARTH GLOBE DISTANCE CONVERSIONS
- "999" END PROGRAM

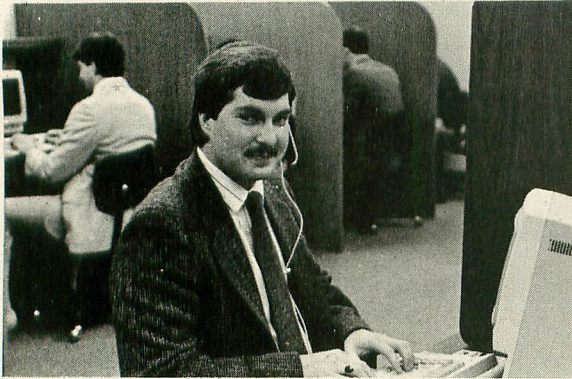
continued on page 78



Since 1981

Lyco Computer Marketing & Consultants

Lyco Means Total Service.



Mark "Mac" Bowser, Sales Manager

I would personally like to thank all of our past customers for helping to make Lyco Computer one of the largest mail order companies and a leader in the industry. Also, I would like to extend my personal invitation to all computer enthusiasts who have not experienced the services that we provide. Please call our trained sales staff at our toll-free number to inquire about our diverse product line and weekly specials.

First and foremost our philosophy is to keep abreast of the changing market so that we can provide you with not only factory-fresh merchandise but also the newest models offered by the manufacturers at the absolute best possible prices. We offer the widest selection of computer hardware, software and accessories.

Feel free to call Lyco if you want to know more about a particular item. I can't stress enough that our toll-free number is not just for orders. Many companies have a toll-free number for ordering, but if you just want to ask a question about a product, you have to make a toll call. Not at Lyco. Our trained sales staff is knowledgeable about all the products we stock and is happy to answer any questions you may have. We will do our best to make sure that the product you select will fit your application. We also have Saturday hours — one more reason to call us for all your computer needs.

Once you've placed your order with Lyco, we don't forget about you. Our friendly, professional customer service representatives will find answers to your questions about the status of an order, warranties, product availability, or prices.

Lyco Computer stocks a multimillion dollar inventory of factory-fresh merchandise. Chances are we have exactly what you want right in our warehouse. And that means you'll get it fast. In fact, orders are normally shipped within 24 hours. Free shipping on prepaid orders over \$50, and there is no deposit required on C.O.D. orders. Air freight or UPS Blue/Red Label shipping is available, too. And all products carry the full manufacturers' warranties.

I can't see why anyone would shop anywhere else. Selection from our huge in-stock inventory, best price, service that can't be beat—we've got it all here at Lyco Computer.

TO ORDER, CALL TOLL-FREE: 1-800-233-8760
New PA Wats: 1-800-233-8760
 Outside Continental US Call: 1-717-494-1030

Hours: 9AM to 8PM, Mon. - Thurs.
 9AM to 6PM, Friday — 10AM to 6PM, Saturday

For Customer Service, call 1-717-494-1670,
 9AM to 5PM, Mon. - Fri.

Or write: Lyco Computer, Inc.
 P.O. Box 5088, Jersey Shore, PA 17740



Risk-Free Policy: • full manufacturers' warranties • no sales tax outside PA
 • prices show 4% cash discount; add 4% for credit cards • APO, FPO, international: add \$5 plus 3% for priority • 4-week clearance on personal checks
 • we check for credit card theft • compatibility not guaranteed • return authorization required • price/availability subject to change • Prepaid orders under \$50 in con., U.S. add \$3.00.

Monitors

Thomson:

230 Amber TTL/12"	\$79.95
4120 CGA	\$219.95
4160 CGA	\$254.95
4460 EGA	\$349.95
GB 200 Super Card	\$184.95
4570	\$CALL

Magnavox:

BM7652	\$79.95
BM7622	\$79.95
7BM-613	\$74.95
7BM-623	\$79.95
CM8502	\$199.95
CM8505	\$209.95
CM8562	\$239.95
CM8762	\$239.95
BCM-515	\$249.95

Blue Chip:

BCM 12" Green TTL	\$64.95
BCM 12" Amber TTL	\$69.95

NEC:

Multisync II	\$599.95
--------------------	----------

ATI Technologies:

Graphics solution	\$129.95
EGA Wonder	\$199.95
VIP	\$299.95

Modems

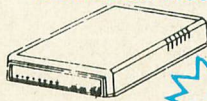
Avatec:

1200e	\$69.95
1200i PC Card	\$69.95
1200hc Modem	\$89.95
2400	\$179.95
2400i PC Card	\$169.95

Hayes:

Smartmodem 300	\$149.95
Smartmodem 1200	\$285.95
Smartmodem 2400	\$425.95

Smarteam 1200 Baud Modem



Hayes Compatible

OUR PRICE
\$89.95

ATARI

Access:

Triple Pack	\$11.95
Leader Board Pack	\$14.95

Activision:

Music Studio	\$19.95
Solid Gold Vol. #1	\$10.95

Batteries Included:

Paperclip 80 Col	\$31.95
------------------------	---------

Broderbund:

Print Shop	\$25.49
Print Shop Compan.	\$22.95
Graphic Lib. I, II, III	\$13.49
Bank St. Writer	\$27.95

Electronic Arts:

Pinball Con Set	\$8.95
Lords of Conquest	\$8.95
Starfleet I	\$32.95
Chess Master 2000	\$25.95
Music Con Set	\$8.95
Super Boulderdash	\$8.95
One on One	\$8.95

Firebird:

The Pawn	\$22.95
----------------	---------

Microleague:

Microleag. Baseball	\$22.95
General Manager	\$16.95
Stat Disk	\$13.95
87 Team Disk	\$13.95

Microprose:

Conflict in Vietnam	\$22.95
F-15 Strike Eagle	\$19.95
Kennedy Approach	\$13.95
Silent Service	\$19.95
Top Gunner	\$13.95

Strategic Simulations:

Battle of Antietam	\$28.95
Phantasia	\$22.95
Wargame Construc.	\$16.95
Wizards Crown	\$22.95
Phantasia II	\$22.95
Shiloh	\$22.95
Eternal Dagger	\$22.95

Sublogic:

Flight Simulator II	\$31.49
---------------------------	---------

ATARI ST

Microleague:

Microleague Baseball ..	\$33.95
General Manager	\$16.95
Wrestling	\$25.95

Microprose:

Silent Service	\$22.95
F-15 Strike Eagle	\$24.95
Gunship	\$28.95

Strategic Simulations:

Phantasia	\$22.95
Phantasia II	\$22.95
Road War 2000	\$22.95
Colonial Conquest	\$22.95

Sublogic:

Flight Simulator II	\$31.49
Scenery Disk	\$14.95

Timeworks:

Wordwriter ST	\$44.95
Partner ST	\$39.95
Data Manager ST	\$44.95

Unison World:

Art Gallery 1 or 2	\$14.95
Print Master	\$19.95
Fonts & Borders	\$17.95
Music Studio	\$27.95
Bureaucracy	\$22.95

Electronic Arts:

Arctic Fox	\$25.95
Empire	\$32.95
Starfleet I	\$32.95
Chess Master 2000	\$25.95
Gridiron	\$32.95

Epyx:

Sub Battle Simulator ...	\$22.95
World Games	\$22.95
Wrestling	\$22.95
Winter Games	\$11.95

Firebird:

Pawn	\$25.95
Starglider	\$25.95
Golden Path	\$25.95
Guild of Thieves	\$25.95
Tracker	\$25.95

ATARI ST

Joysticks

Access:

Leader Board	\$22.95
Tournament #1	\$11.95
10th Frame	\$22.95

Activision:

Champion. Baseball	\$22.95
Champion. Basketball ..	\$22.95
Championship Golf	\$New
GFL Football	\$22.95

Tac 3	\$9.95
Tac 2	\$10.95
Tac 5	\$12.95
Tac 1 + IBM/AP	\$26.95
Economy	\$5.95
Silk Stick	\$6.95
Black Max	\$10.95
Boss	\$11.99
3-Way	\$19.99

1-800-233-8760

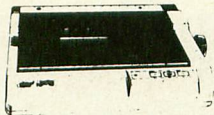
CIRCLE #105 ON READER SERVICE CARD.



NX-1000

- 144 cps Draft
- 36 cps NLQ
- EZ Operation Front Panel Control

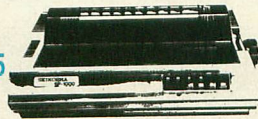
\$179⁹⁵



SEIKOSHA Sp 180Ai

- 100 cps draft
- 20 cps NLQ

\$129⁹⁵

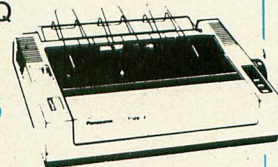


Panasonic

1091 Model II

- 192 cps Draft
- 32 cps NLQ

\$199⁹⁵



PRINTERS



NX-1000	\$179.95
NX-1000C	\$179.95
NX-1000 Color	\$225.95
NX-1000C Color	\$229.95
NX-15	\$309.95
NR-10	\$339.95
NR-15	\$439.95
NB-15 24 Pin	\$699.95
NB24-10 24 Pin	\$425.95
NB24-15 24 Pin	\$579.95
Laser 8	\$CALL

BROTHER

M1109	\$195.95
M1409	\$299.95
M1509	\$335.95
M1709	\$475.95
Twinwriter 6 Dot & Daisy	\$899.95
M1724L	\$599.95
HR20	\$339.95
HR40	\$569.95
HR60	\$709.95

SEIKOSHA

SP 180Ai	\$129.95
SP 180VC	\$129.95
SP 1000VC	\$139.95
SP 1000AP	\$169.95
SP 1200VC	\$155.95
SP 1200Ai	\$165.95
SP 1200AS RS232	\$165.95
SL 80Ai	\$299.95
MP1300Ai	\$269.95
MP5300Ai	\$399.95
MP5420FA	\$995.95
SP Series Ribbon	\$7.95
SK3000 Ai	\$339.95
SK3005 Ai	\$419.95
SPB 10	\$CALL
SL 130Ai	\$599.95

Toshiba

321SL	\$489.95
341 SL	\$659.95
P351 Model II	\$899.95
351 SX 400 cps	\$1019.95

EPSON®

LX800	\$199.95
FX86E	\$279.95
FX286E	\$424.95
EX800	\$399.95
LQ500	\$339.95
LQ1000 w/Tractor	\$549.95
LQ2500	\$819.95
GQ3500	\$LOW
LQ850	\$525.95
LQ1050	\$715.95



Okimate 20	\$119
Okimate 20 w/cart	\$179.95
120	\$189.95
180	\$219.95
182	\$209.95
182+	\$225.95
183	\$249.95
192+	\$339.95
193+	\$449.95
292 w/interface	\$449.95
293 w/interface	\$585.95
294 w/interface	\$819.95
393	\$955.95

Panasonic

1080i Model II	\$179.95
1091i Model II	\$199.95
1092i	\$319.95
1592	\$409.95
1595	\$459.95
3131	\$299.95
3151	\$479.95
KXP 4450 Laser	\$CALL
1524 24 Pin	\$559.95
Fax Partner	\$589.95

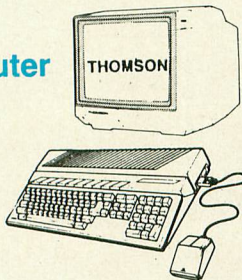


120 D	\$169.95
180 D	\$189.95
MSP-10	\$259.95
MSP-40	\$309.95
MSP-15	\$349.95
MSP-50	\$399.95
MSP-45	\$459.95
MSP-55	\$539.95
Premiere 35	\$499.95
Tribute 224	\$649.95
Tribute 124	\$469.95

ATARI® 520 ST Computer

- Built-in Drive
- Thomson 4120 Monitor

\$729⁹⁵



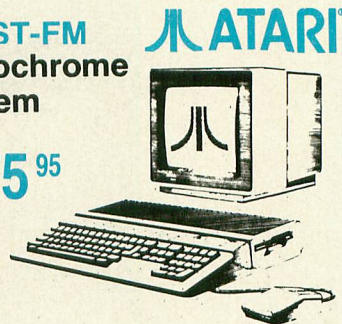
ATARI HARDWARE

520 ST FM Mono	\$675.95
520 ST FM Color	\$729.95
1040 ST Mono	\$739.95
1040 ST Color	\$955.95
130XE Computer	\$135.95
SX551 Drive	\$174.95
SF 314 Disk Drive	\$219.95
Indus GT Atari Drive	\$169.95
SHD 204 20 MEG Drive	\$579.95
XM301 Modem	\$42.95
SX212 Modem	\$89.95
GTS 100 (3.5" DSDD ST)	\$195.95
GTS 1000 5¼ DSDD ST	\$CALL

520 ST-FM Monochrome System

\$675⁹⁵

Internal drive included



ATARI® 1040 ST Color System

\$955⁹⁵



Attention Educational Institutions:

If you are not currently using our educational service program, please call our representatives for details.

ATARI® 1040 Monochrome System

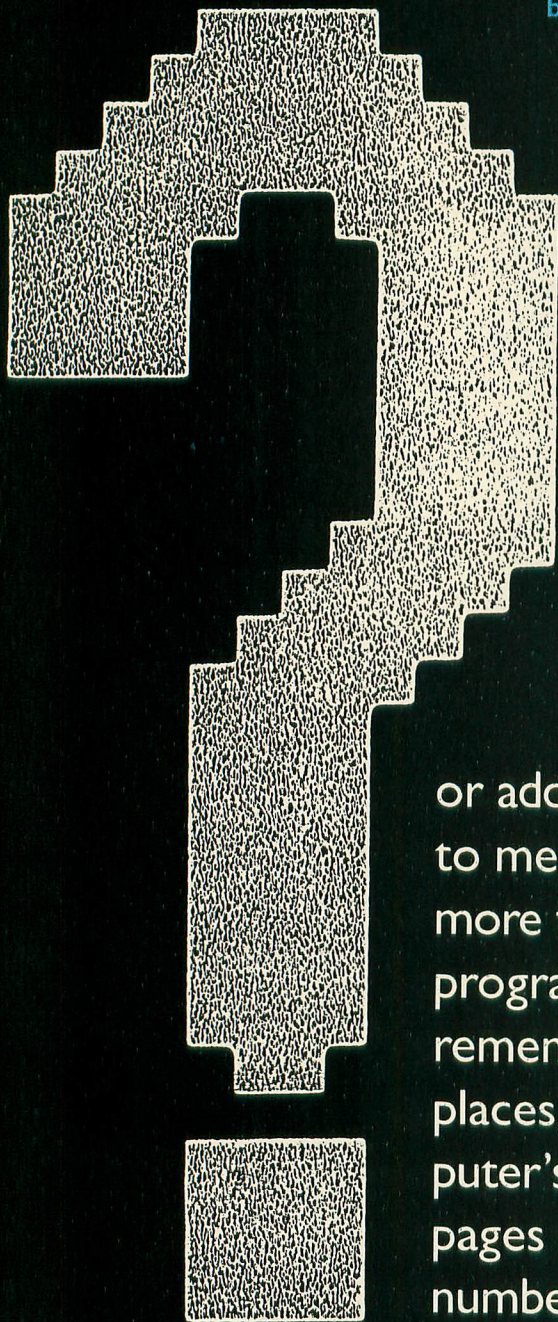
\$789⁹⁵



Master Me

C O L U M N

by Robin Sherer

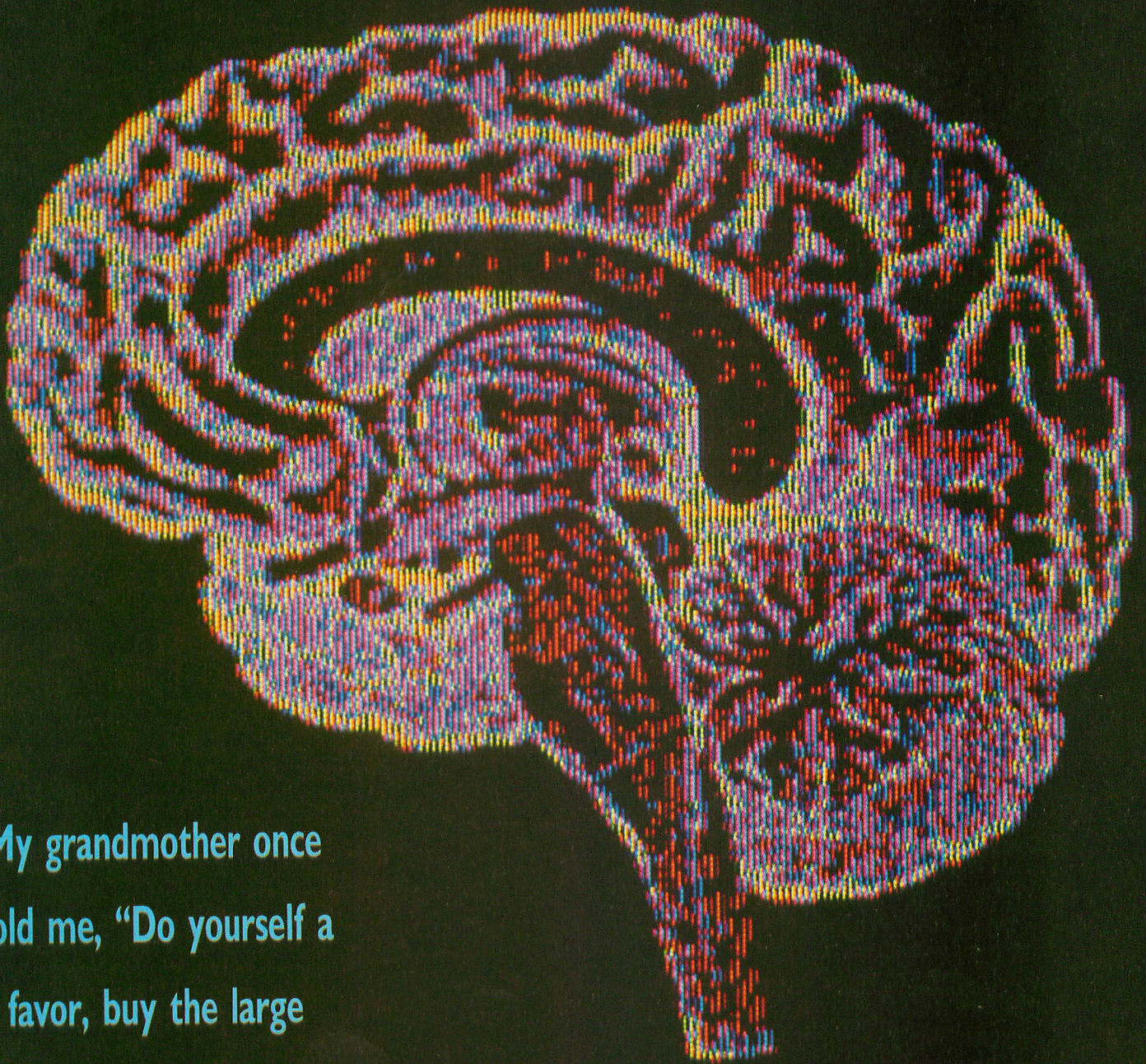


What is a memory location?

Good question! Your Atari has many “places” within it that can contain numbers. These places we call memory cells, locations or addresses. You may use any of these words to mean the same thing, but “address” is the more formal term that you will hear computer programmers using. Since the computer can remember the numbers in each of these places, it is common to call them the computer’s “memory.” Memory is like many blank pages of paper. Each page can hold only 256 numbers, and we can have up to 256 of these pages.

So where does this leave us? Well, 256 locations per page times 256 pages gives us 65536 locations total that *can* be in your computer. Computers count in

mory Map

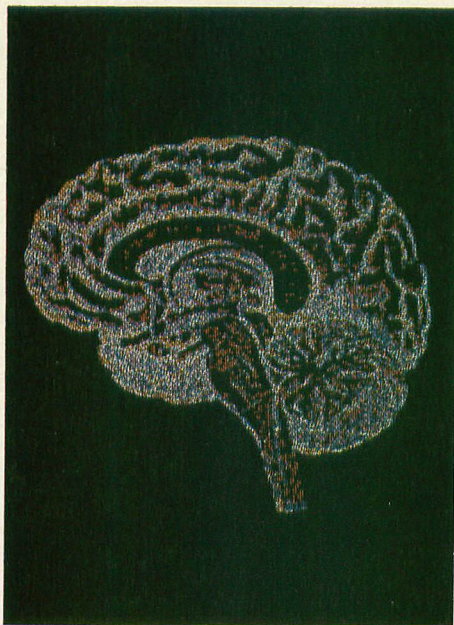


My grandmother once told me, "Do yourself a favor, buy the large economy size. You'll need it."

Master Memory Map

We encounter another small problem when we mess with memory.

Some memory addresses will allow us to read what is there, but will not allow us to write to them.



terms of something called a "K." For reasons beyond my control, one K of memory is actually 1024 locations. Why? I'll explain all in a few pages. For now divide 65536 by 1024 to get a possible memory size for the Atari of 64K.

The Atari is capable of using 64K of memory. (The XE computers use something called "Bank Switching" to get around this limit.) The factory gives you 16K ROM and 64K RAM when you buy the computer (we'll explain ROM and RAM later), but there is a catch (of course). A lot of memory that comes with the Atari is already filled up with numbers that tell the computer how to run. The blank area available for you to use is a lot less than you're led to expect. If you want more memory, you have to go down to your friendly dealer and buy it. My grandmother once told me, "Do yourself a favor, buy the large economy size. You'll need it." She was talking about soap, but her wisdom applies to computers as well.

Now that you understand how much memory you have, let's talk about it a little. Each memory address can hold numbers from 0 to 255. Computers start counting at 0 because "nothing" is a very valid piece of information. I certainly worry when my wallet has "0" in it! Let's learn how computers count.

Bits and bytes

A "byte" is really not complicated at all. It is simply a group of eight "bits." When eight bits are structured into a byte, each of those bits has special significance. You look puzzled! What, you say, is a bit?

A bit is the smallest piece of information that a computer can deal with. To help understand how bits are used by the computer, it may help to imagine the microprocessor as a bus station. This bus station is on a single-lane road. That means a bus can only travel in one direction at a time as there is not enough room for two buses to pass each other. Therefore, a bus may either be arriving at the station or departing. The microprocessor, or bus station, can schedule its bus with a signal light that says "I am accepting arrivals" or "I am sending departures."

In fact, in real computer hardware architecture, the wires that carry information to and from a microprocessor are called the "data bus." We don't need eight separate input and eight separate output wires because, like the single-lane road connected to the bus station, the wires are bi-directional. In other words, information can either be arriving (input) or

continued on page 20

NEW HACK PACK Special OFFER

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

Atari Software Protection Techniques Vol I & II

These Book and Disk packages detail the most advanced copy protection methods in use today. They guide you through the methods used to create the protection as well as the copying techniques to get around them. They include information on Phreaking • Hacking • On-line security • Black boxes • Self-destructing programs • Pirate bulletin board systems • Logic Bombs • New piracy laws • Hardware data keys • Weak sectoring (Phantom, Fuzzy and unstable sectors) • Overfilled tracks • CRC errors • Bank Select cartridges and MUCH, MUCH MORE. The disks include automatic program protectors, Protection Scanners, directory hiding and more.

BOOK I and DISK I \$24.95
BOOK II (Advanced protection) and DISK II \$24.95
Special Offer, Order both sets for Only \$39.95

CHIPMUNK

Automatic Disk Back-Up System. Make perfectly running unprotected back-up copies of hundreds of the most popular Atari programs. Chipmunk's sophisticated programming Automatically finds and **REMOVES copy protection** from most Atari programs. Back-up even heavily protected programs with ease. Finally, a back-up system that needs no special hardware or skills.

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

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

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

NEW CHEAT

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

BASIC TURBOCHARGER

NOW for the first time a BASIC programmer can get the power, flexibility and incredible speed of machine language. BASIC TURBOCHARGER is a **book and disk package** that contains over 150 ready to use machine language routines. Complete instructions show how to add them to your own BASIC programs to get these features and more: • Smooth Scrolling • Player/Missile control • Load & Save Picture files • Sorting and Searching • Special Effects Graphics • Incredible Speed • Much, Much More • Over 150 programs. You've heard of the power of Assembler, now harness it for your own needs. **\$24.95**



24 HOUR HOTLINE **216-374-7469**

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

ATARI 8-BIT POWER

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

NEW

PARROT II

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

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

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

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

NEW POP-N-ROCKER

a fast paced, multi-player trivia game that mixes questions with real songs (digitized)

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

COMPUTEREYES & MAGNIPRINT II+

Turn your computer into a digital portrait studio. This complete package lets you **capture, save & print** digital images from your **Video Camera, VCR or TV**. **COMPUTEREYES** hardware plugs directly into your joystick ports for easy use. Print your picture on a 6 foot poster. **\$119.95**

ComputerEyes camera system

Comes complete with everything above, plus a black and white video camera and connecting cable. **\$329.95**

Graphics 9 Software - Add a new dimension to your **COMPUTEREYES** pictures - captures images in 16 shades of grey. **\$12.00**

Magniprint II+

Easily the most powerful print program available today. Print graphics from almost any format in hundreds of shapes, sizes, and shades. Supports **color printing** and lets you create **giant posters**. **Magniprint II+** lets you stretch and squeeze, invert, add text, adjust shading and much more.

Works with EPSON, NEC, Citoh, Panasonic, Gemini, Star, XMM801, and compatible printers. (850 interface or equivalent required). **\$24.95**

Graphics Transformer

Now you can combine the most powerful features of all your graphics programs. Create print shop icons from a Koolaid pad picture, from a photo digitized with ComputerEyes, or any picture file. **Graphics Transformer** lets you **Shrink, Enlarge and Merge** pictures for unequaled flexibility. **\$22.95**

YOUR ATARI COMES ALIVE

SAVE MONEY. Finally an alternative to buying expensive computer add-ons. Your Atari Comes Alive shows you how to **build them yourself**. This "How-To" **book and disk package** gives you complete step by step instructions and programs needed to build and control these exciting devices and MORE:

• Light Pen • Light & Motor Controllers • Alarm Systems • Voice Recognition • Environmental Sensors • Data Decoders • More than 150 pages. **Your Atari Comes Alive** **\$24.95**



GIANT WALL SIZED POSTERS.

CIRCLE #106 ON READER SERVICE CARD.

B&C ComputerVisions

3257 Kifer Road
 Santa Clara, CA 95051
 (408) 749-1003



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

SUPER SPECIALS
RECONDITIONED ATARI MERCHANDISE

All merchandise has been tested and reconditioned and is in like-new condition except where noted by the letter "B" after the price. The "B" price indicates product may have scratches or other superficial surface marks. 30 day warranty.

<p>ATARI TRAK BALL \$9.95 A</p> <p>SPICE UP THE ACTION IN YOUR ARCADE GAMES!</p>	<p>ATARI SPACE AGE JOYSTICK \$5.00 A</p> <p>GUN TRIGGER ACTION!</p>	<p>STANDARD ATARI JOYSTICK \$4.50 A</p> <p>STOCK UP ON A FEW SPARES</p>	<p>1020 COLOR PLOTTER/ PRINTER \$29.95 A</p> <p>40 Columns wide Includes paper and color pen set</p>	<p>\$19.95 each (on cartridge)</p> <p>Ball Blazer Battle Zone Blue Max Fight Night Hardball Jungle Hunt Pole Position Galaxian Millipede Dig Dug Ms. Pac Man Final Legacy</p>
<p>400 (16K) COMPUTER \$29.95 A</p> <p>48K UPGRADE KIT \$25.00</p>	<p>600XL (16K) COMPUTER \$49.95 A</p> <p>64K UPGRADES \$19.95 - solder in \$34.95 - plug in</p>	<p>800 (48K) COMPUTER \$79.95 A</p> <p>INCL. BASIC CART & MANUAL</p>	<p>NUMERIC KEY PAD \$7.95 A</p> <p>INCL. HANDLER DISK USE WITH THE BOOKKEEPER AND BASIC</p>	<p>850 INTERFACE \$89.95 A</p> <p>LIMITED SUPPLY</p>
<p>1030 MODEM WITH EXPRESS \$29.95 A</p> <p>GET ON-LINE TODAY!</p>	<p>\$5.00 each (on cartridge) Basic Rev. A Donkey Kong Missile Command Pac Man Star Raiders Music Composer</p>	<p>810 DISK DRIVE \$120.00 A</p> <p>INCLUDES POWER SUPPLY, I/O CABLE & DOS 2</p>	<p>ATARI BOOKKEEPER \$14.95 - NO BOX (\$19.95 WITH RECON KEYPAD)</p> <p>\$24.95 - IN BOX (\$29.95 WITH RECON KEYPAD)</p>	<p>DISKETTES AS LOW AS 20 CENTS</p> <p>10 FOR \$4.00 100 FOR \$29.00 1000 FOR \$200</p> <p>MOST ARE UNNOTCHED WITH OLD SOFTWARE</p>

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. 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 complete line of ATARI products and have a large public domain library. Write or call for free catalogue. (408) 749-1003 TUE - FRI 10AM - 6 PM

CIRCLE #107 ON READER SERVICE CARD.

Master Memory Map

continued from page 18

departing (output), but not both. The microprocessor also has a signal of its own that determines whether it will receive (input) or send (output) information.

Let's take a closer look at that bus. It is known as the Byte Express, has eight seats and always carries eight passengers. Those passengers are little messengers known as bits, and, as a group, they are known as a byte. These messengers, or bits, are rather moody. They are either turned "on" or they are turned "off." That is called "binary" as they are bi-state signals; on being a "1" state and off being a "0" state. Their vocabulary is just as limited—the only thing they are willing to tell you is their mood. Now how do we get any meaningful information out of a group of eight little messengers standing in front of us, each screaming "on" or "off" at one time?

Well, when the bus arrives, we could have the whole byte stand in front of us and count everyone who is turned "on." That would give us the capability of counting to eight. Seems pretty limited, doesn't it? Hmmm, the group really needs a leader. That leader will be the first bit on the left. We'll call that the Most Significant Bit, or MSB. The last bit on the right will be the Least Significant Bit, or LSB. Terrific! Now that we have a group leader and a group follower, all the bits should be given a rank.

Handing out ranks is serious business and much thought should be given to it. We can start with the LSB and assign that bit the rank of "1," since it is the Least Significant Bit. We can be easy on everyone if we just double that rank for the next bit in line. So, why not just keep doubling the rank for the next bit in line and so on until we get to the MSB or Most Significant Bit. Now our byte looks something like this:



What have we gained? More than meets the eye! When the byte gets off the Byte Express and each bit starts telling us what its current mood is, we can make a differ-

ent and more meaningful interpretation out of the little guys. If everyone is turned "off" except the fourth bit from the right, for example, we can check the rank of that bit and find it is eight (8). Unknown to the bits, they have brought us a message and the message is "8"

Aha, what if we want the bits to get on the bus and carry a message that says "9"? This is a problem because there is no bit with a rank (value) of 9. What to do? I guess the next best thing is to be very nice to the bits that have values of one and eight and turn them both on. When we look at the byte now we see 1 + 8 which gives us our 9. Easy. Even my student Nerdwell can count that high. In fact, Nerdwell can count to 255 because if you add up all the values of the bits, you get 255:

$$1 + 2 + 4 + 8 + 16 + 32 + 64 + 128 = 255$$

How to peek

In using the memory map, we are going to change the values that are stored in many of the bytes that make up

In case you're
wondering, the
difference between
machine language and
assembly language is
not much.

memory. To make the discussion easier, we will usually talk about the bytes as locations or addresses. These addresses are just like those on mailboxes on a long street. They start at 0 at one end of the street and increase until they reach 65535 at the other end (64K . . . remember). Yes, a computer's memory represents a long street. Let's look now at one such address. To see what is inside the location, we will use a BASIC command called PEEK.

To PEEK at memory location 764, type in this line:

```
10 PRINT PEEK(764):GOTO 10
```

Type RUN, press the return key and watch the number 255 print over and over on your screen. Now press the space bar and you should see 32 printing. What is happening? 764 is the location that tells you what key is being pressed. 255 means no key and 32 means the space bar. Now you can see how a program could know if you have pressed a certain key. Try a program something like this:

```
10 A=PEEK(764)
20 IF A=32 THEN PRINT "YOU P
RESSES THE SPACE BAR"
30 GOTO 10
```

Now you can see why it is important to have a memory map. It would be very difficult to know which location to PEEK at without a map of your computer's memory.

How to poke

Now that we can look at memory locations, we also want to be able to change what is inside them. Your wish is our command. Later in the map you will learn that a value of 1 in location 752 will cause the cursor to disappear (the cursor is the little white box on the screen). This can come in handy when you have a whole bunch of text on the screen, and you don't want the cursor up there with it. Since location 752 doesn't normally contain a value of 1, we must change it. The POKE command in BASIC will do this for us:

```
POKE 752, 1
```

That was easy. Now you know that POKEing and PEEKing are what a memory map is all about. For the most part, the map will just tell you different numbers to POKE and PEEK.

ROM and RAM

We encounter another small problem when we mess with memory. Some

memory addresses will allow us to read what is there, but will *not* allow us to write to them. Memory that we can both read from and write to is called RAM, which stands for Random Access Memory. This means that we can put numbers into these kinds of memory locations as well as look at what they already contain. The other kind of memory is called ROM, Read Only Memory. It is just what its name implies—we can only read what number is inside a ROM location, not change it.

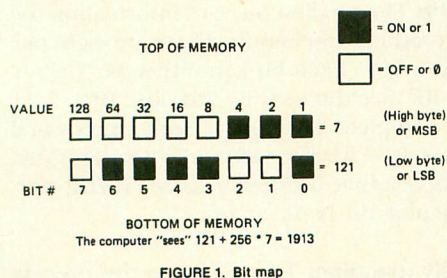
Now the next logical question is how do you know which is which? We will simply tell you as we go through the locations.

Computer mathematics

This section will open new vistas in your horizons. We are going to learn how the computer deals with numbers larger than 255, and also how to use the hexadecimal numbering system.

You must always use decimal numbers with a POKE statement. This means that sometimes you will have to convert between binary, hexadecimal and decimal. The "Bits and Bytes" section covered binary numbers where we turn on and off individual bits. In that section you saw that each memory location can only hold numbers up to 255. To store a value larger than that, we just use two locations in a row.

As an example, look at memory locations 88 and 89 which are called SAVMSC. These locations hold a number that tells where the top of the screen is. Because the screen is usually somewhere near the end of memory (it starts at location 40000 in a 48K Atari), at an address that is way beyond the 255 limit for one memory location, the computer needs two locations to store the address. Remember the ranking of the bits where the MSB, or bit seven as we called it, was valued at 128? If we double that rank again, we get 256. Here is the trick to computer math. Since there is no bit eight to give such a rank to, we give the rank to the entire next byte of 8 bits. Now we just see what the total of this second byte is and multiply it by 256. Figure 1 is a sample:



Humans use dictionaries to speak the same words. In the case of computers, ASCII allows one computer to understand the letters and numbers created on another computer.

When using two byte numbers, we call the first byte in memory the "low byte" because it stores the lower value. It can only count from 0 to 255. The second number counts in multiples of 256 and is called the "high byte."

Let's try to PEEK and POKE a two-byte number. Suppose you wanted to fool the computer into thinking that the screen was in a different place in memory. Such a trick can come in handy when you want to have more than one screen at the same time. First let's look at memory and see where the computer thinks the screen is now. Here's a program to do this:

```
10 SCREEN=PEEK(88)+PEEK(89)*256
20 PRINT SCREEN
```

Addresses 88 and 89 form a two-byte location called SAVMSC which points to where the computer thinks the first byte of screen memory is.

When you run the preceding program, the address that gets printed out will vary, depending on how much memory your computer has. What we're going to do is add 480 to this address so that the computer will think that screen memory starts

halfway down the current screen. All this means is that no text will print in the top half of the screen, and you'll be able to type below the bottom of the screen (although you won't be able to see anything there).

Let's go ahead and replace the old value of SAVMSC with a new number that is exactly 480 more than the old one. Add the following lines to the preceding ones:

```
30 SCREEN=SCREEN+480
40 SCRHI=INT(SCREEN/256)
```

Line 40 finds out what number goes into the high byte of SAVMSC. Remember that the high byte counts the number of pages, or multiples of 256, and that's why we divide by 256.

```
50 SCRLO=SCREEN-SCRHI*256
```

Now we multiply the new high byte by 256 and subtract it from the total to get the low byte.

Finally we place the new values in memory:

```
60 POKE 88,SCRLO:POKE 89,SCRHI
```

The high byte can count from 0 to 255 just like the low byte. This means the largest number we can have using a two-byte address is:

$$255 + (255 * 256) = 65535$$

If we count 0 as a number (since the computer does), that gives a total of 65536, which brings us back again to 64K. We have now come full circle in our discussion, so it is time to go on to something else to challenge you.

Hexadecimal numbers

Come back. You don't have to run away at the sound of those words. Hex, as its friends call it, is not nearly as hard as everyone thinks it is. As a matter of fact, it's really quite simple. But, just in case you don't believe us, I provide both decimal and hexadecimal numbers throughout the map. Now, though, we're going to learn about hex together.

The main use for hex is in assembly language programming. We're not going to worry about that now though, because it's not really important. Instead, we're going to go back to grade school, where we first learned all about the number system. As you'll recall, we use the decimal number system, which means everything is based on powers of 10. For example, the number 452 is equal to $4 * 100 +$

$5 \times 10 + 2 \times 1$, right? In other words, $4 \times 10^2 + 5 \times 10^1 + 2 \times 10^0$.

Now you're not going to believe this, but the particular difference between decimal and hex is that hex is based on powers of 16 instead of 10. For example, 452 hex would equal $4 \times 16^2 + 5 \times 16^1 + 2 \times 16^0$ or $4 \times 256 + 5 \times 16 + 2$, in other words, 1106.

Pretty simple, huh? Actually, there's one more thing that I should probably mention. The number 9 in hex is the same as 9 decimal, but 10 hex is 16 decimal (1×16^1). So how do you write 10, 11, 12, 13, 14 and 15 decimal in hex? Try A, B, C, D, E and F! That's why hex numbers look so confusing. So, for example, F in hex equals 15 decimal, 1F equals 31 ($16 + 15$), and so on. Oh, and by the way, binary is actually the same as hex and decimal, except it counts in powers of 2!

Don't worry, I'm not going to leave you yet. A few more examples should make you a little more comfortable about hex and how to use it, but first a chart (Table 1) to help us out.

Table 1. Hex Conversion Chart

Column #	4th	3rd	2nd	1st	# HEX
	4096	256	16	1	1
	8192	512	32	2	2
	12288	768	48	3	3
	16384	1024	64	4	4
	20480	1280	80	5	5
	24576	1536	96	6	6
	28672	1792	112	7	7
	32768	2048	128	8	8
	36864	2304	144	9	9
	40960	2560	160	10	A
	45056	2816	176	11	B
	49152	3072	192	12	C
	53248	3328	208	13	D
	57344	3584	224	14	E
	61440	3840	240	15	F

Decimal to hex

Let's take the number 9304. Look in the chart to find the largest number that is smaller than 9304. That would be 8192, which is the second number in the fourth column. This means the hex number will have four digits because you found the decimal value 8192 in the fourth column. Write down the "hex #" from the chart in the fourth place:

2 — — —

Now subtract 8192 from the original number of 9304 to leave a remainder of 1112. Looking at the chart again we find the nearest smaller number of 1024 in the third column. Put down the corresponding hex # and subtract 1024 from 1112:

24 — —

Do the same for the new remainder of

88 to find the hex # digit for the second and first place of the hex number. Don't forget to mark hex numbers with a "\$" as is standard with computer types like us. Here is the complete example (Example 1):

Decimal-to-Hex

9304	
-8192 (largest # less than 9304 on chart, 4th column)	2
1112	
-1024 3rd column	4
88	
-80 2nd column	5
8	
-8 1st column	8 = \$2458

Example 1. Decimal-to-hex example

If you find that you have numbers for only the first and third places, this means to add a 0 as a placeholder. \$040F would be an example.

Hex to decimal

Conversion in this direction is even easier. Just look up each hex number in the chart, find the corresponding decimal value, place them in a column and add them up to get a decimal total. Here is the mandatory example (Example 2) you are no doubt expecting:

Hex-to-Decimal

\$4A3F =	4th	3rd	2nd	1st	
	A	3	F		
					15
					48
					2560
					16384
					TOTAL = 19007 (dec)

Example 2. Hex-to-decimal example

Glossary

Next month we will begin publishing the actual memory map. Until then, you should familiarize yourself with the following terms and definitions. As we explain the inner workings of the Atari, we will have to refer to some of the following words. If you find some term we forgot to mention here, it's probably because it is fully explained at the memory location it pertains to. You should also read your BASIC manual in order to understand the terms that have to do with the BASIC language.

6502: This is the heart of the computer, the chip that bosses everybody around. Actually, a lot of people even refer to 6502 as being the computer, since it does have almost all of the brains.

Accumulator: This is a location that is used to temporarily store the results of logic and arithmetic operations. The main accumulator is inside the 6502 chip, but sometimes memory locations are also used as an extra accumulator.

Address: The number assigned to an individual memory location. Each byte in the Atari has its own unique address, much like a house has a street address. The main use of the memory map is to provide you a roadmap to each address so you don't get lost.

Algorithm: A general procedure, plan or method that represents how your program will be written.

ANTIC: This is a chip in the Atari computers that figures out what the screen is supposed to look like.

ASCII: American Standard Code for Information Interchange (pronounced ASK-KEY). Everyone needs a standard or reference to refer to. This allows us to all speak to each other in the same terms. Humans use dictionaries to speak the same words. In the case of computers, ASCII allows one computer to understand the letters and numbers created on another computer. Atari computers do not follow a true ASCII, but have their own code instead which we explain later.

Assembly language: This is a programming language, just like BASIC, except it talks the computer's language instead of having to go through a translator. See "machine language" as well.

ATASCII: ATARI Standard Code for Information Interchange. This is the code the Atari uses to convert letters to numbers and vice versa. See your BASIC manual to find out how it differs from ASCII.

Baud: The rate of transmission of information conveyed between two computers. You usually say "Baud Rate" meaning how fast the two computers are talking to each other. This rate is determined by the bits per second that are being transferred. You encounter this term if you are using a modem, printer, disk drive, terminal or other device that needs to talk to a computer to work. Typical speeds of information transfer are 300, 1200, 2400, 9600 and 19200 bits per second.

Bit: The smallest piece of information the computer can handle. There are eight bits in a byte. Each bit can either be "on" or "off." See the section "Bits and Bytes" for a complete description. Sometimes you'll see "—" for the value of a bit. This just means that it doesn't matter if that particular bit is on or off.

Bit mapping: This refers to the process

of turning individual bits on and off without changing the rest of the byte.

Boot: No, this isn't even close to what it sounds like. "Booting" a program means loading it in when the computer is turned on. For example, if you hold down the START button while turning on the computer, the computer will beep. This means that it expects a boot cassette to be in the cassette player. When you turn on the computer with the disk drive on, you will boot DOS. In other words, any program that loads in without you having to tell it to load is a boot program.

Boundary: As in "4K boundary." This is the end of the block of memory. For example, a 1K boundary would be the end of 1024 byte block.

Buffer: A storage place, usually temporary, where information can come and go without disturbing things.

Bus: A bus is a system of electrical lines shared by all devices that are connected to it. This is a convenient way for these devices to share data. It works just like a party-line telephone. Different parts of the computer talk to each other by getting on the bus and sending messages.

Byte: Pronounced "bite." A collection of eight bits. Each memory address consists of one byte. Since we know at this point bytes and bits can be confusing, we provide a special section elsewhere called "Bits and Bytes" to explain it to you.

Checksum: A checksum is a special byte that the computer uses after talking to something to make sure it understood what was said correctly.

CIO: Central Input/Output. This is Atari's main I/O routine.

Coldstart: A routine the computer goes through after you turn it on and before it lets you tell it what to do.

Color clock: A unit of measurement on the screen. A color clock is the width of a pixel in graphics mode seven. That means that the screen is 160 color clocks wide from border to border.

Controller jack: What you plug your joystick into.

CTIA: This chip takes care of translating the data coming from ANTIC into something the television set can understand.

Cursor: The position on the screen where the next character or pixel will appear. In graphics mode zero, you can see the cursor; it's the white box.

Data: Any kind of information that is needed by a program or by the computer.

Default: When you first turn on the computer, each memory location will contain a value. These initial values are called defaults, meaning that this is what these locations will equal if you don't change them.

Device: Anything that the computer has to talk to is called a device. This includes the disk drive, printer and even the keyboard and television set.

Disable: To turn off. By disabling the BREAK key, for example, you can prevent someone from accidentally stopping your program.

Display list: The program for ANTIC that describes what the screen is to look like.

DLI: Display List Interrupt. See "interrupt."

DMA: Direct Memory Access. The process of getting data from memory to put on the screen.

DOS: Disk Operating System. A program that controls the use of the disk drive. See "OS" as well.

DUP: Disk Utilities Package. This is a bunch of routines to do various things on the disk drive. The DOS menu is actually a list of these routines.

Enable: To turn on. The opposite of disable.

File: A whole bunch of data stored on disk or cassette.

Flag: A signal that a certain condition has been met. In many BASIC programs, variables are used as flags, as demonstrated in the following example:

```
10 IF A = B AND C = D THEN FLAG = 1
.
.
.
50 IF FLAG = 1 THEN 100
```

Floating point: A type of arithmetic

where the decimal point can appear anywhere in the numbers (i.e., it can float around). An example of such numbers would be 1.0, 23.97 and 1.45678E + 04. Floating point numbers take up much more memory than fixed point (integer) numbers.

FMS: File Manager System. This is a group of routines, or handler, to help the computer talk to the disk drive.

GTIA: A fancy version of CTIA.

Handler: A series of routines that tell the OS how to handle a particular device.

HBLANK: Horizontal BLANK. The television set draws the screen one line at a time, from top to bottom and left to right. HBLANK is the time during which it is moving from the end of one line to the beginning of the next.

Hi-res: Pronounced "high rez." This is an abbreviation for "high resolution," which refers to a graphics display with very small dots.

Immediate mode: Using the computer without running a program. For example, if you type in:

```
PRINT 3 + 2
```

and then press RETURN, you will get a result of 5 on the screen immediately.

Index: This is a variable used to keep track of where we are in a loop. For example, in the following statement X would be an index:

```
FOR X = 1 TO 100
```

Internal: If something is internal, then that usually means it is built into the computer.

Interrupt: An interrupt is something that interrupts whatever the computer is doing and tells it to do something else before it continues. You should also see DLI, IRQ, NMI and VBLANK.

I/O: Input/Output (I/O) is nothing more than a fancy way of referring to the computer talking to a device, or vice versa.

IOCB: Input/Output Control Block. This is a place that you use to talk to CIO.

IRQ: Interrupt ReQuest. This is a kind of interrupt that you can tell the computer to ignore (the 6502 can enable or disable it).

Jiffy: A jiffy is $\frac{1}{60}$ of a second, the time that it takes the television set to completely draw the screen once. In European (PAL) systems, a jiffy is $\frac{1}{50}$ of a second.

Jump: The same thing as GOTO. The expression "jump through location" means that the computer will GOTO the address stored in that location.

K: As in 1K, 8K, 16K, etc. 1K is equal to 1024 bytes.

Logical line: A logical line is the space that a program line takes up. It can be one, two or three screen lines (try typing in a BASIC line that is more than three screen lines).

Machine language: Machine language is a way of talking directly to the 6502 chip. Other languages like BASIC have to be translated into machine language before the 6502 can understand them. That takes time, which is why machine language programs run so much faster than BASIC ones. In case you're wondering, the difference between machine language and assembly language is, not much. Machine language is just a bunch of numbers. Assembly language gives these numbers names so that they make more sense.

Masking: When you're bit mapping, you have to have a way of ignoring the bits you're not interested in. This process is called masking, since you essentially place a mask over the bits you don't want to look at.

Nibble: This is going to sound funny, but I swear it's the truth. A nibble is half a byte or four bits.

NMI: Non-Maskable Interrupt. Unlike IRQs, you can't tell the 6502 to ignore this kind of interrupt. DLIs and VBLANK interrupts are both NMIs.

NTSC: A name for the television system that is used in North America. European television is slightly different and uses a system called PAL.

Offset: If you have a whole bunch of bytes making up a table of values or a buffer or something similar, then the offset is the number of the byte in this bunch that you are currently interested in.

OS: Operating System. Its job is to make the computer run. You can think of the OS as the coach directing the players in a game. We can change some of the num-

bers in the operating system to make the computer do what we want, instead of what it normally does.

Page: Computer memory in the Atari is divided into 256 sections, called pages. Each page consists of 256 bytes. The pages are numbered 0 through 255, and you can tell what page a particular location is in by looking at the high byte of its address. For example, location \$09AB would be page nine. See the section called "Computer Mathematics" for an explanation of what a "high byte" and "low byte" are, and also for an explanation of "hexadecimal," which is what that funny number with a "\$" in front of it is.

PAL: The television system used in Europe. See NTSC as well.

Parallel: There are two ways that the computer can talk to something else. One of these is called parallel I/O, which simply means that the data is sent out one byte at a time. See "serial" for the other.

PIA: This chip takes care of the controller jacks.

Pixel: A fancy word for a dot on the screen.

Playfield: Anything that appears on the screen other than a player or missile.

Pointer: A pointer does exactly what it sounds like: points somewhere. Usually this "somewhere" is the location of some information that is needed. The pointer holds the address of this location.

ROM: Read Only Memory. Computer memory that you can't change with the POKE command or anything else (it's OK to PEEK them though). ROM locations even remember their values after you turn the computer off! BASIC and the Atari operating system are stored in ROM.

Scan line: If you look very closely at the screen, you'll see that it's made up of a whole bunch of tiny horizontal lines. These are called scan lines and are the height of a graphics mode eight pixel.

Screen memory: A bunch of bytes somewhere in memory (usually at the end) that ANTIC converts into a picture and sends to GTIA or CTIA which puts it on the screen. In other words, this is where the data that is to appear on the screen is stored. In case you don't understand the difference between this and the display

list, the display list tells ANTIC how to interpret the screen memory (i.e., where is it, does the data represent characters or pixels, how big are they, etc.).

Sector: A group of 128 bytes on the disk. It may be difficult to do, but try to imagine a disk being made up of 40 concentric rings. Now imagine cutting the whole disk into 18 equal-size wedges. Each of the wedges will have 40 pieces for a total of 720 pieces altogether. Well, each of these pieces is a sector.

Serial: This is a method of I/O that sends data out one bit at a time rather than one byte at a time. See "parallel" also.

Shadow register: A shadow register is a RAM location that acts like a messenger for a chip location. Any changes to the shadow register are sent to the chip and vice versa. This is necessary because a chip location can't be changed permanently, and so it relies on its shadow register to get information from you.

SIO: Serial Input/Output. This refers to a routine in the OS that takes care of serial I/O. See "serial."

Timeout: Sometimes a device needs a little time to think and breathe, so it takes a timeout. If it takes too long a timeout, however, the computer gets upset and refuses to talk to it anymore.

User: You, anybody or anything that uses the computer. BASIC is considered a user by the OS, and the OS is considered a user by the 6502. Similarly, BASIC considers your program a user, and your program considers anyone that runs it a user.

VBLANK: Vertical BLANK. We already saw that the television set draws the screen over and over from top to bottom and left to right (see HBLANK). VBLANK is the time during which the television set is going from the bottom of the finished screen back to the top to start drawing again.

Vector: This is another kind of pointer. It refers to the starting address of a routine. The computer needs to know where to look for things, and the vectors help it along the way. Usually a vector references the starting address of a machine language subroutine.

Warmstart: A routine the computer goes through after you press SYSTEM RESET before it lets you tell it what to do. **A**



Animation

by Ron Goodman

There's a lot of computer animation today, and many Atari owners probably wish they could do some of the animation that they see in the movies and the video arcade. But there are several problems. Most computer animation done in movies is not produced "real-time." That is to say that it's created a frame at a time and then photographed. From there it may be modified with both digital and analog filters, so each frame may end up taking several minutes to produce. Some of the best quality graphics produced with techniques such as "ray-tracing" may even take several hours per frame to create on a mainframe computer.

The second problem is of course that an Atari is not a mainframe computer. The Atari instead has a relatively slow 6502 microprocessor. With the overhead added by using a high-level language like Atari BASIC, most programmers throw up their hands in disgust.

Some people abandon BASIC and try assembly language. This is a good recourse, but even Atari assembly is not that fast, and many people don't want to give up the ease of using a high-level

language.

In this article I will demonstrate my solution to this problem using two machine language routines. They will allow you to do your entire animation program in BASIC without ever touching an editor/assembler and without buying a new computer! Because, in spite of the weaknesses of a 6502 computer, the Atari computer is still the best computer for graphics and animation under \$1,000, except perhaps for the Atari STs.

What is animation?

The process of animation involves displaying pictures, one after the other, usually in an attempt to simulate motion. Cartoonists have used the method for years, and in reality moviemakers do basically the same thing too (although they don't call it animation, since the pictures are generated from a camera rather than an artist or computer). Different methods of animation generally differ in the way that the images are created, and the speed that the images are displayed (frames per second.)

There are two general categories of computer-graphics animation, although they often can be found combined into a hybrid form. The first is frame generating, where the entire frame is created, displayed and then erased when the next frame is generated. The second method is frame modification. In this method an original image is displayed and then changed only in the parts where something is supposed to move.

Frame generation is very flexible and usually relatively easy to program. The first problem is that it can be slow, since even the slightest change requires that the entire picture be erased and regenerated. The second is that it usually requires twice as much video memory: one chunk for the picture being displayed and another chunk for the image being generated.

Frame modification is often much faster, but it's not as flexible and often does not look as good as frame generation. It's unfortunately limited to simple animation—unless you want your program to get very slow and complex. This method is also usually less memory consuming than frame generation, although this advantage can be sacrificed in order to use a workscreen to make smooth picture transitions like frame generation. Frame modification is what most BASIC programmers like to use because of the speed advantage.

Listings 1 and 2 are two programs which visibly do the same animation, but Listing 1 uses frame generation and Listing 2 uses frame modification. The programs move a horizontal line from the far left of the screen to the far right. In frame generation, we display a line at the far left (0,100). Then that line is erased and a new line is displayed at (1,100)—then (2,100), (3,100) and so on until it reaches the far right. This method is slow because the entire line must be redrawn to move it. Also, increasing the length of the line would slow down the program. This method is flexible. The line could easily be made to

move two pixels at a time by just changing the $X = X + 1$ to an $X = X + 2$ or made to move vertically by changing the value of Y.

The frame modification routine has a different approach. It draws the line at the far left (0,100). Then it erases the leftmost pixel of the line and adds a pixel to the right end over and over until the line appears to have moved to the far right of the screen. This method is fast because only two pixels have to be drawn per frame. The line could have been 100 pixels in length with no loss in speed, since the line is not being redrawn each frame. However, the movement is not as flexible. Since we are using a trick to move the line, moving two pixels at a time would not be a simple change. And moving the line vertically would be completely different.

For moving a line across the screen, most people choose frame modification because speed is the most important factor. But type in the two programs in Listings 3 and 4. They each have approximately the same output—four simple stick figures bounce around in a box with a square inside it. In frame modification (Listing 4) the action is sloppy. When part of the picture crosses the inner square it erases it momentarily. In frame generation (Listing 3) the movement is smooth and no damage occurs when two objects cross paths. In addition, frame generation is slightly faster. Why? Because most of the picture changes each frame.

Using frame generation

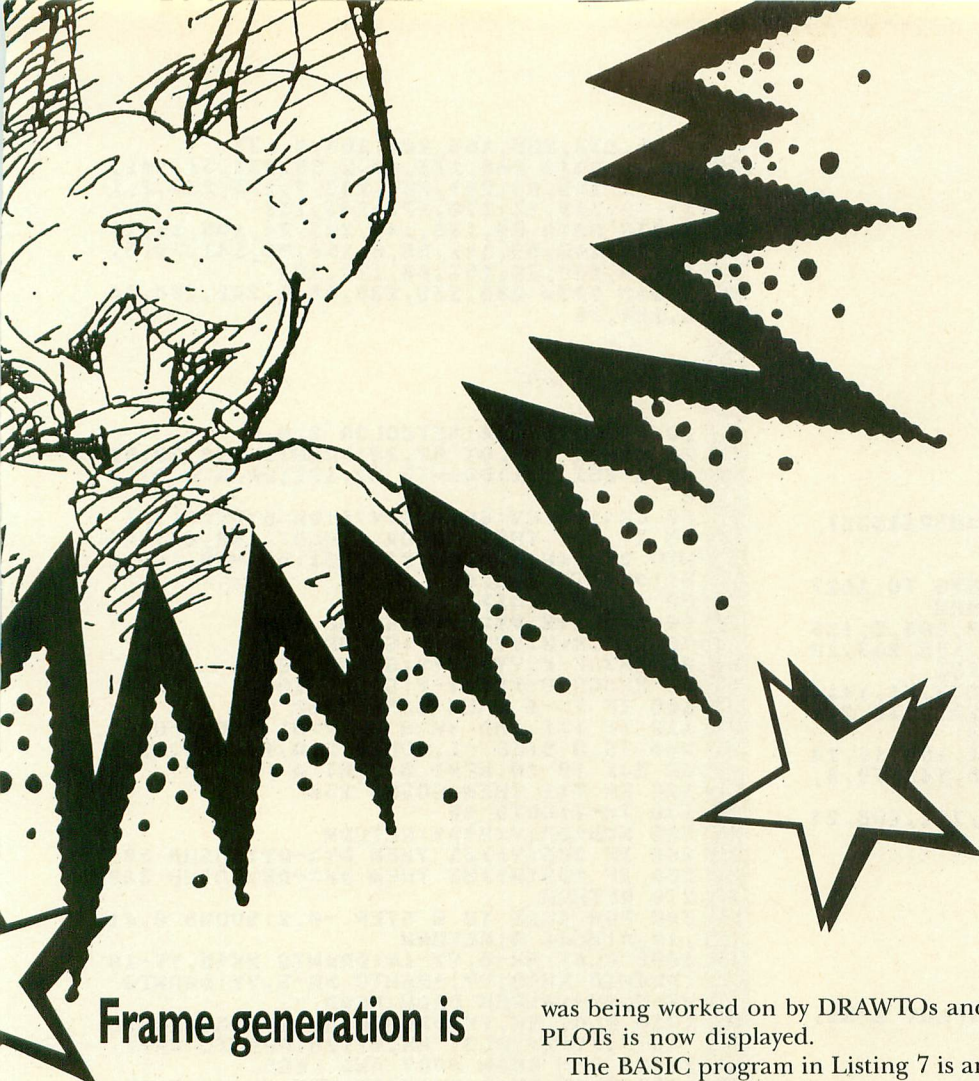
Some nice frame generation can be done on the Atari since it has page-flipping abilities. This is the ability to tell the computer where an image is stored in memory, which allows you to change the Atari's screen from displaying one image to the next virtually instantly. The program in Listing 5 is an optimized assembly language listing of a program that will allow BASIC programmers to easily use this feature. The program in Listing 6 is a BASIC program that can be appended to the end of your BASIC animation program. A GOSUB to Line 30000 will load the machine-language routine from Listing 5 into Page 6 memory. Remember that Page 6 is unused by BASIC, so this routine is safe and takes up no program memory.

This program fools the Atari into displaying one image and working on another. The graphics commands DRAWTO and PLOT will draw the invisible background image to create the next



frame while the current image is being displayed. Calling the machine-language routine with an $X = USR(1536)$ will cause the invisible image to be displayed and the image being displayed to be disposed. Then you draw the next frame and do another $X = USR(1536)$. The machine-language handles all the details in just a 15th of a second, so all you need to do is call $X = USR(1536)$, draw an image, call $X = USR(1536)$, draw next image, call $X = USR(1536)$, draw next image, etc. Although it is not always necessary, it is a good general practice to call $X = USR(1536)$ just after the GRAPHICS statement and before you begin drawing. This program will work in graphics mode 24, 9, 10 and 11 (24 is mode 8 without a text window).

What is the routine doing? First it modifies the display list to point to one chunk of memory. Next it adjusts the video memory pointer at locations 88 and 89 of BASIC to point at another chunk. DRAWTO and PLOT commands are drawn in the chunk pointed to by the video memory pointer and so are not seen. Then, the next time the machine-language routine is called, the pointers swap. The 7680 byte area of memory that was being displayed is filled with zeros (cleared) and is ready to be used for the next frame. And the 7680 byte chunk that



Frame generation is very flexible and usually relatively easy to program. The first problem is that it can be slow, since even the slightest change requires that the entire picture be erased and regenerated.

was being worked on by DRAWTOs and PLOTs is now displayed.

The BASIC program in Listing 7 is an example of a complete frame-generation program. It's a clock program that updates its display about every ten seconds. The hands move smoothly because the picture appears to be drawn instantly. The clock chimes at the half hour and gongs at the hour. It has ten settable alarms. To set the alarms, press the space-bar while the program is running. To quit press ESCAPE. The alarms and the current time are maintained even if the program is stopped (or even deleted!), but the alarms will only sound while the program is running. The screen will progressively dim after a few minutes of inactivity instead of going into attract mode to protect the screen more effectively and attractively. This way the computer can be left on over-night or even for days.

The BASIC program in Listing 8 is another example. In this program you draw a line picture with the joystick. Do this by moving the graphics cursor to various points and pressing the joystick fire button. The program will connect the points. After you draw your picture, press START. Now the joystick will cause your picture to expand or contract by pushing up or down, and move left or right by pressing the stick left or right. Pressing

the fire button will cause the picture to be rotated by ten degrees. The pictures are generated quickly and appear to be redrawn instantly. Unrestricted movement like this is easy with frame generation and the obvious choice over frame modification for such an application.

Using frame modification

Using frame modification doesn't have to be choppy and chunky. You can use the page-flipping technique that we used in frame generation. Listing 9 is an assembly language program for this purpose. It works in a different manner, of course. It must be called before you begin drawing. Now all of your drawing is done in the background screen. When the frame is ready to be displayed you do another USR call and the frame is copied into the foreground for display, leaving the background copy intact. Future modifications are still done on the background. And whenever it is time for an update you do another USR call. So all you need to do is `X=USR(1664)`, draw the picture, `X=USR(1664)`, modify the picture, `X=USR(1664)`, modify the picture, `X=USR(1664)`, etc.

Listing 10 can be appended to your BASIC program to load this routine. Like the frame generation program the machine language is stored in unused Page 6 so it doesn't use program memory. Note also that both Listing 6 and Listing 10 have unique line numbers and can be appended for combined frame modification and frame generation applications. The machine-language routines are also stored in different areas of Page 6 so that there is no conflict of memory there.

Other methods of animation

The Atari has some special features for animation that can be used. Player-missile graphics are useful for horizontal animation, or if some machine language is used, for vertical animation too. There is collision detection in hardware to simplify computations for games, but player/mis-sile graphics can also be used to simplify animation computations. For more complete explanations of these features, you should read *De Re Atari*.

Ron Goodman is a computer music major at the California Institute of Technology. His greatest fascination is with computer graphics and computer music (MIDI/ST applications).



LISTING 1: BASIC

```

IQ 5 GOSUB 30000
RR 10 GRAPHICS 24:COLOR 1
PC 20 X=0:Y=100
OY 30 PLOT X,Y:DRAWTO X+50,Y:A=USR(1536)
DJ 40 X=X+1:IF X<268 THEN 30
RM 50 GOTO 10
TV 30000 RESTORE 30010:FOR X=1536 TO 1627
:READ Y:POKE X,Y:NEXT X:RETURN
PW 30010 DATA 160,0,24,173,48,2,105,5,133
,203,173,49,2,105,0,133,204,165,203,10
5,96,133,205,165,204,105,0,133
QS 30020 DATA 206,173,49,2,56,233,31,141,
230,2,165,89,209,203,208,7,173,230,2,1
33,89,208,13,170,177,203,133
NW 30030 DATA 89,138,145,203,24,105,15,14
5,205,165,89,141,80,6,165,88,141,79,6,
169,0,162,30,153,80,127,200
WF 30040 DATA 208,250,238,80,6,202,208,24
4,104,96
  
```

LISTING 2: BASIC

```

RR 10 GRAPHICS 24:COLOR 1
PC 20 X=0:Y=100
KX 30 PLOT X,Y:DRAWTO X+50,Y
LJ 40 COLOR 0:PLOT X,Y:COLOR 1:PLOT X+51,
Y:X=X+1:IF X<268 THEN 40
RM 50 GOTO 10
  
```

LISTING 3: BASIC

```

IQ 5 GOSUB 30000
FH 10 GRAPHICS 24:SETCOLOR 2,0,0
IW 20 CX=160:CY=86:X=0:Y=0:DX=6:DY=5:T=1
GZ 30 COLOR 1:PLOT 67,20:DRAWTO 252,20:DR
AWTO 252,171:DRAWTO 67,171:DRAWTO 67,2
0
QC 40 PLOT 100,40:DRAWTO 219,40:DRAWTO 21
9,151:DRAWTO 100,151:DRAWTO 100,40
JZ 60 XX=CX+X:YY=CY+Y:GOSUB 1000
LI 70 XX=CX+X:YY=CY-Y:GOSUB 1000
KT 80 XX=CX-X:YY=CY+Y:GOSUB 1000
MC 90 XX=CX-X:YY=CY-Y:GOSUB 1000
VO 95 T=USR(1536)
UI 100 IF X=0 OR Y=0 THEN FOR A=50 TO 0 5
TEP -1:SOUND 0,A,12,A/3.23:FOR B=1 TO
10:NEXT B:NEXT A
HN 110 GOSUB 250:GOSUB 200
QC 130 GOTO 30
PT 200 X=X+DX:Y=Y+DY:RETURN
MM 250 IF ABS(Y)>53 THEN DY=-DY:GOSUB 300
LV 260 IF ABS(X)>83 THEN DX=-DX:GOSUB 300
ZM 270 RETURN
AF 300 FOR A=15 TO 0 STEP -0.2:SOUND 0,41
,12,A:NEXT A:RETURN
SM 1000 PLOT XX-5,YY-10:DRAWTO XX+5,YY-10
:DRAWTO XX+5,YY:DRAWTO XX-5,YY:DRAWTO
XX-5,YY-10:REM DRAW HEAD
VZ 1010 PLOT XX,YY:DRAWTO XX,YY+20:DRAWTO
XX-7,YY+29:PLOT XX,YY+20:DRAWTO XX+7,
YY+29:REM DRAW BODY AND LEGS
TB 1020 PLOT XX-6,YY+9:DRAWTO XX,YY+7:DR
AWTO XX+6,YY+9:REM DRAW ARMS
AI 1030 RETURN
TV 30000 RESTORE 30010:FOR X=1536 TO 1627
:READ Y:POKE X,Y:NEXT X:RETURN
PW 30010 DATA 160,0,24,173,48,2,105,5,133
,203,173,49,2,105,0,133,204,165,203,10
  
```

```

5,96,133,205,165,204,105,0,133
QS 30020 DATA 206,173,49,2,56,233,31,141,
230,2,165,89,209,203,208,7,173,230,2,1
33,89,208,13,170,177,203,133
NW 30030 DATA 89,138,145,203,24,105,15,14
5,205,165,89,141,80,6,165,88,141,79,6,
169,0,162,30,153,80,127,200
WF 30040 DATA 208,250,238,80,6,202,208,24
4,104,96
  
```

LISTING 4: BASIC

```

FH 10 GRAPHICS 24:SETCOLOR 2,0,0
GY 20 COLOR 1:PLOT 67,20:DRAWTO 252,20:DR
AWTO 252,171:DRAWTO 67,171:DRAWTO 67,2
0
IX 30 CX=160:CY=86:X=0:Y=0:DX=6:DY=5:T=1
EC 40 IF T=1 THEN COLOR 1:PLOT 100,40:DR
AWTO 219,40:DRAWTO 219,151:DRAWTO 100,1
51:DRAWTO 100,40
BS 50 COLOR (T=1)
JZ 60 XX=CX+X:YY=CY+Y:GOSUB 1000
LI 70 XX=CX+X:YY=CY-Y:GOSUB 1000
KT 80 XX=CX-X:YY=CY+Y:GOSUB 1000
MC 90 XX=CX-X:YY=CY-Y:GOSUB 1000
BZ 100 IF T=-1 THEN GOSUB 200
FK 110 IF T=1 AND (X=0 OR Y=0) THEN FOR A
=50 TO 0 STEP -1:SOUND 0,A,12,A/3.23:IF
OR B=1 TO 10:NEXT B:NEXT A
DV 120 IF T=1 THEN GOSUB 250
II 130 T=-T:GOTO 40
PT 200 X=X+DX:Y=Y+DY:RETURN
MM 250 IF ABS(Y)>53 THEN DY=-DY:GOSUB 300
LV 260 IF ABS(X)>83 THEN DX=-DX:GOSUB 300
ZM 270 RETURN
AF 300 FOR A=15 TO 0 STEP -0.2:SOUND 0,41
,12,A:NEXT A:RETURN
SM 1000 PLOT XX-5,YY-10:DRAWTO XX+5,YY-10
:DRAWTO XX+5,YY:DRAWTO XX-5,YY:DRAWTO
XX-5,YY-10:REM DRAW HEAD
VZ 1010 PLOT XX,YY:DRAWTO XX,YY+20:DRAWTO
XX-7,YY+29:PLOT XX,YY+20:DRAWTO XX+7,
YY+29:REM DRAW BODY AND LEGS
TB 1020 PLOT XX-6,YY+9:DRAWTO XX,YY+7:DR
AWTO XX+6,YY+9:REM DRAW ARMS
AI 1030 RETURN
  
```

LISTING 5: ASSEMBLY

```

10 ;Page flipping routine. After using
20 ;GR.24, GR.9-11, call this routine
30 ;with X=USR(1536) to toggle
40 ;background and foreground page.
50 ;Background page is the one that
60 ;PLOT and DRAWTO effect, and
70 ;foreground is the one that is
75 ;displayed.
80 ;
90 * = $0600
0100 MEMTOP = $02E5 Mem top pointer.
0110 SAVMSC = $58 Video mem pointer
0120 SDL5TL = $0230 Start of DL.
0130 DLPNT1 = $CB Two unused words
0140 DLPNT2 = $CD in zero page.
0150 LDY #0
0160 CLC
0170 LDA SDL5TL Find the
0180 ADC #5 two
0190 STA DLPNT1 display
0200 LDA SDL5TL+1 list
0210 ADC #0 references
0220 STA DLPNT1+1 to video
0230 LDA DLPNT1 memory
0240 ADC #96 and store
0250 STA DLPNT2 them in
0260 LDA DLPNT1+1 unused
0270 ADC #0 part of
0280 STA DLPNT2+1 page 0.
0290 LDA SDL5TL+1 Find area
0300 SEC to store
0310 SBC #31 screen 2
0320 STA MEMTOP+1 safely.
0330 LDA SAVMSC+1 Insure that
  
```

```

0340    CMP (DLPNT1),Y foreground
0350    BNE NORM and
0360    LDA MEMTOP+1 background
0370    STA SAUM5C+1 are
0380    BNE CLEAR different.
0390    NORM TAX Swap both
0400    LDA (DLPNT1),Y of the
0410    STA SAUM5C+1 foreground
0420    TAX pointers
0430    STA (DLPNT1),Y and
0440    CLC the one
0450    ADC #50F background
0460    STA (DLPNT2),Y pointer.
0470    CLEAR LDA SAUM5C+1 Set up the
0480    STA SCRPN1+1 indexed
0490    LDA SAUM5C addressing
0500    STA SCRPN1 command.
0510    LDA #0 Quickly
0520    LDX #30 clear
0530    LOOP STA 0,Y out
0540    INY 7680 byte
0550    BNE LOOP buffer
0560    INC SCRPN1+1 for the
0570    DEX new
0580    BNE LOOP screen.
0590    PLA Unused parameter
0600    RTS Return.
0610    SCRPN1 = LOOP+1
0620    ;The SCRPN1 pointer is used to
0630    ;modify code on the fly. This
0640    ;allows us to use Indexed
0650    ;addressing which is faster
0660    ;than Post-indexed indirect
0670    ;addressing in the inside loop.
0680    .END

```

LISTING 6: BASIC

```

TV 30000 RESTORE 30010:FOR X=1536 TO 1627
:READ Y:POKE X,Y:NEXT X:RETURN
PW 30010 DATA 160,0,24,173,48,2,105,5,133
,203,173,49,2,105,0,133,204,165,203,10
5,96,133,205,165,204,105,0,133
QS 30020 DATA 206,173,49,2,56,233,31,141,
230,2,165,89,209,203,208,7,173,230,2,1
33,89,208,13,170,177,203,133
WW 30030 DATA 89,138,145,203,24,105,15,14
5,205,165,89,141,80,6,165,88,141,79,6,
169,0,162,30,153,80,127,200
WF 30040 DATA 208,250,238,80,6,202,208,24
4,104,96

```

LISTING 7: BASIC

```

TM 10 REM Press space bar to set alarms.
SR 20 GRAPHICS 0:SETCOLOR 2,0,0:POSITION
13,8:?"[ALRM]";
FV 30 XFR=100:YMK=90:YHR=70:XMI=90:YSE=90
NV 40 YFR=80:YMK=72:YHR=56:YMI=72:YSE=72:
YDI=34:ALRMB=1645:HALF=0
IG 50 IF PEEK(ALRMB-1)<>80 THEN FOR X=ALR
MB TO ALRMB+19:POKE X,0:NEXT X:POKE AL
RMB-1,80
TI 60 DIM S$(2),T$(2),M$(2),W$(8):SHFT=60
*12*3600:CX=160:CY=95:PIOVER2=1.5708:G
OSUB 670:GRAPHICS 0:SETCOLOR 2,0,0
SQ 70 CLOSE #4:OPEN #4,12,0,"K:"
MX 80 ? "Is the clock already set? (Y/N)"
:? "(Or press H for HELP.)";
FS 90 GET #4,A:IF A=89 THEN GOSUB 1480:G
OTO 260
QW 100 IF A=72 THEN GOSUB 1360:GOTO 80
XQ 110 IF A<>78 THEN 90
CQ 120 ? "N"
GZ 130 TRAP 140:?" :PRINT "Is it (A)M or (
P)M?";
GX 140 GET #4,A:IF A=65 THEN S$="AM":GOTO
170
FE 150 IF A=80 THEN S$="PM":GOTO 170
NK 160 GOTO 140
DY 170 ? CHR$(A)
YM 180 TRAP 180:PRINT " What is the hour
";:INPUT H

```

```

TF 190 TRAP 190:PRINT " How many minutes
";:INPUT M
WI 200 TRAP 200:PRINT " How many seconds
";:INPUT S
IT 210 TRAP 0:GOSUB 1480
CY 220 JIF=60*(S+60*M+3600*H):IF H<>12 AN
D S$="PM" THEN JIF=JIF+SHFT
KN 230 IF H=12 AND S$="AM" THEN JIF=JIF+5
HFT
GI 240 B18=INT(JIF/65536):JIF=JIF-65536*B
18:B19=INT(JIF/256):B20=JIF-256*B19
MR 250 POKE 20,0:POKE 18,B18:POKE 19,B19:
POKE 20,B20:REM POKE 20,0 INSURES NO T
URNOVERS WHILE SETTING.
RU 260 JIF=(PEEK(18)*65536+PEEK(19)*256+P
EEK(20)):IF JIF/5399999 THEN JIF=JIF-5
184000:GOTO 240
WS 270 JIF=INT(JIF/60)
PF 280 H=JIF/3600:JIF=JIF-3600*INT(H):M=J
IF/60:S=JIF-60*INT(M):IH=INT(H):IM=INT
(M)
ZA 290 H=H/1.9108-PIOVER2:M=M/9.5493-PIOV
ER2:S=S/9.5493-PIOVER2
JX 300 IF PEEK(77)>122 THEN POKE 77,122
ZW 310 SETCOLOR 1,0,15.4-PEEK(77)/10
CI 320 GOSUB 560:REM DRAW CLOCK FRAME
LD 330 PLOT CX,CY:DRAWTO CX+0.7*XHR*COS(H
-0.12),CY+0.7*YHR*SIN(H-0.12):DRAWTO C
X+XHR*COS(H),CY+YHR*SIN(H)
UH 340 DRAWTO CX+0.7*XHR*COS(H+0.12),CY+0
.7*YHR*SIN(H+0.12):DRAWTO CX,CY
ME 350 PLOT CX+1,CY:DRAWTO 1+CX+0.7*XHR*C
OS(H-0.12),CY+0.7*YHR*SIN(H-0.12):DRAW
TO 1+CX+XHR*COS(H),CY+YHR*SIN(H)
FD 360 DRAWTO 1+CX+0.7*XHR*COS(H+0.12),CY
+0.7*YHR*SIN(H+0.12):DRAWTO 1+CX,CY
QG 370 PLOT CX,CY:DRAWTO CX+0.7*XMI*COS(M
-0.07),CY+0.7*YMI*SIN(M-0.07):DRAWTO C
X+XMI*COS(M),CY+YMI*SIN(M)
MM 380 DRAWTO CX+0.7*XMI*COS(M+0.07),CY+0
.7*YMI*SIN(M+0.07):DRAWTO CX,CY
TH 390 PLOT CX+1,CY:DRAWTO 1+CX+0.7*XMI*C
OS(M-0.07),CY+0.7*YMI*SIN(M-0.07):DRAW
TO 1+CX+XMI*COS(M),CY+YMI*SIN(M)
XJ 400 DRAWTO 1+CX+0.7*XMI*COS(M+0.07),CY
+0.7*YMI*SIN(M+0.07):DRAWTO 1+CX,CY
ZL 410 REM PLOT CX,CY:DRAWTO CX+X5E*COS(S
),CY+Y5E*SIN(S):REM TOO SLOW!!!
HT 420 REM PLOT CX+1,CY:DRAWTO 1+CX+X5E*C
OS(S),CY+Y5E*SIN(S)
TP 430 GOSUB 1240
UL 440 IF PEEK(764)=33 THEN GOSUB 940
JA 450 IF PEEK(764)=28 THEN GRAPHICS 0:G
OSUB 1200:CLR:END
WW 460 POKE 764,255
DI 470 X=USR(1536)
AF 480 IF IM=0 AND HALF=0 THEN GOSUB 760
JR 490 IF IM=1 THEN HALF=0
BD 500 IF IM=30 AND HALF=0 THEN GOSUB 810
JT 510 IF IM=31 THEN HALF=0
EG 520 C=0
NV 530 IF IH=PEEK(ALRMB+2*C) AND IM=PEEK(
ALRMB+2*C+1) THEN GOSUB 890:GOTO 260
GJ 540 C=C+1:IF C<10 THEN 530
OS 550 GOTO 260
BD 560 FOR C=0 TO 11:PLOT X(C),Y(C):DRAW
TO EX(C),EY(C):NEXT C
UT 570 FOR C=0 TO 11:PLOT X(C)+1,Y(C):DRA
WTO EX(C)+1,EY(C):NEXT C
XV 580 FOR C=0 TO 11:PLOT X(C)+2,Y(C):DRA
WTO EX(C)+2,EY(C):NEXT C
AP 590 PLOT PX(11),PY(11):FOR C=0 TO 11:D
RAWTO PX(C),PY(C):NEXT C
QE 600 PLOT PX(11)+2,PY(11):FOR C=0 TO 11
:DRAWTO PX(C)+2,PY(C):NEXT C
NO 610 PLOT PX(11)+1,PY(11):FOR C=0 TO 11
:DRAWTO PX(C)+1,PY(C):NEXT C
RG 620 PLOT PX1(11)+2,PY1(11):FOR C=0 TO
11:DRAWTO PX1(C)+2,PY1(C):NEXT C
OM 630 PLOT PX1(11)+1,PY1(11):FOR C=0 TO
11:DRAWTO PX1(C)+1,PY1(C):NEXT C
XU 640 PLOT PX1(11),PY1(11):FOR C=0 TO 11
:DRAWTO PX1(C),PY1(C):NEXT C
ZK 650 FOR X=0 TO 4:FOR Y=X TO 5:PLOT PX2
(X),PY2(X):DRAWTO PX2(Y),PY2(Y):NEXT Y
:NEXT X

```

```

ZO 660 RETURN
XK 670 DIM X(11),Y(11),EX(11),EY(11),PX(1
1),PY(11),PX1(11),PY1(11),PX2(11),PY2(
11):C=0
PP 680 FOR X=0 TO 6.27 STEP 0.5236:X(C)=C
05(X)*(XMK-2)+CX:EX(C)=COS(X)*(XMK+2)+
CX
TO 690 Y(C)=SIN(X)*(YMK-2)+CY:EY(C)=SIN(X
)*(YMK+2)+CY
NW 700 PX1(C)=COS(X)*XFR*1.1+CX:PY1(C)=SI
N(X)*YFR*1.1+CY
EE 710 PX(C)=COS(X)*XFR+CX:PY(C)=SIN(X)*Y
FR+CY:C=C+1:NEXT X
MZ 720 FOR C=0 TO 5:PX2(C)=COS(C*1.0472)*
XFR/5+CX:PY2(C)=SIN(C*1.0472)*YFR/5+CY
:NEXT C
CE 730 IF PEEK(1640)<>96 THEN GOTO 30000
ZL 740 RETURN
SK 750 REM ROUTINE TO DO HOURLY CHIME
BC 760 HALF=1:GOSUB 850:FOR X=1 TO N:FOR
X=3 TO 15 STEP 4: SOUND 0,100,10,X: SOUND
0,1,50,12,X: SOUND 2,102,10,X: NEXT X
KV 770 FOR X=14 TO 0 STEP -0.3: SOUND 0,10
0,10,X: SOUND 1,50,12,X: SOUND 2,102,10,
X: NEXT X
SJ 780 FOR X=1 TO 200:NEXT X:NEXT C
ZU 790 RETURN
PL 800 REM ROUTINE TO DO HALF HOUR CHIME
RL 810 HALF=1:GOSUB 850:FOR X=10 TO 3 STE
P -1:FOR C=0 TO 1: SOUND C,100+C,10,X:N
EXT C:NEXT X
ZB 820 FOR X=10 TO 0 STEP -1:FOR C=0 TO 3
: SOUND C,100+C,10,X: NEXT C:FOR C=1 TO
25:NEXT C:NEXT X
ZK 830 RETURN
QC 840 REM ROUTINE USED BY HOURLY/HALF HO
UR CHIME FOR BOOKKEEPING AND TO SILENCE
CHIMES AT LATE NIGHT.
JY 850 IF IH<8 OR IH>22 THEN POP :RETURN
:REM QUIET MODE FROM 11PM TO 7:59AM
QH 860 N=IH:IF IH>12 THEN N=IH-12
ZS 870 RETURN
YB 880 REM ALARM SOUND ROUTINE
QP 890 POKE 764,255:U=30:SETCOLOR 1,0,15:
POKE 77,0
RV 900 FOR X=1 TO 23:FOR Y=0 TO 255 STEP
3: SOUND 0,Y,10,15: SOUND 1,U,10,15:U=U+
1:IF U>255 THEN U=0
OI 910 IF PEEK(764)=255 THEN NEXT Y:NEXT
X
CN 920 POKE 764,255: SOUND 0,0,0,0: SOUND 1
,0,0,0: RETURN
KX 930 REM ROUTINE TO SET ALARMS. NOTE A
LARMS ARE SET IN MEMORY SO ARE NOT LOS
T BETWEEN RUNS
SM 940 GRAPHICS 0:SETCOLOR 2,0,0:FOR X=0
TO 9: ? CHR$(65+X);"J ";C=PEEK(ALRMB+2
*X)
TG 950 IF C=0 THEN ? "No alarm set.":NEXT
X:GOTO 970
LC 960 D=PEEK(ALRMB+1+2*X):GOSUB 1140: ? C
;" ";M$;" ";T$:NEXT X
YN 970 ? : ? "Press: X to exit or": ? "
A-J to set an alarm"
SJ 980 GET #4,A:IF A=88 THEN GOSUB 1480:R
ETURN
NJ 990 IF A<65 OR A>74 THEN 980
BQ 1000 A=A-65:IF PEEK(ALRMB+2*A)=0 THEN
1030
WT 1010 ? : ? "Alarm [ ]:CHR$(A+193);" [ ] is
already set at" : ?
IK 1020 C=PEEK(ALRMB+2*A):D=PEEK(ALRMB+1+
2*A):GOSUB 1140: ? C;" ";M$;" ";T$:
FF 1030 ? : ? : ? " Set [ ]:CHR$(
A+65);" [ ] for what hour": ? "
[0 to unset alarm]":INPUT C
TV 1040 IF C=0 THEN POKE ALRMB+2*A,C:GOTO
940
WP 1050 ? "How many minutes after the hou
r":INPUT D:POKE ALRMB+2*A+1,D
UJ 1060 PRINT " (A)M or
(P)M?":
TV 1070 GET #4,X:IF X=65 THEN T$="AM":GOT
O 1100
QQ 1080 IF X=80 THEN T$="PM":GOTO 1100
RC 1090 GOTO 1070

```

```

IJ 1100 IF C<12 AND T$="PM" THEN C=C+12
WE 1110 IF C=12 AND T$="AM" THEN C=C+12
RJ 1120 POKE ALRMB+2*A,C:GOTO 940
KD 1130 REM TO CONVERT C(HR) AND D(MIN) T
O PRINTABLE FORMS C;M$;T$
VF 1140 T$="PM":IF C<12 OR C=24 THEN T$="
AM"
HR 1150 IF C>12 THEN C=C-12
IY 1160 IF D<10 THEN M$="0":M$(2)=STR$(D)
:GOTO 1180
EK 1170 M$=STR$(D)
AZ 1180 RETURN
ZF 1190 REM ROUTINE TO SHOW CURRENT TIME
VALUES. USEFUL FOR TIMING SOMETHING.
IY 1200 JIF=PEEK(18)*65536+PEEK(19)*256+P
EEK(20): ? "JIFFIES = ";JIF:JIF=INT(JIF
/60): ? "SECONDS = ";JIF
LY 1210 C=INT(JIF/3600):JIF=JIF-3600*C:D=
INT(JIF/60):S=JIF-60*D:GOSUB 1140: ? "T
IME IS" ";C;" ";M$;" ";S;" ";T$
AJ 1220 RETURN
QA 1230 REM PRINT TIME IN DIGITAL FORM TO
0
TN 1240 MID=36
EB 1250 COLOR 0:FOR X=CY+YDI+1 TO CY+YDI+
10:PLOT CX-WID,X:DRAWTO CX+WID,X:NEXT
X
BS 1260 COLOR 1:FOR X=CY+YDI TO CY+YDI+11
STEP 11:PLOT CX-WID,X:DRAWTO CX+WID,X
:NEXT X
EJ 1270 FOR X=0 TO 1:PLOT CX-WID+X,CY+YDI
:DRAWTO CX+WID+X,CY+YDI+11:NEXT X
YS 1280 FOR X=0 TO 1:PLOT CX+WID+X,CY+YDI
:DRAWTO CX+WID+X,CY+YDI+11:NEXT X
AI 1290 C=IN:D=IM:GOSUB 1140:M$=STR$(C):I
F C<10 THEN M$=" ":M$(2)=STR$(C)
PI 1300 M$(3)="":M$(4)=M$:M$(6)=" "
GU 1310 M$(7)=T$:Y=CY+YDI+2:X=CX-WID:GOSU
B 1330:RETURN
LL 1320 REM WRITE STRING M$ AT X,Y IN GR.
0
TN 1330 A=40*Y+PEEK(88)+PEEK(89)*256+INT(
(X)/8):B5=256*PEEK(756)
JY 1340 FOR C=1 TO LEN(M$):D=ASC(M$(C,C))
:IF D<95 THEN D=D-32
BS 1350 BT=B5+D*8:FOR CC=BT TO BT+7:POKE
A+C+40*(CC-BT),PEEK(CC):NEXT CC:NEXT C
:RETURN
IS 1360 GRAPHICS 0:SETCOLOR 2,0,0:SETCOLO
R 1,0,10
OD 1370 ? " HOW TO USE THE ANADIG CLO
CK "
FX 1380 ? : ? "Setting the time:"
ST 1390 ? "Since this clock works off an
internalclock, you will only have to s
et the"
SD 1400 ? "time every time you power up t
he": ? "computer. To set the clock jus
t"
QF 1410 ? "answer the questions. To set
the": ? "clock accurately, enter the se
conds"
XO 1420 ? "question just as that time pas
ses."
ND 1430 ? : ? "Using the alarms:"
TD 1440 ? "You can set up to 10 alarms.
To set alarms, press the space-bar."
WZ 1450 ? : ? "Quitting the program:"
UX 1460 ? "To quit the clock just press E
SC."
DN 1470 ? : ? :RETURN
YW 1480 GRAPHICS 24:SETCOLOR 2,0,4:SETCOL
OR 4,0,6:COLOR 1:RETURN
TV 30000 RESTORE 30010:FOR X=1536 TO 1627
:READ Y:POKE X,Y:NEXT X:RETURN
PW 30010 DATA 160,0,24,173,48,2,105,5,133
,203,173,49,2,105,0,133,204,165,203,10
5,96,133,205,165,204,105,0,133
QS 30020 DATA 206,173,49,2,56,233,31,141,
230,2,165,89,209,203,208,7,173,230,2,1
33,89,208,13,170,177,203,133
NW 30030 DATA 89,138,145,203,24,105,15,14
5,205,165,89,141,80,6,165,88,141,79,6,
169,0,162,30,153,80,127,200
WF 30040 DATA 208,250,238,80,6,202,208,24
4,104,96

```

LISTING 8: BASIC

```
ZK 10 GRAPHICS 0:GOSUB 30000:SETCOLOR 2,0
,0:SETCOLOR 1,0,8
ZI 20 ? " DEMONSTRATOR FOR FRAME GENERAT
ION":? :? "To use program, use joystick
k to move"
BY 30 ? "graphics cursor around. Press t
he":? "fire button to mark points for
the"
SJ 40 ? "program to connect. When you ar
e":? "done, press START. Now, pressin
g the"
EE 50 ? "joystick UP and DOWN will shrink
and":? "expand the picture. Moving t
he"
YZ 60 ? "joystick left and right will mov
e the"
VD 70 ? "picture left and right. Pressin
g the":? "fire button will rotate the
picture":? "10 degrees."
IC 80 ? :? "If you try to move the pictur
e too":? "far left or right or a rotat
ion moves"
IU 90 ? "part of the picture off the scre
en,":? "the picture automatically shri
nks to":? "fit!"
UF 100 OPEN #3,4,0,"K":? :? "Press RETURN
to begin.":GET #3,A:CLOSE #3
AF 110 CX=160:CY=96:C10=C05(0.17453):S10=
SIN(0.17453)
ZT 120 GRAPHICS 24:COLOR 1:SETCOLOR 2,0,0
AD 130 FOR X=16 TO 319 STEP 16:PLOT X,CY:
DRAWTO X+1,CY:NEXT X:FOR Y=12 TO 191 S
TEP 12:PLOT CX,Y:DRAWTO CX+1,Y:NEXT Y
NV 140 DIM A(100),B(100):P=1:X=0:Y=0
JR 150 FOR T=0 TO 1:LOCATE CX+X,CY+Y,Z:CO
LOR (Z=0):PLOT CX+X,CY+Y:NEXT T
PY 160 A=15-5TICK(0):IF A>7 THEN A=A-8:IF
CX+X<318 THEN X=X+1
RN 170 IF A>3 THEN A=A-4:IF CX+X>1 THEN X
=X-1
RR 180 IF A>1 THEN A=A-2:IF CY+Y<190 THEN
Y=Y+1
TG 190 IF A>0 THEN A=A-1:IF CY+Y>1 THEN Y
=Y-1
SU 200 IF STRIG(0)=0 AND P=1 THEN A(1)=X:
B(1)=Y:COLOR 1:PLOT X+CX-1,Y+CY:P=P+1
CC 210 IF STRIG(0)=0 AND P>1 THEN A(P)=X:
B(P)=Y:COLOR 1:PLOT X+CX,Y+CY:DRAWTO A
(P-1)+CX,B(P-1)+CY:P=P+1
UM 220 IF STRIG(0)=0 THEN 220
GJ 230 IF PEEK(53279)=7 THEN 150
ZG 240 IF P<3 THEN ? "You must draw at le
ast 1 line!":END
AE 250 P=P-1:COLOR 1:W=1:S=0
KJ 260 TRAP 350:PLOT S+A(1)*W+CX,B(1)*W+C
Y
NN 270 FOR X=2 TO P:DRAWTO S+A(X)*W+CX,B(
X)*W+CY:NEXT X
DI 280 X=USR(1536)
NU 290 IF STICK(0)=13 THEN W=W+0.1:GOTO 2
60
AJ 300 IF STICK(0)=11 AND S>6-CX THEN S=S-
5:GOTO 260
RV 310 IF STICK(0)=7 AND S<CX-6 THEN S=S+
5:GOTO 260
ID 320 IF STICK(0)=14 AND W>0 THEN W=W-0.
1:GOTO 260
OF 330 IF STRIG(0)=0 THEN FOR X=1 TO P:A=
A(X):B=B(X):A(X)=A*C10-B*S10:B(X)=A*S1
0+B*C10:NEXT X:GOTO 260
PV 340 GOTO 290
FL 350 W=W-0.1:GOTO 260
TV 30000 RESTORE 30010:FOR X=1536 TO 1627
:READ Y:POKE X,Y:NEXT X:RETURN
PW 30010 DATA 160,0,24,173,48,2,105,5,133
,203,173,49,2,105,0,133,204,165,203,10
5,96,133,205,165,204,105,0,133
QS 30020 DATA 206,173,49,2,56,233,31,141,
230,2,165,89,209,203,208,7,173,230,2,1
33,89,208,13,170,177,203,133
WM 30030 DATA 89,138,145,203,24,105,15,14
```

```
5,205,165,89,141,80,6,165,88,141,79,6,
169,0,162,30,153,80,127,200
WF 30040 DATA 208,250,238,80,6,202,208,24
4,104,96
```

LISTING 9: ASSEMBLY

```
10 ;Page flipping routine. After using
20 ;GR.24, GR.9-11, call this routine
30 ;with X=USR(1664) to copy backgrnd
40 ;to foregrnd page. Backgrnd page
50 ;is the one PLOT and DRAWTO
60 ;effect, and foregrnd is the one
70 ;displayed.
80 ;To clear the whole background
90 ;screen, just do an X=USR(1715)
0100 ;
0110 * = $0680
0120 MEMTOP = $02E5 Mem Top pointer.
0130 SAUMSC = $58 Video Mem pointer
0140 SDLSTL = $0230 Pointer to DL
0150 LDH SDLSTL+1 Find an
0160 INH area
0170 STX FORPNT+1 to store
0180 TXA background
0190 SEC screen
0200 SBC #32 and set
0210 STA MEMTOP+1 up the
0220 STA SAUMSC+1 indexed
0230 STA BAKPNT+1 addressing
0240 LDA SAUMSC commands
0250 STA BAKPNT for
0260 STA FORPNT copying.
0270 LDY #0 Copy
0280 LDH #30 the
0290 LOOP LDA 0,Y 7680
0300 STA 0,Y byte
0310 INY background
0320 BNE LOOP buffer
0330 INC BAKPNT+1 to
0340 INC FORPNT+1 the
0350 DEX foreground
0360 BNE LOOP screen.
0370 PLA Unused parameter
0380 RTS Return.
0390 CLS LDA SDLSTL+1 Make sure
0400 SEC there's a
0410 SBC #31 background
0420 STA MEMTOP+1 screen.
0430 STA CLSPNT+1 Set
0440 LDA SAUMSC indexed
0450 STA CLSPNT addressing.
0460 LDA #0 Fill
0470 LDH #30 the
0480 LDY #0 7680
0490 LOOP2 STA 0,Y byte
0500 INY background
0510 BNE LOOP2 buffer
0520 INC CLSPNT+1 with
0530 DEX zeroes
0540 BNE LOOP2 (clear).
0550 PLA Pull unused argument.
0560 RTS Return.
0570 CLSPNT = LOOP2+1
0580 BAKPNT = LOOP+1
0590 FORPNT = LOOP+4
0600 .END
```

LISTING 10: BASIC

```
TN 31000 RESTORE 31010:FOR X=1664 TO 1751
:READ Y:POKE X,Y:NEXT X:RETURN
BF 31010 DATA 174,49,2,232,142,164,6,138,
56,233,32,141,230,2,133,89,141,161,6,1
65,88,141,160,6,141,163,6
ZY 31020 DATA 160,0,162,30,185,80,127,153
,80,159,200,208,247,238,161,6,238,164,
6,202,208,238,104,96,173,49
EA 31030 DATA 2,56,233,31,141,230,2,141,2
04,6,165,88,141,203,6,169,0,162,30,160
,0,153,80,127,200,208,250
UY 31040 DATA 238,204,6,202,208,244,104,9
6
```

BASIC Editor II

by Clayton Walnum



BASIC Editor II is a utility to help you enter BASIC program listings published in *ANALOG Computing*. To simplify the identification of errors, each program line is evaluated immediately after it's typed, eliminating the need for cumbersome checksum listings. When you've finished entering a program using *BASIC Editor II*, you can be certain it contains no typos.

An option is provided for those who wish to use standard BASIC abbreviations. Also, the program retains all Atari editing features. Finally, for those who prefer to type programs the conventional way, using the built-in editor, a post-processing mode is available. It allows you to check typing after the entire listing has been entered.

Typing in the Editor

To create your copy of *BASIC Editor II*, follow the instructions below— exactly.

Disk version

- (1) Type in Listing 1, then verify your work with *Unicheck* (see Issue 39).
- (2) Save the program to disk with the command *SAVE "D:EDITORLI.BAS"*.
- (3) Clear the computer's memory with the command *NEW*.
- (4) Type in Listing 2, then verify your work with *Unicheck*.
- (5) Run the program (after saving a

backup copy) and follow all the on-screen prompts. A data file will be written to your disk.

(6) Load Listing 1 with the command *LOAD "EDITOR1.BAS"*.

(7) Merge the file created by Listing 2 with the command *ENTER "D:ML.DAT"*.

(8) Save the resultant program with the command *LIST "DEDITOR1.LST"*.

Cassette version

(1) Type in Listing 1 and verify your work with Unichack.

(2) Save the program to cassette with the command *CSAVE*. (Do not rewind the cassette.)

(3) Clear the computer's memory with the command *NEW*.

(4) Type in Listing 2 and verify your work with Unichack.

(5) Run the program and follow the on-screen prompts. A data file will be written to your cassette.

(6) Rewind the cassette.

If the program you're entering is particularly long, you may need to take a break.

(7) Load Listing 1 with the command *CLOAD*.

(8) Merge the file created by Listing 2 with the command *ENTER "C:"*.

(9) On a new cassette, save the resultant program with the command *LIST "C:"*.

Using the Editor

Take a look at one of the BASIC program listings in this issue. Notice that each program line is preceded by a two-letter code. This code is the checksum for that line; it's not a part of the program.

To enter a program listing from the magazine, load BASIC Editor II with the *ENTER* command, and run it. You'll be asked if you wish to allow abbreviations (see your BASIC manual). If you do, type *Y* and press *RETURN*. Otherwise, type *N*.

Note: If you set BASIC Editor II to allow abbreviations, the program will run slightly slower.

Your screen will now be divided into two "windows." The upper window will display each line after it's processed, as well as the checksum generated for that line. The lower window is where program lines are typed and edited.

When the program's waiting for input, the cursor will appear at the left margin of the typing window. Type a program line and press *RETURN*. The line will be evaluated and reprinted in the message window, along with the checksum generated.

If the checksum matches the one in the magazine, then go on to the next program line. Otherwise, enter the command *E* (edit) and press *RETURN*. The line you just typed will appear in the typing window, where you may edit it. When you think the line has been corrected, press *RETURN*, and it'll be reevaluated.

Note: You may call up any line previously typed, with the command *E* followed by the number of the line you wish to edit. For example, *E230* will print Line 230 in the typing window. *Do not attempt to edit any program lines numbered 32600 and higher.* These lines fall within the BASIC Editor II program.

If you're using BASIC abbreviations, the two versions of the command *E* work slightly differently. The command *E*, without a line number, will call up the line exactly as you typed it. When you append the line number, the line will be printed in its expanded (unabbreviated) form.

Leaving the Editor

You may leave BASIC Editor II at any time, by entering either *B* (BASIC) or *Q* (quit). If you type *B*, the Editor will return you to BASIC. Enter *LIST* to review your work, if you wish. Note that lines 32600 and above are the Editor program. Your work will appear before these lines. To return to the Editor, type *GOTO 32600*.

Type *Q* and you'll be asked if you really want to quit. If you type *Y*, the Editor program will be erased from memory, and you may then save your work in any manner you like. If you type *N*, the *Q* command will be aborted.

Large listings

If the program you're entering is particularly long, you may need to take a break. When you want to stop, type *Q* and press *RETURN*, then save your work to disk or cassette. When you're ready to start again, load the program you were working on, then load BASIC Editor II with the *ENTER* command. Type *GOTO 32600*, and you're back in business.

The post-processor

Many people may not want to use BASIC Editor II when entering a program listing, preferring, instead, the Atari's built-in editor. For that reason, BASIC Editor II will allow you to check and edit your programs after they've been typed.

To take advantage of this option, type any magazine program in the conventional manner, then save a copy to disk or cassette (just in case). With your typed-in program still in memory, load BASIC Editor II with the *ENTER* command, then type *GOTO 32600*.

Respond with *N* to the "abbreviations" prompt. When the Editor appears on your screen, enter the command *P* (post-process), and the first program line will appear in the typing window. Press *RETURN* to enter it into the Editor.

The line will be processed, and the checksum, along with the program line, will be printed in the upper window. If the checksum matches the one in the magazine, press *RETURN* twice, and the next line will be processed.

If you find you must edit a line, enter the command *E*, and the line will be moved back to the typing window for editing.

When the entire listing has been checked, you'll be asked if you wish to quit. Type *Y* and press *RETURN*. The Editor program will be removed from memory, and you may then save the edited program in any manner you wish.

Murphy's Law

Anyone who's been associated with computing knows this is the industry Murphy had in mind. You may find that, after typing a program with BASIC Editor II, it still won't run properly. There are two likely causes for this.

First, it may be that you're not following the program's instructions properly. Always read the article accompanying a program *before* attempting to run it. Failure to do so may present you with upsetting results.

Finally, though you can trust BASIC Editor II to catch your typos, it can't tell you if you've skipped some lines entirely. If your program won't run, make sure you've typed all of it. Missing program lines are guaranteed trouble.

One last word: Some people find it an unnecessary and nasty chore to type *REM* lines. I don't condone the omission of these lines, since they may be referenced within the program (a bad practice, but not unheard of). If you want to take chances, BASIC Editor II is willing to comply.

CRISIS CENTER

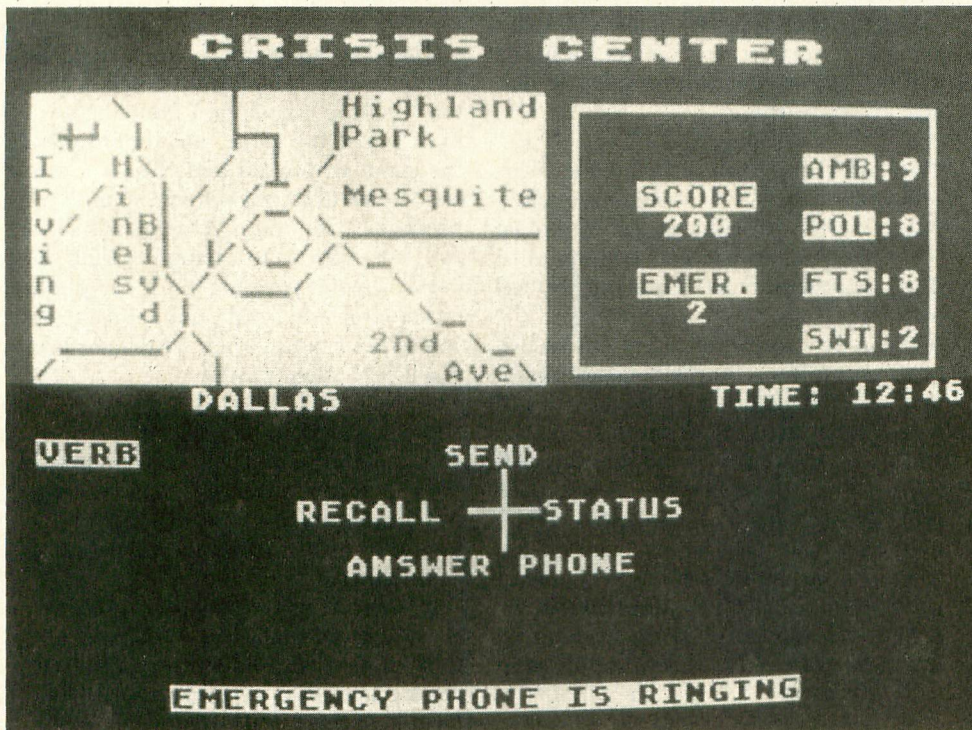
by Joe McManus



You've passed all the security procedures and are now standing in Crisis Command Headquarters. The 30-yard long electro-map seems uncommonly serene. As you relieve the Watch Commander, you can't help noticing his haggard face and bloodshot eyes. You log on, and the display generates your equipment status and an overview of the city.

If you have the right combination and number of vehicles to handle the situation, the people will be saved. On the other hand, if you don't get the right combination there in time....

HINT: Up to four emergencies may be ongoing at the same time.



Looks good! You have your full complement of emergency vehicles. Suddenly, an alarm rings and a message flashes on the screen...

"Sniper on roof at Theater Center!"

Crisis Center is a 54-minute real-time simulation that takes place in one of three cities. The object of the game is to resolve emergencies that arise by sending the appropriate equipment needed to handle the situation.

For example, there may be a hotel burning somewhere. You have ambulances, fire trucks, police cars and SWAT teams at your disposal. Crowd control could become a real problem; so you had better send some police cars along with the fire

trucks. And, of course, you'll need some ambulances.

If you have sent the right combination and number of vehicles to handle the situation, the people will be saved. On the other hand, if you don't get the right combination there in time...well, you lose points for losing lives! If you lose enough points, you'll be fired and the game will end. The more serious emergencies are worth more points. Therefore, you should try to resolve the most critical situations before handling the lesser emergencies.

Playing the game

After pressing START, you'll be asked to choose a city. Each city has its own level of difficulty based on crime rate and population. Dallas is the easiest game, fol-

lowed by L.A. and then New York. When you have made your selection, a map of your city will appear at the top of the screen and the game will begin.

The top section of the screen also contains a status display that is continually updated throughout the game. The status screen displays readouts of how many of each vehicle are currently available at headquarters, an emergency counter which shows how many crisis situations are currently ongoing, your current score and a real-time clock. The game begins at 12:30 p.m. and will end at 1:39 unless your score falls below zero; in which case the simulation is terminated.

You have two communication links at headquarters. The primary communication system at Crisis Command is a half-

CRISIS CENTER

duplex system. To gain control of this communication system you must wait until any message you are receiving is finished and then press the trigger button to obtain sending priority. Once you have communication control, you'll see a command screen. The other communication link is the emergency phone which is explained in the following command section.

Commands

Select one of the four commands by moving the joystick in the designated direction.

SEND: This allows you to dispatch vehicles to crisis locations. After selecting the send command, another stick diagram appears. Choose which type of vehicle to send. Then the locations are given in a stick diagram with the choice "MORE" at the bottom. If the location you wish to send a vehicle does not appear on the menu, select the MORE option until you find it.

You'll then be asked how many units you want to send. Type the number and press RETURN. You'll hear the units leave Command Headquarters.

RECALL: You'll be prompted for which

type of unit you wish to recall and from what location. If you attempt to recall more units from a location than there actually are, no units at all will be returned and you'll have to try again.

STATUS: This command is included in case you forget where your emergency teams are located. For instance, if all your fire trucks are off-station, you must recall them before they can be sent again.

Select which type of vehicle you need a status report on, and all ten locations will be displayed with how many of that unit type are there.

PHONE: Throughout the game the emergency phone may ring. It rings more often in the higher difficulty games. You must select the "answer phone" command to find out what emergency is being phoned in.

If you do not answer the phone in two to three minutes from when the phone first starts ringing, that emergency will have deteriorated to the point of disaster, and an unanswered phone message will appear that describes the result. The value of that particular emergency will be deducted from your score.

The phone emergency could be one of small point value, like a simple traffic accident. On the other hand, it could be worth lots of points; so you should stop

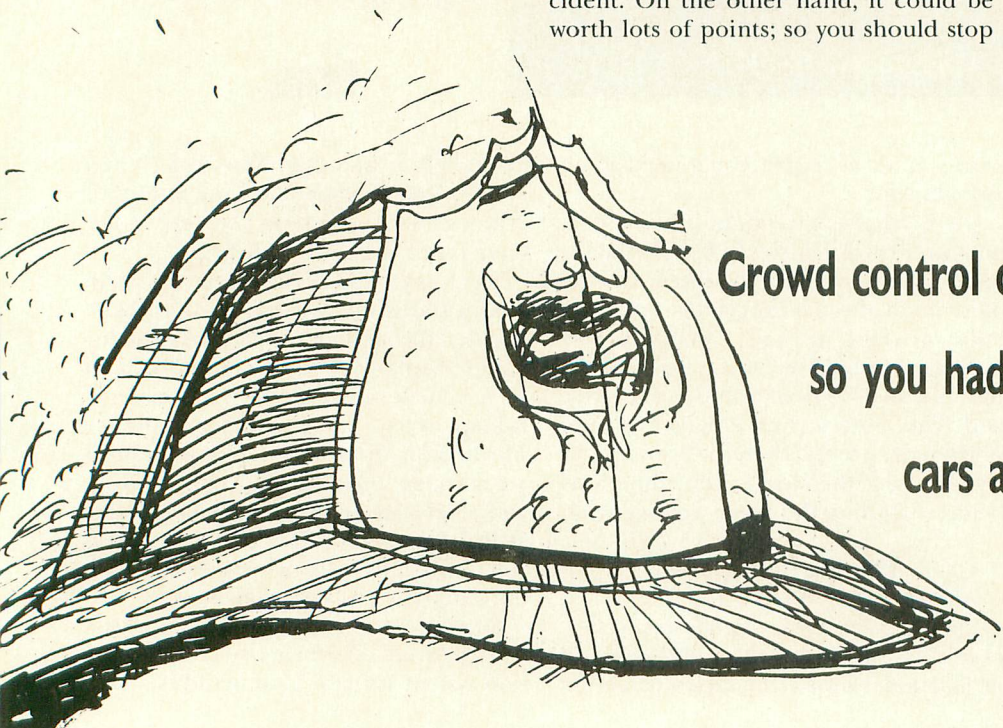
working on your current emergency and answer the phone if you wish to avoid losing points.

You begin the game with 200 points. If your score falls below zero, the game is over and you're fired from your position as Watch Commander. The simulation is designed to be a test of endurance, recall and common sense. The emergencies have been worded to enhance the sense of realism.

There is no pause feature in Crisis Center since you should never leave a watch station unattended. However, if you must pause the game, you should select the status command and choose a vehicle. The status screen will stay up until you press the fire button. Though the clock will not appear to change during the break, it's still running. So if the phone was ringing when you took a break, and you're gone long, when you come back you'll lose some points.

HINT: Up to four emergencies may be ongoing at the same time. One limitation the program has is that it can only solve one emergency at a time, so it only checks for the correct solution after a send command.

If you have two emergencies occurring at the same location and feel you have sent enough equipment to resolve both emergencies, then after you receive points for one of them, send another unit to get the points for the other emergency.



**Crowd control could become a problem;
so you had better send some police
cars along with the fire trucks.**

**And, of course, you'll
need some ambulances.**



LISTING 1: BASIC

```

AA 100 REM *****
NF 110 REM * CRISIS CENTER *
VH 120 REM * BY Joe McManus *
GO 130 REM * for *
EQ 140 REM * ANALOG COMPUTING *
AK 150 REM *****
RA 160 CLR
QH 170 DIM RES$(220),RES1$(220),RES2$(220)
    ),DIS$(220),ANS(44),EQ$(48),PLC(4),EME
    R(4),EQU(4),PLACE(40),W1$(20),W2$(20)
IB 180 DIM TIME$(5),A$(92),B$(240),X$(60)
    ),BLANK$(40),C$(59),I$(59),T$(59),Y$(59)
    ),S$(59),W3$(20),W4$(20),W5$(20)
GD 185 DIM CH$(1):EMS=0
VZ 190 TS=0:C0=0:C1=1:C2=C1+C1:C3=C2+C1:C
    4=C3+C1:C5=C4+C1:C6=C5+C1:C7=C6+C1:C8=
    C7+C1:C9=C8+C1:C10=C9+C1:C11=C10+C1
FA 200 BLANK$=""
    "":R=2400:TIME$="00:00":B
    $(201,240)=BLANK$:DOPLACE=0:GOTO 470
LR 210 X$(C1,20)=BLANK$:X$(21,40)=BLANK$:
    RETURN
AI 220 II=0
ZC 230 II=II+1:IF X$(II,II+1)="" THEN 5
    POT=II+1:RETURN
NG 240 GOTO 230
YE 260 POSITION A,B:BLANK$:BLANK$:GOSU
    B 300:RETURN
OL 270 B=B+C2:IF B>22 THEN FOR I1=C1 TO 6
    00:NEXT I1:B=13:FOR I2=C10+C2 TO C10+C
    8:POSITION C0,I2:BLANK$:NEXT I2
IR 280 GOSUB 300:RETURN
EI 290 SOUND C3,ASC(A$(I,I)),12,C10:FOR D
    =C1 TO 34:NEXT D:SOUND C3,C0,C0,C0:RET
    URN
YZ 300 IF T5<>C1 THEN 460
CN 310 POSITION 33,C3:"AMB:";EQU(C2):PO
    SITION 33,C5:"POL:";EQU(C3):POSITION
    33,C7:"FIS:";EQU(C1)
PK 320 POSITION 23,1:" "
    :FOR II=2 TO 9:POSITION 23,II:"|":P
    OSITION 38,II:"|":NEXT II
VN 330 POSITION 23,10:" "
QG 335 EMS=0:FOR DD=1 TO 4:IF EMER(DD)<>0
    THEN EMS=EMS+1
UH 336 NEXT DD:POSITION 26,C7:"EMER:";
    POSITION 28,C8:EMS
YW 340 POSITION 33,C9:"SWT:";EQU(C4):IF
    SCORE<0 THEN POSITION C4,C0:"ALL"YO
    U ARE FIRED!":POKE 709,C0:GOTO 250
BO 350 IF PEEK(18)=C3 THEN POSITION C4,C0
    :?"ALL" GAME OVER":GOTO 250
HN 360 TIME=(PEEK(20)+PEEK(19))*256+PEEK(1
    8)*256*256/60:POSITION 27,C5:" "
    "":POSITION 26,C4:"SCORE"
DO 370 MIN=INT((TIME/3600-INT(TIME/3600))
    *60)+INT(ASC(TIME$(4,4))-48)*10+INT(AS
    C(TIME$(C5,C5))-48)*1
NL 375 POSITION 27,C5:SCORE
MM 380 HRS=INT((TIME/216000-INT(TIME/2160
    00))*60)+INT(ASC(TIME$(C1,C1))-48)*C10
    +INT(ASC(TIME$(C2,C2))-48)*C1
JZ 385 IF PHONE=1 THEN GOSUB 2255

```

```

BO 390 IF MIN>59 THEN HRS=HRS+C1:MIN=MIN-
    60
HE 400 IF HRS>12 THEN HRS=HRS-12
WZ 410 IF HRS<C10 THEN POSITION 35,C11:
    "0";HRS;:GOTO 430
ET 420 POSITION 35,C11:HR5;
AC 430 POSITION 37,C11:"|"
NX 440 IF MIN<C10 THEN POSITION 38,C11:
    "0";MIN;:GOTO 460
YA 450 POSITION 38,C11:MIN;
ZM 460 RETURN
WN 470 GRAPHICS C0:POKE 752,C1:POKE 710,4
    8:POKE 712,48:POKE 709,54:A=C2:DL=PEEK
    (560)+PEEK(561)*256+C4:POKE DL-C1,70
QP 480 POKE DL+12,130
TD 490 POKE DL+C2,C6:POKE DL+13,130:POKE
    DL+16,130:FOR I=C0 TO 23:READ A:POKE 1
    686+I,A:POKE 1710+I,A:POKE 1734+I,A
HB 495 NEXT I
OK 500 POKE 1688,2
ZL 510 POKE 1747,150:POKE 1699,174:POKE 5
    12,150:POKE 513,C6:POKE 54286,192:EQU(
    C1)=C8:EQU(C2)=C9:EQU(C3)=C8
VQ 520 EQU(C4)=C2:PHONE=0:GOSUB 580:POSIT
    ION 9,14:"PRESS START TO BEGIN"
AL 530 IF (PEEK(53279)<>C6) AND (STRIG(0)
    <>0) THEN 530
AI 540 POSITION 9,14:" "
KR 545 GOTO 2000
YY 548 B=23:GOSUB 270
FO 550 TIME$="12:45":GOSUB 260:POKE 20,C0
    :POKE 19,C0:POKE 18,C0:FOR I=C1 TO 240
    STEP 40:B$(I,I+39)=BLANK$:NEXT I
SH 560 B=21:GOSUB 270:RESTORE WHERE:FOR I
    =C1 TO 220 STEP 20:READ X$:B$(I,I+19)=
    X$:NEXT I:READ R,DF
KR 570 TS=C0:GOSUB 580:GOTO 750:DATA 72,1
    69,160,141,10,212,141,24,208,141,26,20
    8,169,198,141,0,2,169,6,141,1,2,104,64
UJ 580 RESTORE R:READ C$,I$,T$,Y$,S$
XP 590 A$="CRISIS CENTER"EMERGENCY PHONE I
    S RINGING HOW MANYINCOMING MESSAG
    EYOU CAN'T DO THAT!UNITS"
PH 600 POSITION C4,C0:FOR I=C1 TO 13:AS
    (I,I):GOSUB 290:NEXT I:B=C1:A=C2
TS 610 POSITION A,B:CS:GOSUB 270:POSITI
    ON A,B:IS:GOSUB 270:POSITION A,B:Y$
    :GOSUB 270:POSITION A,B:Y$
WL 615 GOSUB 270:POSITION A,B:S$:GOSUB
    270
XD 620 POSITION 0,B:BLANK$:POSITION C8
    ,B:B$(201,219):POSITION 29,B:"TIM
    E":TS=C1:A=C2:GOSUB 270:RETURN
KI 750 RESTORE 780:B=21:GOSUB 270:C$="May
    or says do somethingHURRY COMMANDER!"
GL 760 FOR I=C1 TO 220 STEP 20:DIS$(I,I+
    9)=BLANK$(C1,20):NEXT I:RES$=DIS$:RES1
    $=DIS$:RES2$=DIS$
MD 770 FOR I=C1 TO 220 STEP 20:READ X$:DI
    S$(I,I+19)=X$:NEXT I
XV 780 DATA There is a FIRE at,Traffic ac
    cident at,4-ALARM FIRE is at,BANK ROBB
    ERY at,A HEART ATTACK at,HIT & RUN at
    790 DATA SNIPER on roof at,HOTEL burni
    ng at,10 CAR pileup at,AIRPLANE on fir

```

```

e at,HIJACK report from
MO 800 EQ$="FIRETRUCKS AMBULANCES POLIC
ECARS SWATTEAMS "
TE 810 FOR I=1 TO 220 STEP 20:READ X$:RES
$(I,I+19)=X$:NEXT I
UM 820 DATA The fire is out at,Accident c
leared at,The fire is out at,Cops stop
holdup at,Victim was saved at
WE 830 DATA Victim was saved at,Sniper ca
ught at,Hotel safe at,Pileup cleared a
t,Airplane saved at,Hijacker caught at
GC 840 FOR I=1 TO 220 STEP 20:READ X$:RES
1$(I,I+19)=X$:NEXT I
NZ 850 DATA Fire killed 4 at,2 victims di
ed at,8 burned to death at,$8000.00 st
olen at,Elderly man died at
DJ 860 DATA The victim died at,7DEAD Snip
er escaped,Hotel destroyed at,70 DEAD
FOUL-UP at,Airplane is ashes at
PQ 870 DATA Plane in CUBA not
HX 880 FOR I=1 TO 220 STEP 20:READ X$:RES
2$(I,I+19)=X$:NEXT I
FS 890 DATA Fire is burning at,Traffic sn
arled at,Fire is raging at,Cops dodgin
g lead at,Victim going fast at
ZS 900 DATA Weak-pulse report at,Sniper g
ot one at,People screaming at,More car
s wrecked at,Fuel may explode at
LQ 910 FOR I=C1 TO 40:PLACE(I)=0:NEXT I:F
OR I=C1 TO C4:PLC(I)=0:EMER(I)=0:NEXT
I:SCORE=200:DATA Hostages scared at
WO 920 FOR I=C1 TO C4:READ AA:EQU(I)=AA:N
EXT I:FOR I=C5 TO 48:READ AA:ANS(I-C4)
=AA:NEXT I
CN 930 DATA 8,9,8,2,3,2,1,0,0,1,2,0,4,2,2
,0,0,0,3,0,0,1,0,0,0,1,1,0,0,2,3,1,4,3
,3,0,2,4,4,0,3,2,2,0,0,2,3,2
HT 940 DATA VERB,SEND,RECALL,STATUS,ANSWE
R PHONE,EQUIPMENT,AMBULANCE(S),POLICEC
AR(S),SWAT TEAMS(S),FIRETRUCK(S),PLACE
UN 950 DATA University Park,Highland Park
,Sunnyvale,More,PLACE,Mesquite,Hines B
ld.,Second Ave.,More
AT 960 X$(21,21)=" ":DATA PLACE,Hensley F
ield,Six Flags Park,Theatre Center,Cot
ton Bowl Stadium
WD 970 IF STRIG(0)=0 THEN 1060
DU 975 IF PHONE=C2 THEN GOSUB 2265:GOTO 1
000
MK 980 R=INT(RND(0)*800):IF R<DIF THEN 10
00
SH 990 GOTO 970
QJ 1000 FOR I=C1 TO C4:IF EMER(I)=0 THEN
1020
NJ 1010 NEXT I:GOTO 1610
CR 1020 R=INT(RND(0)*C11)+C1:IF R>C9 THE
N R1=C7:GOTO 1040
FC 1030 R1=INT(RND(0)*C10)+C1:IF R1=C7 T
HEN 1030
XH 1040 IF PHONE>C1 THEN 2275
FU 1045 GOSUB 210:X$(C1,20)=DI5$(R*20-19,
R*20):II=0:GOSUB 220
VH 1050 X$(SPOT,SPOT+20)=B$(R1*20-19,R1*2
0):E=I:EMER(E)=R:PLC(E)=R1:GOSUB 1380:
GOSUB 1330:GOTO 970
NX 1060 R=INT(RND(0)*200):IF R<DIF OR PHO
NE=1 THEN GOSUB 2200
MQ 1065 B=21:GOSUB 270:RESTORE 940:C=C0:G
OSUB 1240:IF C=C0 THEN B=21:GOSUB 270:
GOTO 970
EU 1070 IF C<13 AND C>C8 THEN V=C2:GOTO 1
110
PQ 1080 IF C=13 THEN 1400
MJ 1090 IF C=14 THEN V=C1:GOTO 1110
YE 1100 IF C<C8 THEN V=C3
PW 1110 GOSUB 1250:CT=C1:C=C0:GOSUB 1240:
IF C=C0 THEN B=21:GOSUB 270:GOTO 970
JK 1120 IF C<13 AND C>C8 THEN EQ=C3
EE 1130 IF C<C8 THEN EQ=C4
SC 1140 IF C=13 THEN EQ=C1
TO 1150 IF C=14 THEN EQ=C2
UF 1160 IF V<>C3 THEN CT=C0:DOPLACE=1:GOT
O 1435
LJ 1170 A=12
ZH 1180 I=9:B=21:GOSUB 270:POSITION A,B:F
OR J=0 TO 11:? EQ$(EQ*12-11+J,EQ*12-11

```

```

+J):GOSUB 290:NEXT J:? " Status"
IN 1185 B=B+2
IR 1190 A=1:FOR I=C0 TO C8 STEP 2:POSITIO
N A,B:? B$(I+1)*20-19,(I+1)*20-4);PLA
CE((I+1)*4)-4+EQ):POSITION 21,B
WF 1200 ? B$(I+2)*20-19,(I+2)*20-4);" ";
PLACE((I+2)*4)-4+EQ);B=B+1:NEXT I
DE 1210 ? :? " Press trigger when rea
[0]"
Q5 1220 IF STRIG(0)<>0 THEN 1220
JY 1230 B=21:GOSUB 270:FOR I=18 TO 22:POS
ITION 0,I:? BLANK$;:NEXT I:GOTO 970
ZW 1240 IF NOT DOPLACE THEN READ W1$,W2$
,W3$,W4$,W5$:CT=CT+C1:GOTO 1270
XY 1245 W1$="PLACE":READ W2$,W3$,W4$:IF C
T=C2 THEN READ W5$:GOTO 1247
YU 1246 W5$="MORE"
JF 1247 CT=CT+C1:GOTO 1270
HW 1250 W1$(C1,LEN(W1$))=BLANK$:W2$(C1,LE
N(W2$))=BLANK$:W3$(C1,LEN(W3$))=BLANK$
:W4$(C1,LEN(W4$))=BLANK$
IF 1260 W5$(C1,LEN(W5$))=BLANK$
Z5 1270 POSITION A,B:? W1$:POSITION A+18-
(LEN(W2$)/C2),B:? W2$;:POSITION A+18,B
+C1:? "|":POSITION A+16-LEN(W3$),B+C2
DU 1275 ? W3$
NR 1280 POSITION A+17,B+2:? "+":POSITIO
N A+20,B+2:? W4$;:POSITION A+18,B+3:?
"||":POSITION A+18-(LEN(W5$)/2),B+4
EQ 1285 ? W5$
DU 1290 IF W1$(C1,C1)=" " THEN RETURN
WJ 1300 FOR I=250 TO 30 STEP -20:FOR J=15
TO C0 STEP -C1:SOUND C2,I,C10,J:NEXT
J
WY 1310 FOR DE=C1 TO I:IF STICK(0)<>15 TH
EN C=STICK(0):RETURN
DA 1320 NEXT DE:NEXT I:RETURN
FX 1330 GOSUB 260:POSITION A,B:FOR I=C1 T
O LEN(X$):? X$(I,I);:GOSUB 290:NEXT I:
RETURN
IT 1340 GOSUB 260:POSITION A,B:FOR I=70 T
O 87:? A$(I,I);:GOSUB 290:NEXT I:B=21:
GOSUB 270:RETURN
NH 1350 B=21:GOSUB 270:POSITION A,B:FOR I
=46 TO 53:? A$(I,I);:GOSUB 290:NEXT I:
RETURN
LB 1360 FOR J=C0 TO C3:SCORE=SCORE+(ANS(E
MER(E)*C4-C3+J)*15):NEXT J:RETURN
UG 1370 FOR J=C0 TO C3:SCORE=SCORE-(ANS(E
MER(E)*4-3+J)*15):NEXT J:RETURN
UG 1380 B=21:GOSUB 270:FOR I=C1 TO C3:POK
E 1712,242:POSITION A+11,B:? A$(54,69)
;"[33]";:POSITION A,B:? BLANK$;
RZ 1390 POKE 1712,160:FOR J=C1 TO 20:NEXT
J:NEXT I:RETURN
DJ 1400 IF PHONE>0 THEN POSITION 0,22:? B
LANK$;:B=21:GOSUB 270:? "ANSWERING PHO
NE...":PHONE=3:GOTO 1000
RF 1410 B=21:GOSUB 270:POSITION 9,19:? "N
O PHONE WAS RINGING":GOSUB 1950:POSITI
ON 0,19:? BLANK$;:GOTO 970
VJ 1435 RESTORE WHERE
LA 1440 GOSUB 1250:C=C0:GOSUB 1240:IF C=C
0 THEN B=21:GOSUB 270:GOTO 970
XG 1450 IF C=13 AND CT<C3 THEN 1440
ZR 1460 IF C=14 THEN PLC1=CT*C3-C2
WR 1470 IF C<13 AND C>C8 THEN PLC1=CT*C3-
C1
DW 1480 IF C<C8 THEN PLC1=CT*C3
ZZ 1490 IF C=13 AND CT=C3 THEN PLC1=10
IC 1500 DOPLACE=0:TRAP 1500:GOSUB 1350:IN
PUT C:TRAP 40000:IF V=1 THEN 1540
IF 1510 IF C>PLACE(PLC1*4-4+EQ) THEN GOSU
B 1340:GOTO 970
ZB 1520 PLACE(PLC1*4-4+EQ)=PLACE(PLC1*4-4
+EQ)-C:EQU(EQ)=EQU(EQ)+C:R=C1:IF C=C1
THEN R=C0
NM 1530 GOSUB 260:POSITION A+C10,B:FOR I=
88 TO 91+R:? A$(I,I);:GOSUB 290:NEXT I
:? " RETURNED";:B=21:GOSUB 270
UV 1535 GOTO 970
ID 1540 IF EQU(EQ)-C<C0 THEN GOSUB 1340:G
OTO 970
VF 1550 EQU(EQ)=EQU(EQ)-C:PLACE(PLC1*4-4+
EQ)=PLACE(PLC1*4-4+EQ)+C:FOR I=C1 TO C
4:IF PLC(I)=PLC1 THEN E=I:GOTO 1570

```

```

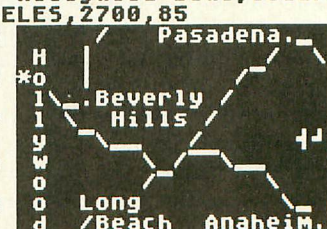
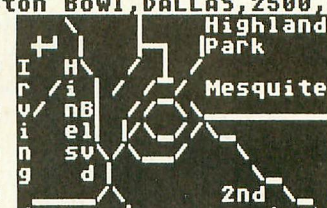
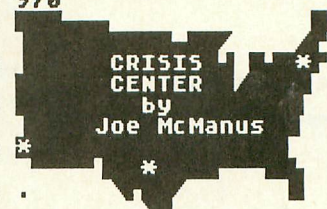
PB 1560 NEXT I:GOTO 1710
RU 1570 FOR J=C0 TO C3:IF PLACE(PLC1*4-3+
J)<AN5(EMER(E)*4-3+J) THEN 1710
VM 1580 NEXT J:GOSUB 1780:GOSUB 210:X$(C1
,20)=RES$(EMER(E)*20-19,EMER(E)*20):II
=0:GOSUB 220
YK 1590 X$(SPOT,SPOT+20)=B$(PLC1*20-19,PL
C1*20):GOSUB 1360
PD 1600 EMER(E)=C0:PLC(I)=C0:B=21:GOSUB 2
70:GOSUB 1330:GOTO 970
VO 1610 B=21:GOSUB 210:GOSUB 270:GOSUB 13
80:E=INT(RND(0)*4)+1:IF RND(C0)<0.5 TH
EN 1670
NC 1620 X$(C1,20)=RES2$(EMER(E)*20-19,EME
R(E)*20):II=0:GOSUB 220
IE 1630 X$(SPOT,SPOT+20)=B$(PLC(E)*20-19,
PLC(E)*20)
SH 1640 IF RND(C0)<0.5 THEN S=24:G=39:GOT
0 1660
XV 1650 S=C1:G=23
TL 1660 GOSUB 260:POSITION A,B:FOR I=5 TO
G:G? C$(I,I):GOSUB 290:NEXT I:GOTO 17
00
MY 1670 X$(C1,20)=RES1$(EMER(E)*20-19,EME
R(E)*20):II=0:GOSUB 220
YO 1680 X$(SPOT,SPOT+20)=B$(PLC(E)*20-19,
PLC(E)*20):GOSUB 1370
OY 1690 EMER(E)=C0:PLC(E)=C0
IY 1700 GOSUB 1330:GOSUB 270:GOTO 970
HE 1710 VOL=12:ON EQ GOTO 1720,1740,1760,
1770
LC 1720 FOR I=C1 TO C4:FOR J=70 TO 15 STE
P -C4:SOUND C1,J+20,14,VOL:SOUND C2,J,
C10,VOL:NEXT J
NI 1730 FOR J=15 TO 70:SOUND C1,J+20,14,V
OL:SOUND C2,J,170,VOL:NEXT J:VOL=VOL-C
4:NEXT I:GOTO 970
FC 1740 FOR I=C1 TO C4:FOR J=40 TO 20 STE
P -C4:SOUND C1,J+C8,14,VOL:SOUND C2,J,
C10,VOL:NEXT J
MT 1750 FOR J=20 TO 40:SOUND C1,J+C8,14,V
OL:SOUND C2,J,C10,VOL:NEXT J:VOL=VOL-C
4:NEXT I:GOTO 970
MA 1760 FOR I=C1 TO C4:SOUND C1,110,14,VO
L:FOR D=C1 TO 45:NEXT D:SOUND C1,65,C1
0,VOL:FOR D=C1 TO 45:NEXT D:VOL=VOL-C4
HZ 1765 NEXT I:GOTO 970
QR 1770 FOR I=C1 TO C5:FOR J=80 TO 50 STE
P -C2:SOUND C1,220,12,VOL:SOUND C2,J,C
10,VOL:NEXT J:VOL=VOL-C3:NEXT I
VL 1775 GOTO 970
HW 1780 FOR DD=1 TO 3:SOUND 1,96,10,10:FO
R J=1 TO 6:NEXT J:FOR J=1 TO 6:SOUND 1
,0,0:NEXT J:NEXT DD
RT 1790 SOUND 1,81,10,10:SOUND 2,53,10,8:
SOUND 0,64,10,8:FOR J=1 TO 35:NEXT J:IF
OR J=0 TO 2:SOUND J,0,0,0:NEXT J
AP 1800 RETURN
YL 1900 OPEN #2,4,0,"K":GET #2,CH
VJ 1910 CH$=CHR$(CH):CLOSE #2:FOR DD=20 T
O 30:SOUND 1,DD,12,10:NEXT DD:SOUND 1,
0,0,0:RETURN
FM 1950 FOR DD=1 TO 50:SOUND 0,100,12,10:
NEXT DD:SOUND 0,0,0,0:RETURN
LO 2000 REM CHOOSE CITY
TP 2010 POSITION 14,13:?"CHOOSE A CITY"
XW 2015 POSITION 2,15:?"1. DALLAS 2. NE
W YORK 3. LOS ANGELES"
BQ 2020 GOSUB 1900:IF (CH$(1)) OR (CH$(
3)) THEN GOSUB 1950:GOTO 2020
RC 2030 CHOICE=ASC(CH$)-48:ON CHOICE GOTO
2040,2050,2060
YW 2040 WHERE=2480:GOTO 548
ZM 2050 WHERE=2580:GOTO 548
AC 2060 WHERE=2680:GOTO 548
XE 2200 REM EMERGENCY PHONE
GJ 2210 IF PHONE=0 THEN 2215
JZ 2212 TIME=(PEEK(20)+PEEK(19)*256+PEEK(
18)*256*256)/60
YR 2214 PHMIN=INT((TIME/3600-INT(TIME/360
0))*60)+INT(ASC(TIME$(4,4))-48)*10+INT
(ASC(TIME$(5,5))-48)*1
TL 2215 PHONE=1:POSITION 7,22:?"A$(14,42)
OW 2220 FOR JJJ=1 TO 3
BU 2230 FOR DD=0 TO 15:FOR K=1 TO 3:SOUND
2,22+K,10,DD:SOUND 1,50-K,10,15-DD:NE

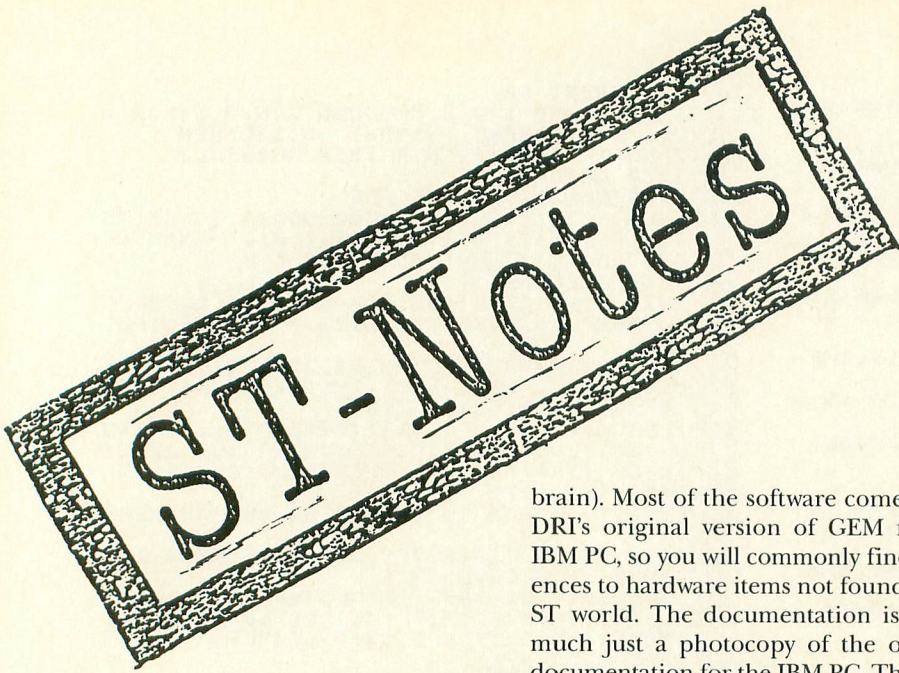
```

```

XT K:NEXT DD
JC 2240 SOUND 1,0,0,0:SOUND 2,0,0,0:FOR J
J=1 TO 10:NEXT JJ:NEXT JJJ:RETURN
RN 2255 IF PHMIN+2<MIN THEN PHONE=C2
BR 2256 RETURN
NP 2260 REM DIDN'T ANSWER
IF 2265 B=21:GOSUB 270:FOR DD=C1 TO C3:PO
KE 1712,242:POSITION A+11,B:?"UNANSWE
RED PHONE!!";POSITION A,B
EQ 2266 ? BLANK$;
SM 2270 POKE 1712,160:FOR J=C1 TO 20:NEXT
J:NEXT DD:POSITION 0,22:?"BLANK$":RET
URN
AN 2275 ON PHONE GOTO 2290,2280,2295
FF 2280 GOSUB 210:X$(C1,20)=RES1$(R*20-19
,R*20):II=0:GOSUB 220
BV 2285 X$(SPOT,SPOT+20)=B$(R1*20-19,R1*2
0):E=I:EMER(E)=R:PLC(E)=R1:GOSUB 1330:
GOSUB 1370:PHONE=0:EMER(E)=0:PLC(E)=0
VN 2287 GOTO 970
BZ 2290 ? "ERROR IN ANSWER OR NOT ANSWER"
:STOP
GO 2295 GOSUB 210:X$(C1,20)=DIS$(R*20-19,
R*20):II=0:GOSUB 220
SS 2300 X$(SPOT,SPOT+20)=B$(R1*20-19,R1*2
0):E=I:EMER(E)=R:PLC(E)=R1:GOSUB 1330:
PHONE=0:POSITION 0,12:?"BLANK$;
UJ 2305 GOTO 970
UR 2400 DATA
SP 2410 DATA
YK 2420 DATA
SR 2430 DATA
XV 2440 DATA
WT 2470 REM *** DALLAS ***
SI 2480 DATA University Park,Highland Par
k,Sunnyvale,Mesquite,Hines Blvd,Second
Ave,Hensley Field
FI 2490 DATA Six Flags Park,Theatre Cente
r,the Cotton Bowl,DALLAS,2500,40
IK 2500 DATA
VV 2510 DATA
SF 2520 DATA
ZC 2530 DATA
ZN 2540 DATA
KO 2570 REM *** NEW YORK ***
WC 2580 DATA Brooklyn,Bronx,Queens,Manhat
tan,Staten Island,Forty Second St.,J.F
.K. Airport,Lincoln Tunnel
ET 2590 DATA Statue of Liberty,United Nat
ions,NEW YORK,2600,120
YJ 2600 DATA
IM 2610 DATA
DQ 2620 DATA
JB 2630 DATA
TZ 2640 DATA
FK 2670 REM *** LOS ANGELES ***
NV 2680 DATA Santa Monica,Culver City,Bev
erly Hills,Hollywood,San Pedro,Pasaden
a,L.A. International,Disneyland
RV 2690 DATA Hollywood Bowl,Security Plaz
a,LOS ANGELES,2700,85
SP 2700 DATA
NA 2710 DATA
PD 2720 DATA
LU 2730 DATA
JT 2740 DATA

```





Developers Bible



When Leonard Tramiel, Atari vice president of Research and Development, was asked at the San Jose Atarifest in 1986 if he had any plans to rework the developers documentation for the Atari ST, his response was a cold, "Yes." And nothing else. Since then, nothing has been done to clean up the poor and sloppy documentation that is sold at \$350 as the official Atari ST developers kit.

The developers kit includes six diskettes with Digital Research's development software for the 68000 (Atari ST's

brain). Most of the software comes from DRI's original version of GEM for the IBM PC, so you will commonly find references to hardware items not found in the ST world. The documentation is pretty much just a photocopy of the original documentation for the IBM PC. There are addendums describing special operating-system programs on the ST that are available to developers; however, everything is scattered over the 500+ pages of documentation.

Other computer manufacturers (Apple and Commodore) have written very complete documentation of their operating systems and licensed the manuals to large publishing companies such as Osborne/McGraw-Hill, Addison-Wesley and Bantam. These large book-publishing companies have terrific distribution to bookstores that carry technical books. At the local mall, you can find the complete listing of the Amiga operating system for just \$19.95.

Atari has never been known for its manual writing abilities, so many outside authors have written developer guide books and reference manuals for the ST. One of the best appears to be *The Concise Atari ST 68000 Programmers Reference Guide* by Katherine Peel. The guide is unique to the ST market because of its complete and thorough description of the Atari ST, inside and out. The table of contents includes chapters on Monitor Output, Printer Interfacing, Floppy Disk I/O, DMA port access, Intelligent Keyboard I/O, Yamaha programmable sound-generator programming, TOS overview, BIOS/BDOS, GEM AES/VDI, System Initialization, etc. All of the topics covered are necessary for development of applications and utilities for the ST.

Much of the *Guide's* information can also be found in other documentation from Atari and from the ST book series written by Databecker of West Germany, published by Abacus here in the U.S. However, the *Guide* pulls it all together

into one large manual that includes a complete table of contents and index. The *Guide* is being marketed through Michtron in the U.S. and Canada. It has a suggested list price of only \$19.95. We highly recommend the *Guide* as a substitute for the Atari developers kit (which by the way still hasn't been revised in the three years since the ST was released.)

Game Boxes

You've just bought a new game for your ST. You rush home, tear off the plastic wrap, shove the disk into your ST's drive and begin playing. That's probably the last time you will see the box and manual.

Have you ever taken the time to read the box?

Take for example this newly released piece of software from a popular games manufacturer:

"All is well in Hyturian until the pirate planet, Nono, stealthily slips through the outer fringes of your screen and begins raiding solar shipping, lasering outposts and generally misbehaving."

Sometimes the thought of the game designers sitting around throwing darts at a board covered with words, such as Nono, comes to mind. Let's hope they haven't forgotten words like Shoporia (the planet of the malls), Diskiedroops (those horrible creatures that store information) and Niknik (well, I won't go into that one).

Atarifest, DC

The Washington, D.C., area Atari user group is going to be holding its fourth Atarifest on October 1st and 2nd, 1988. The Atarifest will again showcase both ST and 8-bit software and hardware products. This show is unique in that it brings together several Atari groups from neighboring states. For more information, contact D.G. Elmore, 506 N. York Road, Sterling, VA 22170.

**Companies mentioned:
Glentop Publishers Ltd.
Standfast House
Bath Place
High Street
Barnet, Herts EN5 5XE
01-441-4130**

**Michtron
576 S. Telegraph
Pontiac, MI 48053
(313) 334-5700**

**Abacus
P.O. Box 7211
Grand Rapids, MI 49510
(616) 241-5510**

Wordlock

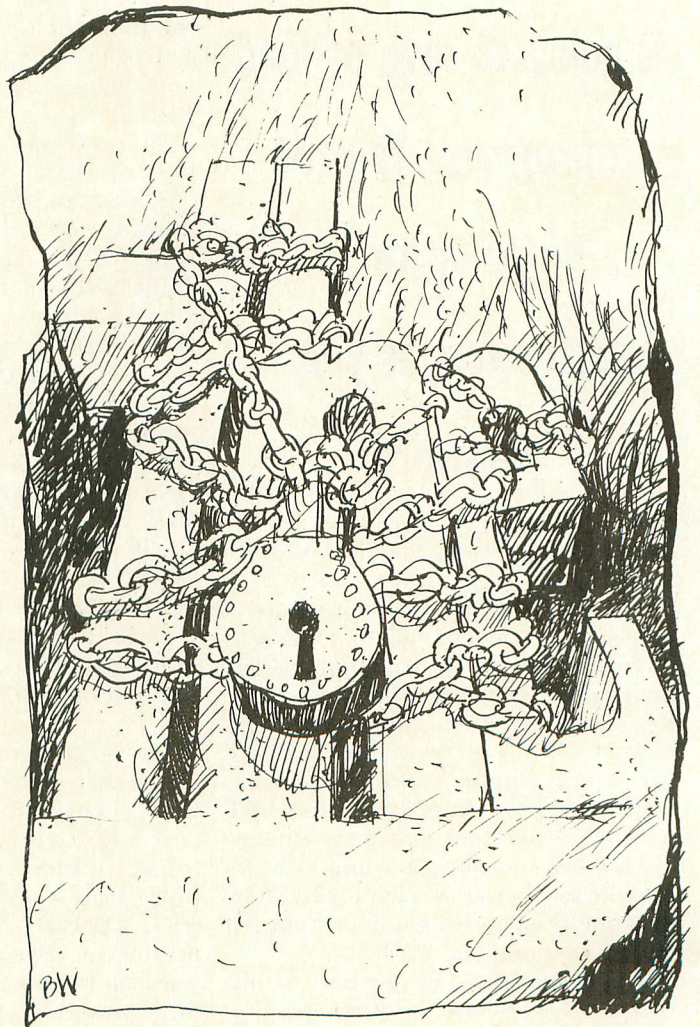
by Andy A. Lee

If you have \$10 million at home, where do you think is a safer place for it? The heavy-duty safe in your study or the center of your driveway, where everyone can see it?

The safe, of course, is the better place. The programs on your disks are like money on your driveway. You know most people who come by and see the money would take it if you're not around. This applies to your programs too! If you leave your disks lying around, sooner or later your programs will be copied without your authorization.

Wordlock is a binary program that will stop all that. Using a password of up to

The programs on your disks are like money on your driveway. You know most people who come by and see the money would take it if you're not around.



Wordlock

14 characters, it assures you that no one else will have access to your programs.

How does it do that?

First of all, let's take a look at the programs on your disks. Atari programs are made up of sequences of bytes, or numbers between 0 and 255. When you're saving a program, the computer puts the program directly from memory to the disk. So when you're loading it, the com-

puter just copies the sequences of bytes back to memory. And the computer loads it back to memory exactly as they are on disk. If we add one to every byte as it goes to the disk, we will have a sort of code.

If we want to restore the program, we should subtract one from every byte, and then put the result of the subtraction in memory. If we don't subtract one (copy the bytes exactly as they are on disk), the program will not be restored, because every byte will be off by one. See how it works?

Similar techniques are used in *Wordlock*, though they are much more complicated than just subtraction or addition. Every character of your password or passphrase is used in the coding process. So if you misspell the word by one letter or leave a letter out, you will not be able to load the program.

Type in Listing 1 using the M/L Editor, and name the resultant file `AUTORUN.SYS`.

To use *Wordlock*, you need to load the binary program as soon as possible after the boot process. This means if you are using a DOS that recognizes `AUTORUN.SYS`, you should name *Wordlock* `AUTORUN.SYS` so it will be loaded right after DOS. Such DOSs include DOS 2, DOS 2.5, DOS 3, SpartaDOS 23C.b, etc. If you are using a DOS that gives you control after the system is booted, please refer to your user's manual. It should tell you how to load a binary file. For example, `OSA+` and `SPARTADOS 1.1` give you a "D1:" prompt after the boot process. You'll only need to type in "WORDLOCK" and press RETURN if you name the *Wordlock* file `WORDLOCK.COM`.

Once *Wordlock* is loaded, it will be present until you turn the computer off. The reset key does nothing to it.

Password entering

The user will be asked to enter a password whenever the computer is going to save or load a program. You can enter up to 14 characters as your password. Carefully check your password before you press RETURN. If the program you're loading or saving doesn't need a password, all you have to do when it asks for the password is press RETURN. This tells

Wordlock to save or load the program as it is in memory or on disk. You need to do this to load old programs. Here's how to assign a password to your old programs:

1. Load the program.
2. When *Wordlock* asks for the password, press RETURN.
3. Save the program.
4. *Wordlock* will ask for a password. Enter your password for this program.
5. *Remember the password!*

Last word

Since it's impossible to restore your programs if you don't have the password,

The user will be asked to enter a password whenever the computer is going to save or load a program. You can enter up to 14 characters as your password.

Since it's impossible to restore your programs if you don't have the password, use only one or two passwords on all your programs—or write your passwords down someplace.

puter just copies the sequences of bytes back to memory.

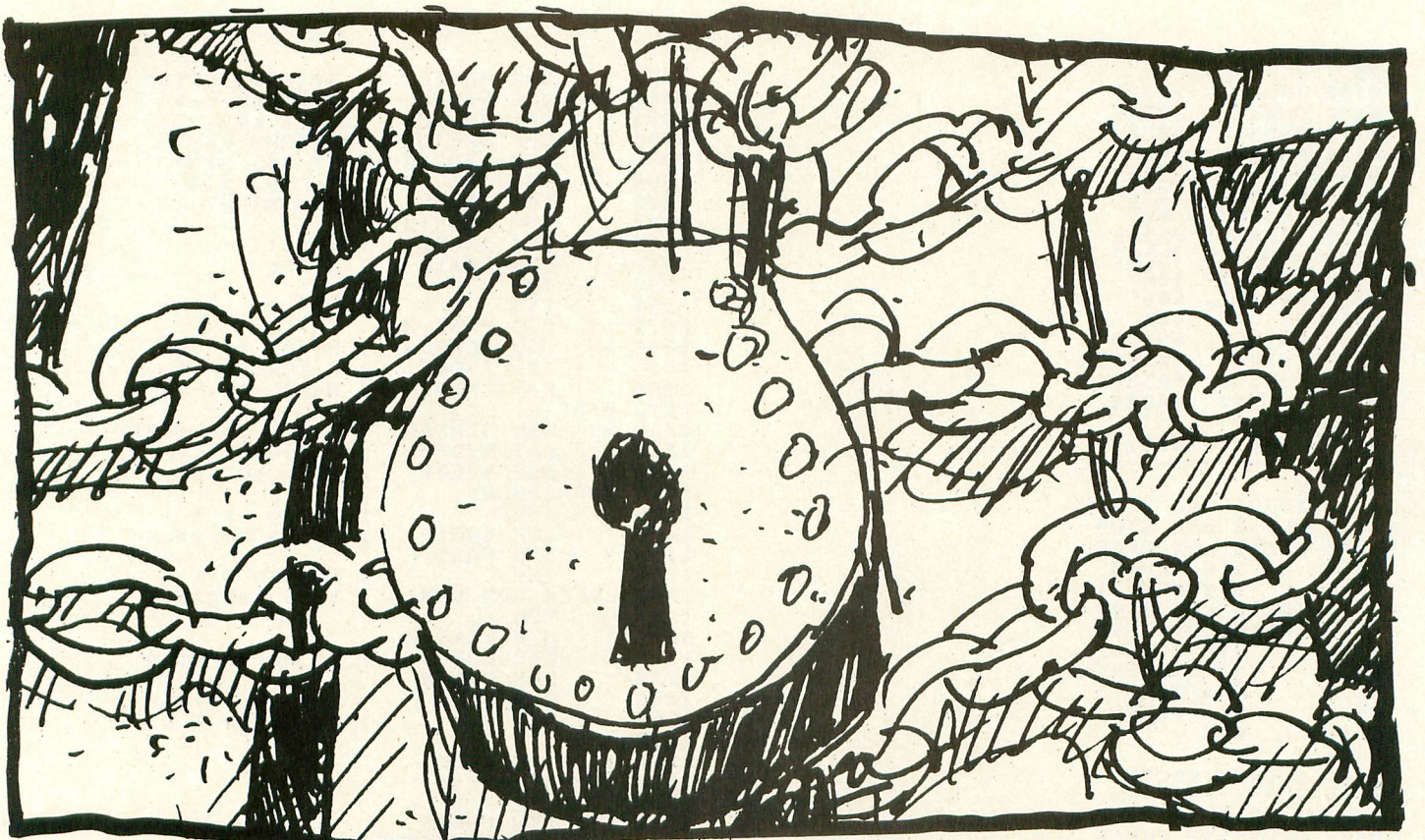
Let's play a game: Assume we have a number, 2534, and we want to remember this number by writing it down on paper. But we don't want to let others know what this number is. So we add one to every digit of 2534. After we do that, we will have a new number, 3645. We can now write that down on a piece of paper and nobody will guess the number is actually 2534. When we come back tomorrow to read the number, all we have to do is subtract one from every digit of 3645 to get the original number, 2534.

So much for fun—let's get back to our computer. We know the programs are

use only one or two passwords on all your programs—or write your passwords down some place. It's also a good idea to use at least four characters in your password.

Listing 2 is the assembly source code for *Wordlock*, written using `MAC/65`. *Wordlock* locates itself at `MEMLO`, the first available memory in RAM for users, so it should work with all DOSs and all 8-bit Atari languages.

Andy Lee is 18 years old and attends Concord High School in Elkhart, Indiana. He moved to the U.S. from Taiwan in the summer of '82. He has been programming Atari computers for over two years, and also runs a BBS called Green Alley Bulletin Board System.



LISTING 1: M/L EDITOR DATA

```

1000 DATA 255,255,0,64,78,66,32,255,25
5,162,0,160,0,142,231,2,6391
1010 DATA 140,232,2,162,0,160,0,134,12
,132,13,169,87,141,253,3,4847
1020 DATA 96,142,255,3,32,255,255,8,19
2,1,240,2,40,96,136,174,6806
1030 DATA 255,3,152,157,14,4,189,74,3,
201,4,240,4,201,8,208,5659
1040 DATA 85,169,11,141,254,3,172,254,
3,185,254,64,32,235,64,206,9985
1050 DATA 254,3,208,242,32,246,64,160,
14,217,9,65,240,246,136,208,1373
1060 DATA 248,201,126,208,13,172,254,3
,240,234,32,235,64,206,254,3,1065
1070 DATA 16,226,201,155,240,24,168,17
3,254,3,201,14,240,214,13,255,1751
1080 DATA 3,170,152,157,0,4,32,235,64,
238,254,3,16,198,32,235,7641
1090 DATA 64,172,254,3,208,8,152,174,2
55,3,157,0,4,200,174,255,9410
1100 DATA 3,152,157,15,4,160,1,174,255
,3,40,96,142,255,3,32,4207
1110 DATA 255,255,8,140,254,3,174,255,
3,32,193,64,172,254,3,40,6720
1120 DATA 96,32,193,64,76,255,255,72,1
42,255,3,189,14,4,13,255,7028
1130 DATA 3,168,254,14,4,189,14,4,221,
15,4,208,5,169,0,157,3193
1140 DATA 14,4,104,240,8,217,0,4,240,3
,89,0,4,174,255,3,3338
1150 DATA 96,168,173,7,228,72,173,6,22
8,72,152,96,173,37,228,72,7895
1160 DATA 173,36,228,72,96,32,58,32,10
0,114,111,119,115,115,97,80,4240
1170 DATA 27,28,29,30,31,125,127,156,1
57,158,159,253,254,255,166,212,5034
1180 DATA 164,213,142,48,66,140,50,66,
174,42,3,172,43,3,134,212,4522
1190 DATA 132,213,173,253,3,201,87,208
,3,76,47,66,166,12,164,13,3863
1200 DATA 142,1,64,140,2,64,174,231,2,
172,232,2,134,12,132,13,3968
1210 DATA 142,14,64,140,16,64,24,173,2
31,2,105,24,141,4,64,173,3307
1220 DATA 232,2,105,1,141,6,64,160,0,2

```

```

4,177,212,105,1,141,31,2965
1230 DATA 64,200,177,212,105,0,141,32,
64,160,4,24,177,212,105,1,4209
1240 DATA 141,170,64,200,177,212,105,0
,141,171,64,200,24,177,212,105,9338
1250 DATA 1,141,191,64,200,177,212,105
,0,141,192,64,160,0,24,169,6182
1260 DATA 26,109,231,2,145,212,200,169
,0,109,232,2,145,212,160,4,7937
1270 DATA 24,169,165,109,231,2,145,212
,200,169,0,109,232,2,145,212,9850
1280 DATA 200,24,169,186,109,231,2,145
,212,200,169,0,109,232,2,145,8666
1290 DATA 212,162,17,189,61,66,133,213
,202,189,61,66,133,212,160,1,8253
1300 DATA 56,177,212,233,0,145,212,200
,177,212,233,64,145,212,136,24,1553
1310 DATA 177,212,109,231,2,145,212,20
0,177,212,109,232,2,145,212,202,3290
1320 DATA 16,209,174,231,2,172,232,2,1
34,212,132,213,160,0,185,0,8071
1330 DATA 64,145,212,200,208,248,230,2
13,160,23,185,0,65,145,212,136,898
1340 DATA 192,255,208,246,24,173,231,2
,105,24,141,231,2,173,232,2,7909
1350 DATA 105,1,141,232,2,162,0,160,0,
134,212,132,213,169,87,141,9022
1360 DATA 253,3,96,70,64,100,64,128,64
,136,64,78,64,67,64,83,2213
1370 DATA 64,179,64,187,64,226,2,227,2
,24,65,0,0,0,0,0,7211

```

LISTING 2: ASSEMBLY

```

10 .OPT NO LIST
20 ; *****
30 ; *
40 ; * WORD-LOCK *
50 ; *
51 ; * April, 1986 *
52 ; *
60 ; * by Andy A. Lee *
70 ; *
80 ; *****

```

```

90 ;
0100 INDEX = $03FE
0110 CHN16 = $03FF
0120 BADS = $0E
0130 MAXIN = $0E
0140 DELETE = $7E
0150 RETURN = $9B
0160 DOSINI = $0C
0170 ICAX1 = $034A
0180 MEMLO = $02E7
0190 READ = $04
0200 WRITE = $08
0210 OK = $01
0220 PASS = $0400
0230 CNTR = PASS+14
0240 LENS = PASS+15
0250 *= $4000
0260 ;
0270 ; Rest Init...
0280 ; -----
0290 ;
0300 TODOS JSR $FFFF
0310 MLLSB LDX #$00
0320 MLMSB LDY #$00
0330 STX MEMLO
0340 STY MEMLO+1
0350 TDLSB LDX #$00
0360 TDM5B LDY #$00
0370 STX DOSINI
0380 STY DOSINI+1
0390 LDA #'W
0400 STA $03FD
0410 RTS
0420 ;
0430 ; Open vector
0440 ; -----
0450 ;
0460 OPEN STX CHN16
0470 DOPN JSR $FFFF ; Open the file
0480 PHP ; Save status
0490 CPY #OK ; Process okay?
0500 BEQ OURJOB ; OK! Our turn!
0510 PLP ; Error code
0520 RTS ; Return
0530 ;
0540 ; Our works
0550 ; -----
0560 ;
0570 OURJOB DEY
0580 LDX CHN16
0590 TYA ; reset counter
0600 STA CNTR,X ; to zero
0610 LDA ICAX1,X ; Is this...
0620 CMP #READ ; read a file?
0630 BEQ CNT ; Continue if so
0640 CMP #WRITE ; write a file?
0650 BNE NOPASSWORD ; nopassword
0660 ; if it isn't read or write...
0670 ;
0680 ; PRINT "Password : "
0690 ; -----
0700 ;
0710 CONT LDA #11 ; 11 characters
0720 STA INDEX
0730 PRLOOP
0740 LDY INDEX ; Load index
0750 M1 LDA MSG-1,Y ; A character
0760 S1 JSR SCREEN ; Print it
0770 DEC INDEX ; Next character
0780 BNE PRLOOP ; Done?
0790 ;
0800 ; Input the password.
0810 ; -----
0820 ;
0830 INLOOP
0840 I1 JSR INPUT ; A keystroke
0850 LDY #BADS
0860 CHLOOP
0870 B1 CMP BADIN-1,Y ; Compare with
0880 BEQ INLOOP ; unusable
0890 DEY ; input table.
0900 BNE CHLOOP
0910 CMP #DELETE ; Delete backs?
0920 BNE CH2 ; If not...
0930 LDY INDEX ; Check index
0940 BEQ INLOOP ; No input yet!
0950 S2 JSR SCREEN ; Delete letter
0960 DEC INDEX ; = INDEX - 1
0970 BPL INLOOP ; Next key
0980 CH2 CMP #RETURN ; RETURN key?
0990 BEQ STOPIN ; If so, stop
1000 TAY ; Y reg. = A
1010 LDA INDEX ; Too many?
1020 CMP #MAXIN
1030 BEQ INLOOP ; Yes!!!
1040 ORA CHN16 ; Store
1050 TAX
1060 TYA
1070 STA PASS,X
1080 S3 JSR SCREEN ; Print it too.
1090 INC INDEX ; = INDEX + 1
1100 BPL INLOOP ; Next key
1110 STOPIN
1120 S4 JSR SCREEN ; Print the <CR>
1130 LDY INDEX ; PASSWORD?
1140 BNE SVLEN ; Yes, go on...
1150 NOPASSWORD
1160 TYA ; Y reg. = 0
1170 LDX CHN16 ; Clean password
1180 STA PASS,X
1190 INY ; Length = 1
1200 SVLEN LDX CHN16 ; Save length
1210 TYA
1220 STA LENS,X
1230 LDY #OK ; No error flag
1240 LDX CHN16
1250 PLP
1260 RTS ; All done.
1270 ;
1280 ; Getvector
1290 ; -----
1300 ;
1310 GET STX CHN16
1320 DGET JSR $FFFF ; Get a byte
1330 PHP ; Save status
1340 STY INDEX ; Save Y reg.
1350 LDX CHN16
1360 C1 JSR CODE ; Decode byte
1370 LDY INDEX ; Restore Y reg.
1380 PLP ; Reload status
1390 RTS ; Done!
1400 ;
1410 ; Put vector
1420 ; -----
1430 ;
1440 PUT
1450 C2 JSR CODE ; Code the byte
1460 DPUT JMP $FFFF ; Output to disk
1470 CODE PHA
1480 STX CHN16 ; Save channel #
1490 LDA CNTR,X ; calculate the
1500 ORA CHN16 ; byte to EOR
1510 TAY ; with...
1520 INC CNTR,X ; Reset counter
1530 LDA CNTR,X ; Set back to
1540 CMP LENS,X ; zero?
1550 BNE NOTNOW
1560 LDA #$00 ; set to zero
1570 STA CNTR,X
1580 NOTNOW PLA ; Load the byte.
1590 BEQ EXIT ; ZERO? bypass!
1600 CMP PASS,Y ; = CODE BYTE?
1610 BEQ EXIT ; Bypass
1620 EOR PASS,Y ; Coding...
1630 EXIT LDX CHN16 ; All done
1640 RTS ; Bye-bye!
1650 ;
1660 ; Screen - print a character
1670 ; on screen.
1680 ; -----
1690 SCREEN
1700 TAY ; A = character
1710 LDA $E407 ; location of
1720 PHA ; PRINT routine
1730 LDA $E406 ; in OS
1740 PHA
1750 TYA ; restore A
1760 RTS ; goto routine
1770 ;
1780 ; Input - input a keystroke
1790 ; -----
1800 INPUT

```

```

1810 LDA $E425 ; location of
1820 PHA ; INPUT routine
1830 LDA $E424 ; in 05
1840 PHA
1850 RTS ; goto routine
1860 MSG
1870 .BYTE " : drowssaP"
1880 ;
1890 ; Following characters are not
1900 ; usable for password.
1910 ;
1920 BADIN
1930 .BYTE 27,28,29,30,31,125,127
1940 .BYTE 156,157,158,159,253
1950 .BYTE 254,255
1960 ;
1970 ; Install WORDLOCK
1980 ;
1990 ;
2000 INIT LDX $D4
2010 LDY $D5
2020 STX X+1
2030 STY Y+1
2040 LDX $032A ; Get address
2050 LDY $032B ; of "D" vectors
2060 STX $D4 ; table
2070 STY $D5
2080 ;
2090 ; We will see if WORDLOCK is
2100 ; already installed
2110 ;
2120 CHECK
2130 LDA $03FD
2140 CMP #'W
2150 BNE COPY
2160 JMP EXITINIT
2170 ;
2180 ; We now install WORDLOCK
2190 ;
2200 COPY
2210 LDX DOSINI
2220 LDY DOSINI+1
2230 STX TODOS+1
2240 STY TODOS+2
2250 LDX MEMLO
2260 LDY MEMLO+1
2270 STX DOSINI
2280 STY DOSINI+1
2290 STX TDL5B+1
2300 STY TDM5B+1
2310 CLC ; calculate new
2320 LDA MEMLO ; MEMLO
2330 ADC # <INIT-$4000
2340 STA MLL5B+1
2350 LDA MEMLO+1
2360 ADC # >INIT-$4000
2370 STA MLM5B+1
2380 LDY #0 ; First, copy
2390 CLC ; OPEN vector
2400 LDA ($D4),Y ; so we can use
2410 ADC #501 ; it...
2420 STA DOPN+1
2430 INY
2440 LDA ($D4),Y
2450 ADC #500
2460 STA DOPN+2
2470 LDY #4 ; Now the GET
2480 CLC ; vector too...
2490 LDA ($D4),Y
2500 ADC #501
2510 STA DGET+1
2520 INY
2530 LDA ($D4),Y
2540 ADC #500
2550 STA DGET+2
2560 INY ; Last, PUT
2570 CLC ; vector...
2580 LDA ($D4),Y
2590 ADC #501
2600 STA DPUT+1
2610 INY
2620 LDA ($D4),Y
2630 ADC #500
2640 STA DPUT+2
2650 LDY #0
2660 CLC ; Insert my OPEN

```

```

2670 LDA # <OPEN-$4001 ; vector
2680 ADC MEMLO
2690 STA ($D4),Y
2700 INY
2710 LDA # >OPEN-$4001
2720 ADC MEMLO+1
2730 STA ($D4),Y
2740 LDY #4 ; Insert my GET
2750 CLC ; vector
2760 LDA # <GET-$4001
2770 ADC MEMLO
2780 STA ($D4),Y
2790 INY
2800 LDA # >GET-$4001
2810 ADC MEMLO+1
2820 STA ($D4),Y
2830 INY ; Insert my PUT
2840 CLC ; vector
2850 LDA # <PUT-$4001
2860 ADC MEMLO
2870 STA ($D4),Y
2880 INY
2890 LDA # >PUT-$4001
2900 ADC MEMLO+1
2910 STA ($D4),Y
2920 LDX #17
2930 CAL LDA MOVETHESE,X
2940 STA $D5
2950 DEX
2960 LDA MOVETHESE,X
2970 STA $D4
2980 LDY #1
2990 SEC
3000 LDA ($D4),Y
3010 SBC # <$4000
3020 STA ($D4),Y
3030 INY
3040 LDA ($D4),Y
3050 SBC # >$4000
3060 STA ($D4),Y
3070 DEY
3080 CLC
3090 LDA ($D4),Y
3100 ADC MEMLO
3110 STA ($D4),Y
3120 INY
3130 LDA ($D4),Y
3140 ADC MEMLO+1
3150 STA ($D4),Y
3160 DEX
3170 BPL CAL
3180 LDX MEMLO
3190 LDY MEMLO+1
3200 STX $D4
3210 STY $D5
3220 LDY #0
3230 MOVE LDA TODOS,Y
3240 STA ($D4),Y
3250 INY
3270 BNE MOVE
3271 INC $D5
3272 LDY # <INIT-$4001
3273 MOVE2 LDA TODOS+$0100,Y
3274 STA ($D4),Y
3275 DEY
3276 CPY #$FF
3277 BNE MOVE2
3280 CLC ; calculate new
3290 LDA MEMLO ; MEMLO
3300 ADC # <INIT-$4000
3310 STA MEMLO
3320 LDA MEMLO+1
3330 ADC # >INIT-$4000
3340 STA MEMLO+1
3350 EXITINIT
3360 X LDX #500
3370 Y LDY #500
3380 STX $D4
3390 STY $D5
3400 LDA #'W
3410 STA $03FD
3420 RTS
3430 MOVETHESE .WORD S1,S2,S3,S4
3440 .WORD I1,M1,B1,C1,C2
3450 *= $02E2
3460 .WORD INIT

```

PrintScreen

by Justin E. Wilder

Often, while a program is running in our computer, it would be handy if we could print the wording or data on the screen. Many programs provide a way to print certain information, but we might want to print some other text which appears on the screen. There are utilities that allow printing from the screen to a printer, but to access them we must stop execution of any program which may be running to give a command such as an immediate `USR` statement. What we need is the ability to print any text which is displayed, and still continue operation of the program. The *PrintScreen* utility allows this to be done with a single keystroke.

The BASIC program in the accompanying listing can be entered and used in either of two ways to set up *PrintScreen*. (*Be sure you save the program to disk or tape before running it, because one of the options clears the program from memory when it is finished.*) If you use a disk drive, you may choose to have a binary (machine language) file set up on a disk. This file should have the name `PRINTSCN.OBJ` and will be loaded into memory with the binary load option of DOS before you run another program. You could also change the name of the `PRINTSCN.OBJ` file to `AUTORUN.SYS` (if you do not already have an `AUTORUN.SYS` file on that disk), and it will be installed in memory when you boot the computer. If you already have an `AUTORUN.SYS` file, you can probably add

the `PRINTSCN.OBJ` code to the end of it using the `COPY` with append option of DOS. Be sure to have another copy of your program on disk before trying this in case it does not work properly.

The BASIC program shown in Listing 1 can also be used to install directly into memory. It's installed as a machine-language function in Page 6 of RAM and initialized before the BASIC program is automatically deleted. In this way, you can use this utility whether you have a disk drive or not.

Once *PrintScreen* is installed in memory, you can load and run another program. At any time you want a print-out of the screen, you can make it by holding the control key and pushing the "?" key. It will work properly only with a Graphics 0 display and if the program

does not use Page 6 of memory. Also, if the reset key is pushed, the initialization is lost. From BASIC, you can reinitialize this utility by entering the immediate mode command, which is X=USR(1591).

PrintScreen can be used with any Atari 400, 800, XL or XE computer. It requires any Atari-compatible printer to be properly connected for the printout. Inverse video characters are printed as regular characters and control or graphics characters are represented by a period to avoid conflicts with printer features. Once installed in memory or in a disk file, it does not require BASIC and can be used with machine-language programs as well as BASIC programs.

What makes it work?

PrintScreen uses the keyboard interrupt vector to get temporary control of the computer when CTRL-? is pushed. It uses memory locations 205, 206 and 207 in addition to the last 201 bytes of Page 6 in RAM. Referring to the assembly language listing, the first part is the initialization section which puts the address of the main routine into the keyboard interrupt vector. The previous contents of this vector are placed in a jump (JMP) instruction to send operation to the keyboard handler if a key other than CTRL-? is pushed. If any I/O operation to a device other than the screen or keyboard is in progress, the CTRL-? key is ignored.

Before the microprocessor registers X and Y are used, their contents are saved on the hardware stack, so that they can be restored to their previous values before operation is returned to the program which was running when CTRL-? was pushed. The contents of the processor status register and accumulator are already on the stack. The interrupt mask has also been set, and since the input/output (I/O) operation to the printer involves interrupts, the mask must be cleared with a CLI instruction.

Now, here's the tricky part. The Atari system provides eight I/O Control Blocks, so that up to eight I/O operations can be done at once, right? Wrong! Up to eight files or devices can be open for I/O, but during an actual I/O operation the contents of the associated I/O Control Block are transferred to one block in Page 0 RAM from where control of the operation is maintained. The Atari is thus designed to carry out only one I/O operation at a time. The usual time when a printout of the screen would be requested is when a program is waiting for input from the keyboard. This means that it is in the middle of an I/O to the screen

editor. If the screen printing changes the values in the zero page I/O Control Block, they will not be right when the program resumes and the computer will run out of control (lock up). Therefore the contents of the zero page block are saved on the stack before printing the screen, and are restored to their previous values afterward.

The printing operation is carried out in the normal way through the Central I/O routines in ROM. A control block is first opened for output to the printer. The characters in screen memory are then converted to ASCII values and sent to the printer, one line at a time and finally the block is closed. I/O Control Block seven was chosen, because its other uses are in similar open, do and close operations

Be sure you save the program to disk or tape before running it, because one of the options clears the program from memory when it is finished.

If the screen printing changes the values in the zero page I/O Control Block, they will not be right when the program resumes, and the computer will run out of control.

and are not likely to cause any conflict. When the values which had been saved on the stack are returned to their proper places, control is passed back to the original program by a Return from Interrupt (RTI) instruction to continue as though nothing had happened.

Justin E. Wilder, a 1953 graduate of the University of Michigan, is a senior project engineer for Johnson Controls, Inc. He purchased his Atari 800 in 1980, and his Ultimate Renumber Utility was in the Atari Program Exchange. He is a member of the Indiana-Michigan Atari Group Exchange (IMAGE) users' group.

LISTING 1: BASIC

```

KL 10 REM PRINTSCREEN --J.E. Wilder
WQ 11 REM COPYRIGHT 1988
XU 12 REM BY ANALOG COMPUTING
DI 20 REM To print GR.0 screen push CTRL-
?
HD 30 DIM AN$(1):PRINT "Set up PRINTSCREE
N in:"
ZK 40 PRINT "DISK FILE or MEMORY";:INPUT
AN$:IF AN$="M" THEN 80
WA 50 IF AN$("<">"D" THEN 40
NA 60 ? "CREATING BINARY FILE":OPEN #1,8,
0,"D:PRINTSCN.OBJ"
DL 70 FOR X=1 TO 213:READ A:PRINT #1;CHR$(
A);:NEXT X:END
CO 80 PRINT "INSTALLING PRINTSCREEN IN PA
GE SIX OF MEMORY. TO USE PUSH CTRL-?"
ZA 90 RESTORE 110:FOR X=1591 TO 1791:READ
A:POKE X,A:NEXT X:X=USR(1591):NEW
LD 100 DATA 255,255,55,6,255,6
YY 110 DATA 104,173,8,2,141,97,6,173,9,2
KA 120 DATA 141,98,6,169,79,141,8,2,169,6
LA 130 DATA 141,9,2,96,173,9,210,201,166,
208
QC 140 DATA 10,165,32,201,6,144,4,201,15,
144
NG 150 DATA 3,76,0,0,152,72,138,72,162,15
WI 160 DATA 181,32,72,202,16,250,134,32,1
65,88
UL 170 DATA 133,206,165,89,133,207,169,24
,133,205
GT 180 DATA 88,162,112,169,3,157,66,3,169
,253
LQ 190 DATA 157,68,3,169,3,157,69,3,169,8
0
XM 200 DATA 141,253,3,169,155,141,254,3,1
57,72
MD 210 DATA 3,169,8,157,74,3,32,86,228,16
9
GT 220 DATA 9,157,66,3,160,0,165,205,240,
37
UG 230 DATA 177,206,41,127,201,123,176,4,
201,97
UF 240 DATA 176,8,105,32,201,96,144,2,169
,46
JY 250 DATA 153,253,3,152,200,192,40,144,
227,101
PK 260 DATA 206,133,206,144,2,230,207,169
,155,153
ZC 270 DATA 253,3,32,86,228,48,4,198,205,
16
FU 280 DATA 199,169,12,157,66,3,32,86,228
,162
MX 290 DATA 0,198,205,134,205,16,140,104,
149,32
SC 300 DATA 232,224,16,144,248,104,170,10
4,168,104
ER 310 DATA 64,226,2,227,2,56,6

```

```

0200 CMP #15
0210 BCC PRINT E:;5:or K:
0220 EXIT JMP $00
0230 PRINT TYA
0240 PHA
0250 TXA
0260 PHA
0270 LDX #15
0280 SAVE LDA 32,X Page 0 IOCB
0290 PHA
0300 DEX
0310 BPL SAVE
0320 STX 32 No repeat CTRL-?
0330 LDA 88 Screen Mem
0340 STA 206 Pointer
0350 LDA 89
0360 STA 207
0370 LDA #24
0380 STA 205 Line count
0390 CLI
0400 OPEN LDX #570 IOCB 7
0410 LDA #3 Open
0420 STA 834,X Comd
0430 LDA #1021&255 CASBUF
0440 STA 836,X Buf Adr
0450 LDA #1021/256
0460 STA 837,X
0470 LDA #80 P: Device
0480 STA 1021 CASBUF
0490 LDA #155 EOL
0500 STA 1022
0510 STA 840,X Buf Len
0520 LDA #8 Output
0530 STA 842,X AUX1
0540 JSR $E456 CIOV Open
0550 LDA #9 PUT REC
0560 STA 834,X Comd
0570 NEXT LDY #0
0580 LDA 205 LINE
0590 BEQ EOL
0600 LOOP LDA (206),Y Screen Char
0610 AND #127 No inverse
0620 CMP #123
0630 BCS $HIFT>Z
0640 CMP #97
0650 BCS KEEP >=a
0660 SHIFT ADC #32 To ASCII
0670 CMP #96
0680 BCC KEEP<=_
0690 LDA #46 Dot-Unprintable
0700 KEEP STA 1021,Y CASBUF
0710 TYA
0720 INY
0730 CPY #40
0740 BCC LOOP
0750 ADC 206 Screen pnter +40
0760 STA 206
0770 BCC #+4
0780 INC 207
0790 EOL LDA #155
0800 STA 1021,Y
0810 JSR $E456 CIOV Print line
0820 BMI ERROR
0830 DEC 205 Line
0840 BPL NEXT
0850 ERROR LDA #12 Close
0860 STA 834,X Comd
0870 JSR $E456 CIOV
0880 LDX #0
0890 DEC 205 Line
0900 STX 205
0910 BPL OPEN Clr printer
0920 REFIL PLA
0930 STA 32,X Page 0 IOCB
0940 INX
0950 CPX #16
0960 BCC REFIL
0970 PLA
0980 TAX
0990 PLA
1000 TAY
1010 PLA
1020 RTI
1030 *- $02E2 INITAD
1040 .WORD INIT
1050 .END

```

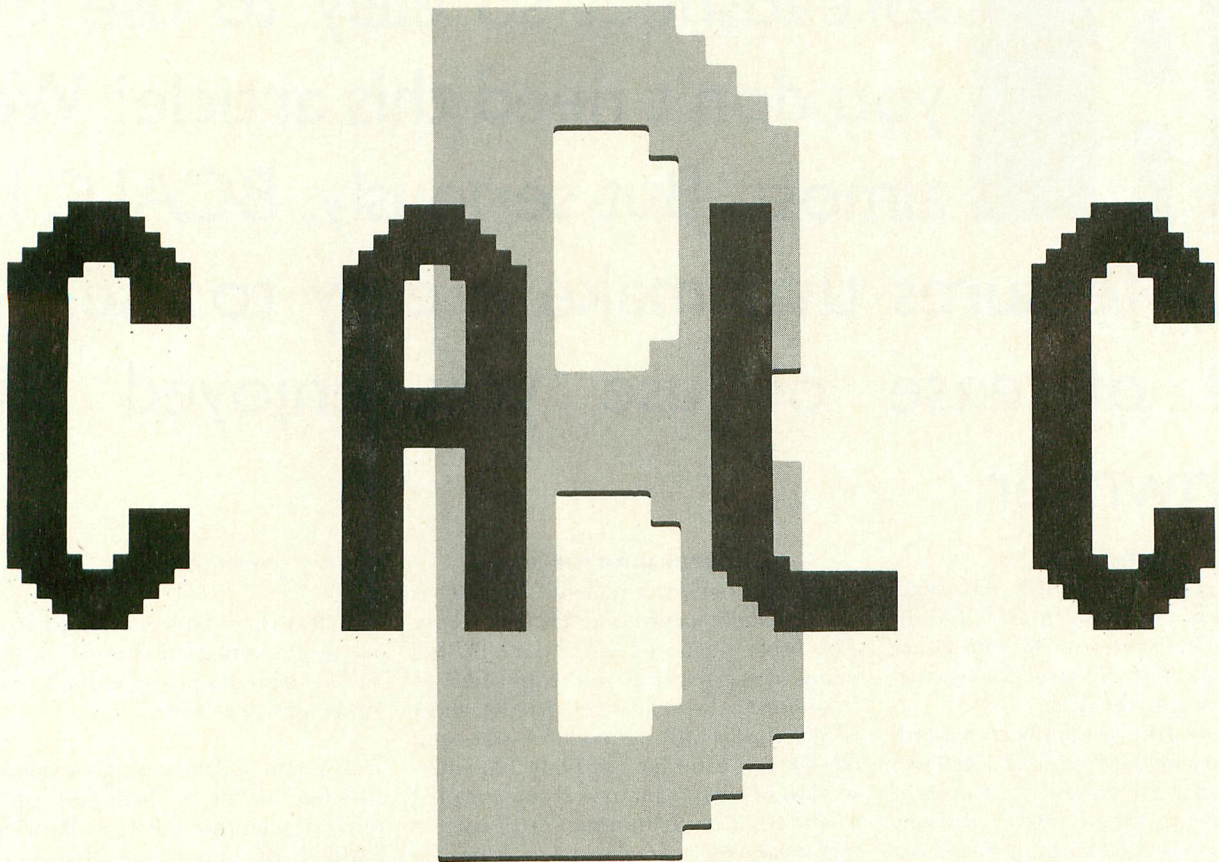
LISTING 2: ASSEMBLY

```

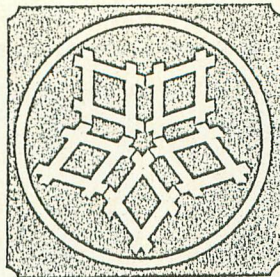
10 ;PRINTSCREEN --J.E. Wilder
15 ;COPYRIGHT 1988
16 ;BY ANALOG COMPUTING
20 ;Prints Gr. 0 screen with CTRL-?
30 *- $0637
40 PLA
50 INIT LDA $0208 UKEYBD
60 STA EXIT+1
70 LDA $0209
80 STA EXIT+2
90 LDA #INTRPT&255
0100 STA $0208 UKEYBD
0110 LDA #INTRPT/256
0120 STA $0209
0130 RTS
0140 INTRPT LDA 53769 KBCODE
0150 CMP #5A6 CTRL-?
0160 BNE EXIT
0170 LDA 32 Device in use
0180 CMP #6
0190 BCC EXIT P: or C:

```


GENERAL



by Barry Kolbe and Bryan Schappel



BCALC

This spreadsheet so easy to use that you don't need this article! Well, almost. But seriously, BCALC has many features that make it easy to use, the kind of ease of use you enjoyed with Atariwriter.

Typing it in

Listing 1 contains the BASIC data statements used to create the *BCALC* file on your disk. Please refer to the *M/L Editor* found elsewhere in this issue for instructions in keying in BCALC.

Once the BCALC file has been created, simply binary load it from DOS. Refer to your DOS manual on how to do this if you are not sure. Please remove the cartridge from your 800 computer or turn BASIC off on your XL/XE computer by holding the option key down when booting up. When the program loads there will be a credit line at the top of the screen. Press any key to begin.

General information

BCALC (hereafter referred to as BC) is a spreadsheet using about 8K of memory. This leaves around 24K for data. BC has many of the features you would find in a commercial spreadsheet. The BC sheet is 64 columns wide (lettered AA—CL) and has 64 rows (numbered 01-64). The intersection of each column and row is called a cell. So if my arithmetic is still correct that means there are 64 x 64 or 4096 cell. Each cell has an overhead of at least four bytes plus whatever data is entered.

There are three kinds of information you can store in a cell. The type of data entered is displayed on the right side of

the top screen line.

1. Text Data: This is anything that does not begin with a number 0-9 or the "=" or "-" sign. It's possible to put numbers in as text (see below).

2. Numeric data: a number that begins with 0-9, "-", or "." Numbers can be entered in scientific notation. For example, 3.45E-45. BC only accepts that part of input which is a valid number. For example, if you type: "-.234fgh" BC takes the "-.234" and ignores the rest. You may not type in commas. To display commas

(continued on second page following)

The screenshot shows the BCALC spreadsheet interface. At the top, it displays 'B-calc (c) 1987 AB15 TYPE:Function'. Below this, there are two columns of data: 'aa' and 'ab'. The first column contains row numbers from 01 to 18. The second column contains text and numerical values. The text includes 'BUDGET FOR 1988', 'MONTHLY BREAKDOWN', and 'TOTAL'. The numerical values are formatted with dollar signs and decimal points. At the bottom, there is a formula bar showing '=SUMCAB05,AB13)' and a status bar showing 'AA TOTAL 01 1988'.

aa	ab	ac	ad
01	BUDGET FOR 1988		
02	MONTHLY BREAKDOWN		
03	-----		LOAN
04			PAYMENTS
05	RENT	\$750.00	
06	FOOD	400.00	\$25.00
07	CAR	350.00	157.90
08	HEAT	125.00	92.00
09	INSURANCE	155.00	79.45
10	MEDICAL	50.00	
11	ELECTRIC	125.00	
12	TELEPHONE	75.00	\$353.45
13	LOANS	353.45	
14			
15	TOTAL	\$2392.10	
16			
17			
18			

=SUMCAB05,AB13)

AA TOTAL 01 1988

BOOT UP TO BIG SAVINGS!



1 YEAR FOR ONLY \$28
SAVE \$14 OFF THE COVER PRICE

1 YEAR WITH DISK ONLY \$105

**SAVE TIME AND MONEY
SUBSCRIBE TO ANALOG**

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

**SUBSCRIBE TO
ANALOG
TODAY**

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

PAYMENT ENCLOSED BILL ME
CHARGE MY: VISA MC # _____

EXPIRATION DATE _____ SIGNATURE _____

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

NAME _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

MAKE CHECK PAYABLE TO L.F.P., INC., P.O. Box 16927, N. Hollywood, CA 91615. Offer expires 10/26/88. Your first issue will arrive in 6 to 8 weeks.

WATCH FOR IT!

BOCALC

use the formatting commands which are discussed later.

3. Functions: An arithmetic or algebraic expression starting with the "=" sign (inverted also works). You mean I have to know algebra? Was my math teacher right in that there is a use for algebra? Perish the thought!

Entering functions is quite easy if one follows the rules. The five operations of \wedge (exponentiation), $*$ (multiplication), $/$ (division), $+$ (addition) and $-$ (subtraction) are used. These are used as operators between numbers, cells or built-in functions. Be aware of the order of operations. My Dear Aunt Sally. Huh? This is a mnemonic to remember the order of operations: multiplications and divisions are done first in order from left to right. Next, additions and subtractions in order from left to right. Exponents and parentheses are done first however. You want an example? Sure thing.

$$= 3 + ((2 + 7) \wedge 2 - 4 * 6) / 6$$

The innermost parentheses are done first: $(2 + 7)$ becomes 9. The 9 is then squared ($\wedge 2$) yielding 81. Next $4 * 6 = 24$ is done. Then the difference $81 - 24 = 57$ is calculated. Dividing by 6 yields 9.5. Add to 3 equals 12.5. Remember that each number above could be replaced by a cell or

a built-in function. For example:

$$= @RND((1 + AB24/12) \wedge 36) - BQ12$$

Always make sure that the cells used do not have text data or an error will result.

Since you might enter text data in lowercase, BC also allows you to enter functions in lower case; so you won't have to toggle that CAPS key. Just type away!

Console keys

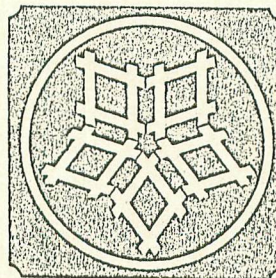
OPTION displays a status screen in the text window. The current filename, number of free bytes, recalculation status (on/off) and recalculation mode (row/column) are shown. Hit a key to exit.

SELECT puts you in the driver's seat. Well, actually in command mode. The text window will change color in this mode. You may use upper- or lowercase letters and even inverse video in commands. Some commands require an argument such as a cell reference (e.g., BG24) or a number. Zeros are necessary in cell references (e.g., CD02).

The commands are:

[DIR]ECTORY n displays up to 16 files on the screen. If you do not type a drive number "n," Drive 1 is the default. Make sure there is a space between "DIR" and "n." That is also true for the other commands. Press a key to read more file-

**We have tried to crash
this program in every
way we can think of.**



BOCALC

names. If there are exactly 16 files read in you will need to hit a key twice.

[WID]th crrr sets the width of column cc to width rr. The width rr must be greater than one but less than 32.

[SAV]E Dn:filespec is used to save the sheet. You must type a complete filespec including the "Dn:". Multiple drives are supported as well as a RAMDisk. If you attempt to save a sheet using the name of a file that already exists, you are asked if you wish to replace it. Merely hit the "Y" or "N" key to make your choice.

[LOA]D Dn:filespec is used to load a sheet that was saved. Seems reasonable.

[GOT]O crrr is the fast way to go from here to there or at least to cell crrr. cc must be in the range AA-CB and rr in the range 01-47.

[SET] crrr sets the column and row references that are displayed at the bottom of the text window. An example is definitely called for. If we type SET AA02 and the cursor is on DE12 we see that data from cell AA12 and from DE02 displayed.

If I were doing my grades (after all, I am a teacher), I might have student names in column AA. But when I'm entering data in cell AG12, column AA is nowhere to be seen. However, if I use SET AA12 the student name from column AA row

12 appears on the last text window line. Also shown is heading for column AG row 02.

[LIS]T Dn:filespec lists the spreadsheet to the disk so the information can be used with a word processor like Atariwriter. Use the cursor keys to define the block of information you would like listed on disk. If you define a bad block an error results. The information listed will be in exactly the same format as you see it on the screen.

LISTed files cannot be reloaded into the sheet. Doing so will probably cause the computer to lockup, and you will likely see a very unusual screen staring at you! Use the SAVE command to store the sheet on disk. Just in case you try to do something clever, like LIST to the same filename that was used to SAVE the sheet, you're told if the file already exists. You then may overwrite it if you wish. I'd recommend using a .LST extension on LISTED files.

START is used to abort input in either input mode, edit mode or command mode.

Special keys

These keys are used with the CONTROL key pressed simultaneously:

[-] (Arrow up) moves the cursor up one row.

[=] (Arrow down) moves cursor down one line.

[+] (Arrow left) moves one cell to the left.

[*] (Arrow right) moves one cell to the right.

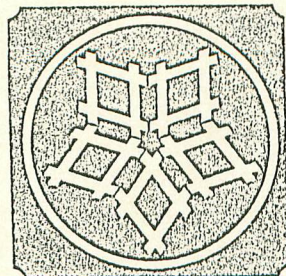
The name of the current cell (e.g., AA01) occupied by the cursor is displayed in the center of the top screen line. To the right of it is a message showing the type of data in the cell. This is good to know since a number on the screen could be a stored number or it could be the result of a function.

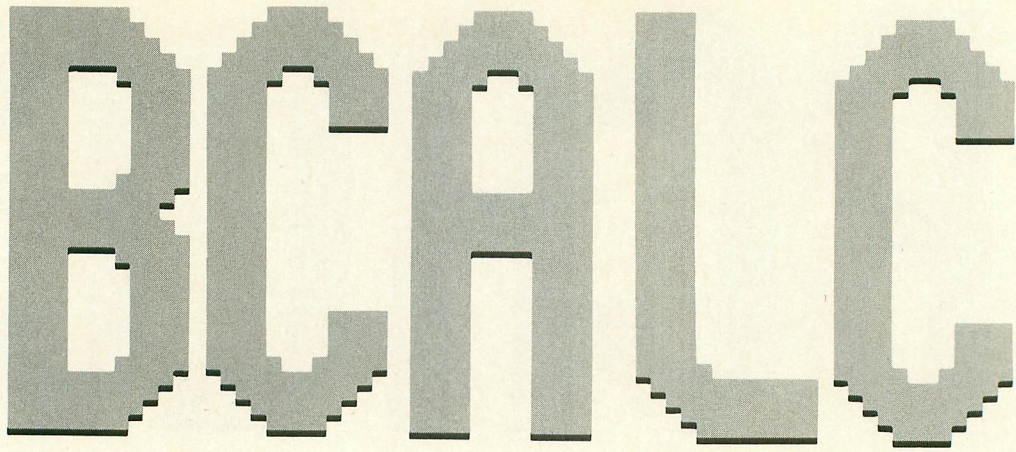
These keys can be used in input mode to terminate input and send the cursor to the next cell. For example, if you are on cell AA01 and you type NAMES(CTRL-*), "NAMES" is entered in cell AA01 and the cursor will move to cell AB01. RETURN keeps the cursor on the current cell. In edit mode these keys move you around the text window to give you full-screen editing.

[H] takes the cursor to AA01.

[D] deletes a block of cells. Follow the prompts. Specifying an improper block results in an error.

[C] copies a block of cells. See below.





[M] moves a block of cells. See below.

[T] forces text mode. This allows you to enter numbers, etc., as text. For example, 342-48-2333 as a number would result in just 342 being entered. In forced text mode it is entered as text.

[G] changes the global format of the column the cursor is on.

[L,C,R] keys control Left, Center and Right justification.

[D] toggles the "\$" symbol.

[.] toggles commas on and off.

[0-9] chooses the number of places displayed to the right of the decimal point.

[ESCAPE] makes the choice final and exits.

All new data entered in this column will take on the new format. Formatting information is stored with cell data when it is entered. So data already there has its own individual format. To change it use CTRL-F.

[F] formats an individual cell. Enter the data normally. Press CTRL-F. The global format information is displayed in the text window. Make the changes you desire as above in G. ESCAPE reformats the cell and makes the changes on the screen.

[E] displays the current cell data in the text window and allows you to edit the cell much as you are used to doing in Atari

BASIC.

[P] prints the sheet. Follow the prompts in defining a block to be printed. The information will be printed exactly as you see it on the screen. So check your column widths, etc. The printing is done by rows. If you have an 80-column printer and select columns whose widths add to more than 80 there will be some wrap-around. Therefore you might want to print left and right halves of your sheet.

It's possible to send printer codes out by typing them into a cell. Since some of these codes might conflict with the way BC handles input you might need to type a space first and then the codes.

[O] toggles the recalculation on/off flag. Turn it off to type in data. Otherwise the whole sheet is recalculated every time you enter data, which takes some time to do.

[R] changes recalculation order from by rows to by columns or vice-versa.

[K] erases a cell from the sheet.

[CAPS] toggles between upper and lowercase letters. This key can be used in any input mode.

[A] recalculates the whole sheet immediately. Use CTRL - E and RETURN to recalculate a particular cell.

[ESC] aborts the delete, copy, move and print options.

Let's discuss copy and move. Copy makes an exact duplicate of the block of cells you chose. Move does the same thing but cell references are changed. For example:

cell	contents
AA01	12.35
AA02	4
AA03	=AA01+3

Cell AA03 has 15.35 as a result. Copying cells AA02 and AA03 to AM23 results in:

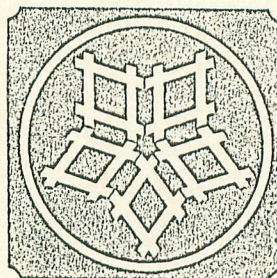
cell	contents
AM22	32
AM23	4 (this cell existed before.)
AM24	=AM23+3

The result in AM24 is 15.35 (the same as before). If you move the same cells to AM23 the result is:

cell	contents
AM22	32
AM23	4
AM24	=AM22+3

Here AM24 will contain 35. Notice how the cell reference was changed in cell AM24.

After a copy or move some cells will be overwritten. None of the source cells are deleted in either except those that are overwritten. Clean up any cells not wanted using the delete option (CTRL - D).



BOCALC

We'd suggest that before doing a copy or move you save the sheet. It's just prudent practice. We have tried to crash this program in every way we can think of, but there is always the unknown. Maybe a circuit gets a little overheated and causes a crash. I even got SynCalc to crash once. Fortunately, I was just fooling around with it. The old adage "Better SAVE than sorry" applies here.

Built-in functions

The built-in functions are typed in this format: @CNT(AA03, BX24) or @SQR(23.45) or @SQR(BD23). Parentheses must be used and cells must be separated by a comma.

The following functions require one argument which could be a cell or a number:

function	operation
SQU	squaring
SQR	square root
ABS	absolute value
RND	round value
EXP	exponentiation base 10
LOG	logarithm base 10
INT	greatest integer

These functions require two arguments and they must be cells.

function	operation
CNT	count the number of cells in a block.
SUM	add the entries in a block.
AVE	find the average of all the entries in a block.

You mean I have to
know algebra? Was my
math teacher right in
that there is a use for
algebra? Perish the
thought!

CNT counts all the cells that have entries in a block whether they be text, numbers or functions. SUM however, adds up only the values in cells containing a numeric result. Text cells are treated as a zero. Empty cells result in the word "Error" displayed on the screen. AVE operates like SUM. However, AVE uses the CNT and SUM routines to get its value. So if you try to average cells containing RENT,400,500,600, the result will be $(0(\text{RENT}) + 400 + 500 + 600)/4 = 375$, an incorrect result.

Functions can be mixed in with any arithmetic expression as in:

$$= \text{AA01} + (4 * \text{@AVE}(\text{CD24}, \text{CE32}) \wedge \text{@RND}(\text{AG12})) - (2 + \text{@SQR}(\text{AA02}))$$

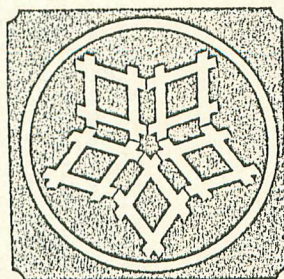
However, functions are not nestable, i.e., you can not type =@ABS(@EXP(AA01)).

Error messages

We used standard error numbers wherever we could to keep the program as small as possible. The most common errors are:

2—out of memory. You have used all available memory. Save the sheet as it is.

138 or 139—Device does not respond or malfunctions. This probably will refer to the printer not being on.



36,32,238,60,32,141,221,5279
1210 DATA 32,92,46,76,189,38,201,45,14
4,31,208,15,173,193,36,201,7589
1220 DATA 58,176,22,201,47,240,18,201,
46,144,14,201,47,240,10,201,7666
1230 DATA 58,176,6,32,5,46,76,189,38,1
69,0,133,194,141,187,72,6212
1240 DATA 32,9,44,165,147,133,145,165,
158,32,39,45,32,104,42,165,3803
1250 DATA 194,32,134,48,173,68,71,240,
3,32,46,68,173,179,72,240,7136
1260 DATA 3,32,225,38,76,201,37,201,31
240,3,76,96,39,164,155,5965
1270 DATA 192,63,176,42,173,188,72,24,
121,203,75,201,41,176,32,72,5958
1280 DATA 173,188,72,133,146,104,141,1
88,72,230,155,165,157,133,145,165,1773
1290 DATA 158,32,39,45,32,104,42,164,1
55,185,10,76,133,144,96,230,7331
1300 DATA 155,169,2,164,155,24,121,202
75,201,40,176,4,136,76,31,4666
1310 DATA 39,200,132,143,32,255,41,32,
1,45,169,2,164,143,196,155,6846
1320 DATA 240,9,24,121,202,75,200,196,
155,208,247,133,146,24,121,202,1631
1330 DATA 75,141,188,72,165,155,201,63
208,9,165,146,24,109,9,76,4923
1340 DATA 141,188,72,76,14,39,201,30,2
08,68,165,146,201,2,208,5,6288
1350 DATA 165,143,208,30,96,198,155,16
5,157,133,145,165,158,32,39,45,6743
1360 DATA 165,146,72,56,164,155,249,20
2,75,133,146,104,141,188,72,76,8986
1370 DATA 14,39,198,155,165,155,133,14
3,32,255,41,32,1,45,169,2,3389
1380 DATA 133,146,164,143,24,121,202,7
5,141,188,72,76,14,39,201,28,4773
1390 DATA 208,90,165,157,208,5,165,156
208,13,96,32,104,42,198,157,7238
1400 DATA 198,147,198,158,76,104,42,32
104,42,198,156,198,158,169,208,727
1410 DATA 133,134,169,34,133,135,169,2
48,133,128,169,34,133,129,162,16,8061
1420 DATA 32,111,40,56,233,40,133,134,
165,135,233,0,133,135,202,16,7729
1430 DATA 239,162,39,169,0,157,80,32,2
02,16,250,133,145,169,2,133,7277
1440 DATA 192,165,156,32,39,45,32,171,
40,76,104,42,201,29,240,1,4018
1450 DATA 96,165,156,24,101,157,201,63
144,1,96,165,157,201,17,176,8066
1460 DATA 12,32,104,42,230,157,230,147
230,158,76,104,42,32,104,42,5854
1470 DATA 230,156,230,158,169,120,133,
134,169,32,133,135,169,80,133,128,9186
1480 DATA 169,32,133,129,162,16,32,111
40,24,105,40,133,134,165,135,5121
1490 DATA 105,0,133,135,202,16,239,169
0,162,39,157,248,34,202,16,7584
1500 DATA 250,169,2,133,192,169,17,133
145,24,101,156,32,39,45,32,2460
1510 DATA 171,40,76,104,42,160,39,177,
134,145,128,136,16,249,165,135,9289
1520 DATA 133,129,165,134,133,128,96,1
34,175,169,12,157,66,3,32,86,4156
1530 DATA 228,166,175,96,32,171,40,76,
151,40,32,255,41,32,210,44,5079
1540 DATA 96,133,133,134,132,177,132,2
01,155,240,5,145,128,200,208,245,4185
1550 DATA 96,160,0,132,145,165,156,133
190,230,190,165,190,162,0,201,2399
1560 DATA 10,144,6,56,233,10,232,208,2
46,72,138,72,164,145,185,180,1746
1570 DATA 69,133,128,185,198,69,133,12
9,104,9,144,160,0,145,128,104,6540
1580 DATA 200,9,144,145,128,230,190,23
0,145,165,145,201,18,208,204,96,2704
1590 DATA 216,169,80,141,180,69,133,12
8,169,32,141,198,69,133,129,160,9239
1600 DATA 1,165,128,24,105,40,133,128,
153,180,69,165,129,105,0,133,6322
1610 DATA 129,153,198,69,200,192,18,20
8,232,169,100,141,0,2,169,37,6582
1620 DATA 141,1,2,169,2,141,200,2,169,
0,141,198,2,169,10,141,4963
1630 DATA 197,2,169,6,162,37,160,125,3
2,92,228,169,192,141,14,212,9330

1640 DATA 160,63,169,8,153,202,75,169,
32,153,10,76,136,16,243,133,6924
1650 DATA 144,169,0,141,201,75,162,209
149,0,202,48,251,133,147,141,682
1660 DATA 69,71,141,68,71,169,2,133,14
6,24,109,202,75,141,188,72,7111
1670 DATA 165,121,5,122,240,10,165,121
141,189,42,165,122,141,190,42,8646
1680 DATA 169,127,168,170,138,153,201,
74,73,128,153,73,75,202,136,48,8001
1690 DATA 16,192,95,208,4,162,63,208,2
35,192,31,208,231,162,95,208,3200
1700 DATA 227,169,202,133,128,169,76,1
33,129,160,0,165,128,153,74,76,7526
1710 DATA 165,129,200,153,74,76,165,12
8,24,105,1,133,128,165,129,105,6812
1720 DATA 0,133,129,200,192,128,208,22
7,160,63,169,255,153,202,76,136,3295
1730 DATA 16,250,169,10,141,189,72,169
77,141,190,72,169,87,133,128,9003
1740 DATA 141,193,72,169,73,133,129,14
1,251,72,160,1,165,128,24,105,7081
1750 DATA 6,133,128,153,193,72,165,129
105,0,133,129,153,251,72,200,341
1760 DATA 192,57,208,232,96,169,40,133
128,169,32,133,129,160,39,169,8452
1770 DATA 128,145,128,136,16,249,169,2
133,138,165,143,133,139,164,139,356
1780 DATA 192,64,240,22,185,202,75,72,
74,24,101,138,133,141,198,141,8748
1790 DATA 104,24,101,138,133,140,201,4
1,144,1,96,169,33,133,142,165,7488
1800 DATA 139,201,26,144,14,230,142,56
233,26,201,26,144,5,230,142,8431
1810 DATA 56,233,26,72,164,141,165,142
73,128,145,128,104,24,105,33,5514
1820 DATA 73,128,200,145,128,230,139,1
65,140,133,138,76,24,42,32,133,6170
1830 DATA 42,172,188,72,136,177,128,73
128,145,128,136,196,146,16,245,684
1840 DATA 164,155,185,10,76,133,144,32
7,64,96,164,147,185,180,69,7383
1850 DATA 133,128,185,198,69,133,129,9
6,160,0,152,153,32,35,200,192,8326
1860 DATA 160,208,248,169,128,141,34,3
5,96,32,172,42,162,255,142,252,1278
1870 DATA 2,96,173,252,2,201,255,240,2
49,133,184,168,192,192,144,2,3499
1880 DATA 160,154,185,254,254,133,185,
201,130,208,15,173,190,2,73,64,8998
1890 DATA 141,190,2,169,255,141,252,2,
208,216,201,129,208,11,173,182,3150
1900 DATA 2,73,128,141,182,2,76,205,42
201,131,240,222,201,132,240,4319
1910 DATA 193,201,133,240,189,165,184,
201,64,176,20,165,185,201,97,144,2209
1920 DATA 14,201,123,176,10,173,190,2,
240,5,5,184,76,181,42,165,7108
1930 DATA 185,162,15,221,176,70,240,6,
202,16,248,77,182,2,96,160,8420
1940 DATA 1,208,6,160,2,208,2,160,3,18
5,216,69,133,128,185,220,464
1950 DATA 69,133,129,160,1,96,160,2,32
140,52,76,184,46,173,189,6856
1960 DATA 72,141,151,43,24,101,154,141
154,43,173,190,72,141,152,43,7748
1970 DATA 105,0,141,155,43,197,106,176
221,173,189,72,56,229,130,133,1339
1980 DATA 152,173,190,72,229,131,133,1
53,173,189,72,24,101,154,141,189,557
1990 DATA 72,173,190,72,105,0,141,190,
72,32,150,43,144,1,96,206,6054
2000 DATA 151,43,173,151,43,201,255,20
8,3,206,152,43,206,154,43,173,752
2010 DATA 154,43,201,255,208,3,206,155
43,76,115,43,173,255,255,141,2441
2020 DATA 255,255,198,152,165,152,201,
255,208,10,198,153,165,153,201,255,653
9
2030 DATA 208,2,56,96,24,96,165,130,14
1,154,43,24,101,154,141,151,7255
2040 DATA 43,165,131,141,155,43,105,0,
141,152,43,173,189,72,56,237,8573
2050 DATA 151,43,133,152,173,190,72,23
7,152,43,133,153,173,189,72,56,9667
2060 DATA 229,154,141,189,72,173,190,7
2,233,0,141,190,72,32,150,43,7330

2070 DATA 144,1,96,238,151,43,208,3,23
8,152,43,238,154,43,208,3,8712
2080 DATA 238,155,43,76,231,43,134,175
170,189,201,74,166,175,96,165,2019
2090 DATA 155,133,150,165,156,24,101,1
57,133,151,10,168,185,74,76,133,8050
2100 DATA 130,200,185,74,76,133,131,16
0,0,177,130,16,93,165,193,24,7046
2110 DATA 105,4,133,154,32,56,43,160,0
,165,150,145,130,165,144,160,9075
2120 DATA 1,145,130,200,165,193,145,13
0,200,169,0,5,194,145,130,32,8203
2130 DATA 249,59,160,0,185,192,36,145,
130,200,196,193,208,246,165,193,5819
2140 DATA 24,105,4,133,154,165,151,32,
101,44,96,10,168,200,200,76,7716
2150 DATA 127,44,185,74,76,24,101,154,
153,74,76,200,185,74,76,105,7293
2160 DATA 0,153,74,76,200,192,128,208,
233,96,197,150,208,6,165,151,2407
2170 DATA 32,165,44,56,176,151,160,2,1
77,130,24,101,130,133,130,165,8321
2180 DATA 131,105,0,133,131,32,249,59,
76,33,44,72,160,2,177,130,5320
2190 DATA 24,105,4,133,154,32,176,43,1
04,10,168,200,200,76,205,44,8233
2200 DATA 185,74,76,56,229,154,153,74,
76,200,185,74,76,233,0,153,9022
2210 DATA 74,76,200,192,128,208,233,96
,162,17,32,219,44,202,16,250,339
2220 DATA 96,189,180,69,133,128,189,19
8,69,133,129,169,0,160,2,145,7838
2230 DATA 128,200,192,40,208,249,96,16
4,143,24,121,202,75,201,40,176,499
2240 DATA 3,200,208,245,132,140,96,169
,0,133,145,165,156,133,139,165,1291
2250 DATA 139,32,39,45,230,139,230,145
,165,145,201,18,208,241,96,10,544
2260 DATA 168,185,74,76,133,130,200,18
5,74,76,133,131,96,72,165,143,9129
2270 DATA 133,138,169,2,133,192,32,241
,44,166,145,32,219,44,104,32,6733
2280 DATA 25,45,160,0,177,130,201,255,
240,24,197,138,240,27,144,19,172
2290 DATA 164,138,185,202,75,24,101,19
2,133,192,230,138,165,138,197,140,3430
2300 DATA 144,224,96,32,242,59,76,60,4
5,165,130,24,105,4,133,132,5185
2310 DATA 165,131,105,0,133,133,164,14
5,185,180,69,133,134,185,198,69,1049
2320 DATA 133,135,165,192,133,141,160,
1,177,130,133,144,160,3,177,130,9566
2330 DATA 16,3,76,33,46,41,3,240,23,20
1,1,240,91,160,2,177,6522
2340 DATA 130,56,233,6,24,101,132,170,
165,133,105,0,168,76,246,45,7943
2350 DATA 32,145,46,160,2,177,130,168,
169,155,153,192,36,136,177,132,973
2360 DATA 153,192,36,136,16,248,169,19
2,133,132,169,36,133,133,32,25,7115
2370 DATA 54,164,138,185,202,75,133,14
9,160,0,132,142,164,142,185,128,1028
2380 DATA 5,32,0,44,164,141,145,134,23
0,142,230,141,165,142,197,149,3562
2390 DATA 208,234,32,145,46,76,74,45,1
66,132,164,133,32,137,221,32,7681
2400 DATA 79,46,32,230,216,32,201,52,7
6,203,45,169,1,133,194,32,7216
2410 DATA 66,46,32,68,218,32,0,216,162
,192,160,36,32,167,221,169,289
2420 DATA 6,133,193,32,9,44,96,162,0,1
69,5,133,149,164,138,185,8280
2430 DATA 202,75,201,4,208,2,198,149,1
64,141,189,215,70,145,134,200,2726
2440 DATA 232,228,149,208,245,76,74,45
,169,36,133,244,169,192,133,243,4006
2450 DATA 169,0,133,242,96,169,5,133,2
44,169,128,133,243,169,0,133,1122
2460 DATA 242,96,32,79,46,32,230,216,1
60,0,177,243,48,6,153,192,9444
2470 DATA 36,200,208,246,41,127,153,19
2,36,200,132,193,192,1,240,22,642
2480 DATA 173,192,36,201,48,208,15,160
,0,185,193,36,153,192,36,200,9644
2490 DATA 196,193,208,245,198,193,96,1
62,127,169,32,157,192,36,202,16,147
2500 DATA 250,96,160,0,185,192,36,201,

155,240,18,41,127,201,123,176,970
2510 DATA 6,201,97,144,2,73,32,153,192
,36,200,208,231,96,32,210,652
2520 DATA 44,32,205,46,32,255,41,32,17
1,40,32,1,45,32,85,48,1039
2530 DATA 76,104,42,169,0,133,143,133,
156,133,158,133,155,133,157,133,907
2540 DATA 147,169,2,133,146,164,143,24
,121,202,75,141,188,72,96,32,7500
2550 DATA 146,42,160,26,185,246,70,153
,34,35,136,16,247,160,32,185,8714
2560 DATA 17,71,153,74,35,136,16,247,1
69,0,56,237,189,72,133,212,386
2570 DATA 165,106,237,190,72,133,213,3
2,170,217,32,230,216,160,0,177,2015
2580 DATA 243,72,41,127,32,0,44,153,61
,35,200,104,16,241,160,0,5619
2590 DATA 185,199,70,201,155,240,11,32
,0,44,153,39,35,200,192,16,5717
2600 DATA 208,238,32,129,62,76,146,42,
32,50,50,32,115,50,176,48,3345
2610 DATA 32,54,61,176,43,32,115,50,16
2,0,185,192,35,157,192,35,7139
2620 DATA 201,155,240,4,200,232,208,24
2,162,3,189,192,35,32,0,44,7129
2630 DATA 157,197,74,202,16,244,165,15
0,141,178,72,165,151,141,177,72,1675
2640 DATA 96,32,121,48,162,1,189,199,7
4,157,173,35,189,197,74,157,482
2650 DATA 153,35,202,16,241,165,156,24
,101,157,32,25,45,160,0,177,5810
2660 DATA 130,201,255,240,7,205,178,72
,240,11,144,3,76,242,47,32,7487
2670 DATA 242,59,76,151,47,160,3,177,1
30,41,3,240,31,201,1,208,7547
2680 DATA 235,32,111,48,32,230,216,160
,0,177,243,72,41,127,32,0,5934
2690 DATA 44,153,156,35,104,48,33,200,
192,14,208,237,160,2,177,130,130
2700 DATA 133,176,32,249,59,160,0,177,
130,32,0,44,153,156,35,200,6864
2710 DATA 196,176,240,4,192,14,208,239
,173,177,72,32,25,45,165,155,8819
2720 DATA 133,138,160,0,177,130,201,25
5,240,6,197,138,240,9,144,1,186
2730 DATA 96,32,242,59,76,252,47,160,3
,177,130,41,3,240,31,201,8152
2740 DATA 1,208,237,32,111,48,32,230,2
16,160,0,177,243,72,41,127,9385
2750 DATA 32,0,44,153,176,35,104,48,33
,200,192,14,208,237,160,2,8759
2760 DATA 177,130,133,176,32,249,59,16
0,0,177,130,32,0,44,153,176,6958
2770 DATA 35,200,196,176,240,4,192,14,
208,239,96,166,155,165,158,32,1694
2780 DATA 151,69,144,6,169,3,133,194,2
08,34,160,3,177,130,41,3,5663
2790 DATA 133,194,76,134,48,32,249,59,
166,130,164,131,76,137,221,160,1673
2800 DATA 119,169,0,153,32,35,200,192,
160,208,248,96,165,194,10,10,9746
2810 DATA 10,170,160,31,189,74,71,153,
0,32,232,200,192,39,208,244,2212
2820 DATA 96,32,210,44,169,49,141,193,
70,32,50,50,32,115,50,176,5148
2830 DATA 3,141,193,70,162,32,129,4
0,169,1,133,145,169,3,162,6777
2840 DATA 32,157,66,3,169,192,157,68,3
,169,70,157,69,3,169,6,4977
2850 DATA 157,74,3,169,0,157,75,3,32,8
6,228,16,6,32,140,52,2637
2860 DATA 76,196,46,162,32,169,5,157,6
6,3,169,128,157,68,3,169,6340
2870 DATA 5,157,69,3,169,0,157,73,3,16
9,20,157,72,3,32,86,2591
2880 DATA 228,16,11,32,129,62,162,32,3
2,129,40,76,196,46,32,52,3142
2890 DATA 49,162,0,160,11,189,128,5,20
1,155,240,10,32,0,44,145,5543
2900 DATA 128,200,232,76,15,49,230,145
,165,145,201,17,208,181,32,129,699
2910 DATA 62,32,210,44,169,0,133,145,2
40,169,164,145,185,180,69,133,2260
2920 DATA 128,185,198,69,133,129,96,32
,50,50,32,115,50,32,156,50,3575
2930 DATA 32,119,54,144,5,160,165,76,2
15,48,32,160,54,144,3,76,5352

2940 DATA 184,46,169,8,160,11,208,18,3
2,50,50,32,115,50,32,156,3114
2950 DATA 50,32,119,54,176,223,169,4,1
60,7,133,173,132,174,162,32,9213
2960 DATA 32,129,40,162,32,165,173,157
74,3,169,3,157,66,3,169,5940
2970 DATA 199,157,68,3,169,70,157,69,3
169,0,157,75,3,32,86,3089
2980 DATA 228,16,3,76,215,48,162,32,16
5,174,168,157,66,3,169,202,9930
2990 DATA 157,68,3,169,75,157,69,3,192
7,208,10,169,255,157,72,9272
3000 DATA 3,157,73,3,208,17,173,189,72
56,233,202,157,72,3,173,9470
3010 DATA 190,72,233,75,157,73,3,32,86
228,16,7,192,136,240,3,7205
3020 DATA 76,215,48,162,32,32,129,40,1
65,173,201,8,240,19,162,32,7743
3030 DATA 169,202,24,125,72,3,141,189,
72,169,75,125,73,3,141,190,7861
3040 DATA 72,76,184,46,32,50,50,32,115
50,176,43,32,54,61,176,3956
3050 DATA 38,165,151,201,47,176,32,165
150,201,55,176,26,32,104,42,6605
3060 DATA 32,210,44,165,150,133,143,13
3,155,165,151,133,156,133,158,169,3183
3070 DATA 0,32,215,46,76,190,46,96,160
114,185,192,36,153,192,35,9542
3080 DATA 136,16,247,96,32,46,68,76,18
4,46,169,1,141,187,72,96,6947
3090 DATA 165,155,170,165,156,24,101,1
57,133,138,32,151,69,144,1,96,6827
3100 DATA 165,138,32,165,44,32,104,42,
169,2,133,192,165,157,133,145,9739
3110 DATA 165,158,32,39,45,32,104,42,9
6,160,0,185,192,35,201,155,8489
3120 DATA 240,9,201,32,240,7,200,192,3
2,208,240,56,96,200,185,192,3862
3130 DATA 35,201,155,240,246,201,32,20
8,7,200,192,32,208,240,240,235,7299
3140 DATA 24,96,162,0,185,192,35,157,1
99,70,201,155,240,4,200,232,3870
3150 DATA 208,242,96,32,50,50,32,115,5
0,176,49,32,54,61,176,44,3985
3160 DATA 230,151,165,151,201,2,144,36
201,33,176,32,164,150,153,202,1322
3170 DATA 75,32,210,44,32,255,41,32,1,
45,164,143,132,155,169,2,6780
3180 DATA 133,146,24,121,202,75,141,18
8,72,32,104,42,96,166,155,165,9265
3190 DATA 158,32,151,69,144,1,96,160,1
177,130,133,0,200,177,130,9159
3200 DATA 133,154,200,177,130,41,3,208
22,32,249,59,164,154,136,177,655
3210 DATA 130,153,192,36,136,16,248,16
9,255,133,148,32,48,62,96,201,15
3220 DATA 2,240,29,32,249,59,166,130,1
64,131,32,137,221,32,230,216,2727
3230 DATA 160,0,177,243,72,41,127,153,
192,36,104,48,218,200,208,242,4046
3240 DATA 165,154,56,233,6,133,154,76,
3,51,169,1,133,191,32,231,8314
3250 DATA 50,32,145,62,8,169,0,133,191
40,144,1,96,104,104,165,6784
3260 DATA 0,133,144,76,34,38,32,146,42
164,155,185,10,76,133,144,7488
3270 DATA 162,18,202,48,31,189,64,72,1
57,34,35,224,5,144,243,189,155
3280 DATA 77,72,157,74,35,189,90,72,15
7,114,35,189,103,72,157,154,9155
3290 DATA 35,76,108,51,76,243,51,173,2
52,2,201,255,240,249,162,255,8259
3300 DATA 142,252,2,201,28,208,1,96,16
2,2,221,136,72,240,5,202,63
3310 DATA 16,248,48,14,165,144,41,252,
133,176,138,5,176,133,144,76,75
3320 DATA 243,51,201,58,208,9,165,144,
73,8,133,144,76,243,51,201,200
3330 DATA 32,208,9,165,144,73,4,133,14
4,76,243,51,162,9,221,165,243
3340 DATA 72,240,5,202,16,248,48,175,1
38,72,165,144,41,15,133,144,8566
3350 DATA 104,10,10,10,5,144,133,14
4,165,144,41,3,170,189,123,7940
3360 DATA 72,141,51,35,165,144,41,8,74
74,74,170,189,121,72,141,7794
3370 DATA 91,35,165,144,41,4,74,74,170

,189,121,72,141,131,35,165,8388
3380 DATA 144,41,240,74,74,74,170,1
89,126,72,141,171,35,76,145,8932
3390 DATA 51,32,96,51,164,155,165,144,
153,10,76,96,165,158,166,155,846
3400 DATA 32,151,69,144,1,96,32,96,51,
160,1,165,144,145,130,166,8648
3410 DATA 157,134,145,165,158,32,39,45
164,155,185,10,76,133,144,76,7952
3420 DATA 104,42,173,68,71,73,3,141,68
71,170,160,0,189,50,71,5985
3430 DATA 153,24,71,232,200,192,3,208,
244,96,173,69,71,73,6,141,8787
3440 DATA 69,71,170,160,0,189,56,71,15
3,44,71,232,200,192,6,208,983
3450 DATA 244,96,132,177,162,32,32,129
40,164,177,132,212,169,0,133,29
3460 DATA 213,32,170,217,32,230,216,16
0,255,200,177,243,72,41,127,32,2532
3470 DATA 0,44,153,231,70,104,16,241,3
2,121,48,160,20,185,225,70,9246
3480 DATA 153,156,35,136,16,247,169,25
5,141,252,2,32,106,62,96,160,9820
3490 DATA 255,200,177,243,153,0,5,16,2
48,160,127,169,48,153,192,36,55
3500 DATA 136,16,250,169,36,141,192,36
169,193,133,132,169,36,133,133,1074
3510 DATA 165,144,72,41,240,74,74,74,7
4,141,176,72,104,41,4,208,7287
3520 DATA 12,160,255,200,177,243,145,1
32,16,249,76,99,53,160,255,200,4443
3530 DATA 185,0,5,48,5,201,46,208,246,
136,132,179,162,29,169,2,9392
3540 DATA 141,175,72,185,0,5,41,127,15
7,128,5,202,136,48,13,206,7423
3550 DATA 175,72,16,239,169,44,157,128
5,202,16,226,232,160,1,189,1353
3560 DATA 128,5,153,192,36,200,232,224
30,208,244,166,179,189,0,5,1800
3570 DATA 16,11,136,185,192,36,9,128,1
53,192,36,208,12,232,189,0,9447
3580 DATA 5,153,192,36,48,3,200,208,24
4,165,144,41,8,240,8,169,143
3590 DATA 192,133,132,169,36,133,133,1
69,0,141,113,54,160,255,200,185,3292
3600 DATA 192,36,48,46,201,69,208,246,
238,113,54,136,185,192,36,9,310
3610 DATA 128,153,192,36,162,255,232,2
00,185,192,36,72,41,127,157,114,1663
3620 DATA 54,169,48,153,192,36,104,16,
237,232,142,113,54,169,155,157,2076
3630 DATA 114,54,160,255,200,185,192,3
6,16,16,41,127,153,192,36,200,9790
3640 DATA 169,46,153,192,36,132,179,76
213,53,201,46,208,230,132,179,3939
3650 DATA 200,185,192,36,16,250,41,127
153,192,36,165,179,24,109,176,610
3660 DATA 72,168,173,176,72,208,1,136,
169,155,153,193,36,173,113,54,513
3670 DATA 240,16,162,0,189,114,54,153,
193,36,201,155,240,4,232,200,3683
3680 DATA 208,242,173,193,36,201,45,20
8,22,173,194,36,201,44,208,15,111
3690 DATA 160,0,185,195,36,153,194,36,
201,155,240,3,200,208,243,160,5681
3700 DATA 255,200,177,132,201,155,208,
249,132,179,162,127,169,32,157,128,412
9
3710 DATA 5,202,16,250,165,144,41,3,17
0,240,36,202,208,14,164,138,1485
3720 DATA 185,202,75,56,229,179,16,22,
162,0,240,21,164,138,185,202,1686
3730 DATA 75,56,229,179,16,7,73,255,74
168,76,66,54,74,170,160,8815
3740 DATA 255,200,177,132,201,155,240,
6,157,128,5,232,208,243,169,128,5338
3750 DATA 133,132,169,5,133,133,96,0,1
18,54,249,61,155,173,199,70,424
3760 DATA 201,68,208,23,173,200,70,201
58,240,14,201,56,240,14,201,1978
3770 DATA 49,144,8,201,52,176,4,144,4,
24,96,56,96,173,201,70,7240
3780 DATA 201,58,240,245,208,245,169,0
141,34,35,162,32,32,129,40,6867
3790 DATA 169,36,157,66,3,169,70,157,6
9,3,169,199,157,68,3,32,5989
3800 DATA 86,228,189,67,3,72,162,32,32

,129,40,104,201,1,240,2,6539
3810 DATA 24,96,169,72,162,139,160,1,3
2,48,69,173,252,2,162,255,1010
3820 DATA 201,43,240,236,201,35,208,24
3,142,252,2,56,96,169,0,133,620
3830 DATA 180,32,68,218,162,0,134,193,
189,192,36,157,192,35,201,155,2824
3840 DATA 240,3,232,208,243,134,193,16
2,1,189,192,35,201,64,240,8,1948
3850 DATA 232,228,193,208,244,76,101,5
8,134,138,169,56,133,131,169,86,1387
3860 DATA 133,130,232,134,139,160,0,17
7,130,240,234,48,18,221,192,35,2044
3870 DATA 208,4,232,200,208,241,169,4,
32,251,59,166,139,208,230,41,3547
3880 DATA 127,221,192,35,208,240,160,3
,177,130,133,176,232,134,139,10,2394
3890 DATA 170,189,127,56,141,246,55,18
9,128,56,141,247,55,169,0,133,557
3900 DATA 178,164,139,185,192,35,201,4
0,240,2,56,96,200,185,192,35,1038
3910 DATA 201,68,176,246,201,65,176,35
,132,242,132,139,32,241,61,32,611
3920 DATA 0,216,176,230,32,254,55,176,
225,164,242,185,192,35,201,41,4481
3930 DATA 208,3,76,206,55,201,44,240,2
11,208,207,132,139,32,54,61,786
3940 DATA 176,200,164,178,165,150,153,
159,0,165,151,153,163,0,165,176,2345
3950 DATA 201,7,240,9,166,150,165,151,
32,22,56,176,227,164,178,169,2871
3960 DATA 128,153,198,0,32,254,55,176,
215,165,139,24,105,4,133,139,9507
3970 DATA 168,76,133,55,200,132,140,16
4,138,166,140,32,206,61,162,0,9781
3980 DATA 189,192,35,201,155,240,3,232
,208,246,134,193,32,221,61,164,4725
3990 DATA 138,169,35,153,192,35,165,19
5,153,193,35,32,255,255,144,1,2205
4000 DATA 96,76,1,55,165,180,164,178,1
53,195,0,230,178,200,192,3,3077
4010 DATA 176,8,170,32,199,61,230,180,
24,96,56,96,134,150,32,25,6917
4020 DATA 45,160,0,177,130,201,255,240
,6,197,150,240,11,144,2,56,293
4030 DATA 96,32,242,59,76,27,56,96,160
,3,177,130,41,3,240,26,6422
4040 DATA 201,1,240,10,160,2,177,130,5
6,233,6,32,251,59,169,4,8066
4050 DATA 32,251,59,166,130,164,131,32
,137,221,24,96,83,81,213,0,8499
4060 DATA 65,66,211,1,83,81,210,2,82,7
8,196,3,69,88,208,4,6304
4070 DATA 76,79,199,5,73,78,212,6,67,7
8,212,7,83,85,205,8,6557
4080 DATA 65,86,197,9,0,245,56,147,56,
207,56,14,57,166,57,175,8060
4090 DATA 57,163,56,45,57,61,57,132,57
,32,248,57,165,212,41,127,8932
4100 DATA 133,212,166,195,32,199,61,24
,96,169,0,141,250,61,32,248,706
4110 DATA 57,32,79,46,32,230,216,32,45
,58,165,212,72,32,0,216,8144
4120 DATA 104,16,15,173,250,61,240,10,
162,179,160,74,32,152,221,32,636
4130 DATA 96,218,76,156,56,32,248,57,1
62,36,160,62,32,167,221,32,9081
4140 DATA 209,222,162,185,160,74,32,15
2,221,32,40,219,144,1,96,32,7918
4150 DATA 204,221,176,6,32,251,61,76,2
04,56,96,32,248,57,32,209,9731
4160 DATA 222,162,185,160,74,32,152,22
1,32,219,218,144,1,96,32,204,1000
4170 DATA 221,144,228,96,32,248,57,162
,191,160,74,32,152,221,32,102,789
4180 DATA 218,144,1,96,32,230,216,32,4
5,58,32,79,46,32,0,216,5168
4190 DATA 76,156,56,32,4,58,32,11,58,3
2,22,58,32,184,57,76,2397
4200 DATA 156,56,96,32,4,58,32,11,58,3
2,22,58,32,79,57,176,2537
4210 DATA 3,76,156,56,96,32,35,58,32,6
8,218,165,167,133,169,32,8864
4220 DATA 182,221,166,169,165,171,32,2
2,56,176,7,240,7,32,102,218,9047
4230 DATA 144,2,56,96,230,169,165,169,
197,168,240,227,144,225,230,171,9626

4240 DATA 165,171,197,172,240,213,144,
211,24,96,32,4,58,32,11,58,5071
4250 DATA 32,22,58,32,184,57,176,19,32
,73,58,32,79,57,176,11,3955
4260 DATA 32,87,58,32,40,219,176,3,76,
156,56,96,32,248,57,32,6805
4270 DATA 204,221,144,244,96,32,248,57
,32,209,222,144,245,96,32,35,1305
4280 DATA 58,169,0,133,173,133,174,165
,167,133,169,166,169,165,171,32,3677
4290 DATA 22,56,176,6,230,173,208,2,23
0,174,230,169,165,169,197,168,7158
4300 DATA 240,233,144,231,230,171,165,
171,197,172,240,219,144,217,32,68,6300
4310 DATA 218,165,173,133,212,165,174,
133,213,32,170,217,24,96,165,178,3931
4320 DATA 240,4,201,1,240,3,104,104,56
,96,165,178,201,2,208,246,3065
4330 DATA 96,160,1,185,198,0,16,238,13
6,16,248,96,165,164,197,163,3763
4340 DATA 144,228,165,160,197,159,144,
222,96,162,7,181,159,149,167,202,5421
4350 DATA 16,249,96,160,0,177,243,48,1
5,201,46,240,3,200,208,245,4349
4360 DATA 141,250,61,169,155,145,243,9
6,41,127,145,243,200,208,244,166,7952
4370 DATA 180,189,251,72,168,189,193,7
2,170,32,167,221,96,166,180,189,5505
4380 DATA 251,72,168,189,193,72,170,32
,152,221,96,160,1,185,192,35,1475
4390 DATA 201,68,176,4,201,65,176,8,20
0,196,193,208,240,76,254,58,5263
4400 DATA 132,138,152,24,105,4,133,140
,32,54,61,176,19,165,151,32,6905
4410 DATA 25,45,160,0,177,130,201,255,
240,6,197,150,240,9,144,1,1726
4420 DATA 96,32,242,59,76,140,58,160,3
,177,130,41,3,240,241,201,2397
4430 DATA 1,240,10,160,2,177,130,56,23
3,6,32,251,59,169,4,32,7237
4440 DATA 251,59,166,130,164,131,32,13
7,221,169,4,133,139,32,207,58,360
4450 DATA 164,138,76,114,58,164,138,16
6,140,32,206,61,165,193,56,229,3071
4460 DATA 139,133,193,32,221,61,164,13
8,169,35,153,192,35,165,180,200,3818
4470 DATA 153,192,35,132,138,170,189,2
51,72,168,189,193,72,170,32,167,3872
4480 DATA 221,230,180,96,160,1,185,192
,35,201,155,240,53,201,40,240,4575
4490 DATA 13,201,41,240,9,201,35,208,9
,200,200,76,60,59,200,76,163
4500 DATA 0,59,132,242,132,138,32,241,
61,32,0,216,176,41,166,242,2307
4510 DATA 138,133,140,56,229,138,133,1
39,32,207,58,164,138,200,196,193,5160
4520 DATA 240,27,185,192,35,201,155,24
0,20,162,4,221,82,73,240,9,851
4530 DATA 202,16,248,201,41,240,230,56
,96,200,76,0,59,160,0,185,9682
4540 DATA 192,35,201,155,240,7,201,40,
240,44,200,208,242,160,0,185,4636
4550 DATA 192,35,201,155,240,9,201,41,
240,42,200,208,242,240,37,162,5953
4560 DATA 0,189,193,35,157,64,36,201,1
55,240,5,232,200,76,123,59,1773
4570 DATA 32,7,60,176,15,96,200,185,19
2,35,201,41,240,8,201,155,2539
4580 DATA 240,2,208,242,56,96,132,208,
136,185,192,35,201,40,208,248,5727
4590 DATA 132,207,162,255,164,207,232,
200,196,208,240,9,185,192,35,157,6650
4600 DATA 64,36,76,176,59,169,155,157,
64,36,32,7,60,176,213,165,9769
4610 DATA 208,56,229,207,201,1,240,204
,166,207,173,64,36,157,192,35,3109
4620 DATA 173,65,36,232,157,192,35,164
,208,232,200,185,192,35,157,192,6478
4630 DATA 35,201,155,208,244,76,87,59,
160,2,177,130,32,251,59,169,1607
4640 DATA 4,24,101,130,133,130,165,131
,105,0,133,131,96,160,0,132,8743
4650 DATA 142,32,164,60,201,155,240,35
,144,1,96,32,221,60,165,183,1345
4660 DATA 201,94,208,14,32,19,61,176,2
41,32,249,60,32,197,61,76,9091
4670 DATA 7,60,165,142,24,105,3,168,76

,9,60,160,0,132,142,32,5819
4680 DATA 164,60,201,155,240,48,144,1,
96,32,221,60,165,183,201,42,1420
4690 DATA 208,7,32,219,218,176,241,144
,9,201,47,208,16,32,40,219,733
4700 DATA 176,230,32,249,60,32,197,61,
164,142,76,55,60,165,142,24,8783
4710 DATA 105,3,168,76,55,60,160,0,132
,142,32,164,60,201,155,240,2071
4720 DATA 39,144,1,96,32,221,60,165,18
3,201,43,208,7,32,102,218,843
4730 DATA 176,241,144,9,201,45,208,235
,32,96,218,176,230,32,249,60,4358
4740 DATA 32,197,61,164,142,76,114,60,
24,96,185,64,36,201,35,240,75
4750 DATA 6,201,155,240,44,56,96,200,1
85,64,36,201,155,240,246,133,5717
4760 DATA 181,200,185,64,36,133,183,20
1,155,240,22,200,185,64,36,201,3513
4770 DATA 155,240,14,201,35,208,222,20
0,185,64,36,201,155,240,214,133,6654
4780 DATA 182,24,96,166,181,32,238,60,
32,141,221,166,182,32,238,60,2670
4790 DATA 32,156,221,96,189,193,72,133
,252,189,251,72,133,253,96,165,6986
4800 DATA 142,24,105,2,168,24,105,3,17
0,189,64,36,153,64,36,201,8253
4810 DATA 155,240,5,200,232,76,3,61,96
,166,181,32,238,60,32,141,9954
4820 DATA 221,32,209,222,144,2,56,96,1
66,182,32,238,60,32,156,221,2138
4830 DATA 32,219,218,176,241,32,204,22
1,176,236,24,96,32,123,61,176,2480
4840 DATA 5,32,79,61,144,1,96,164,205,
200,32,123,61,176,247,32,1247
4850 DATA 148,61,164,205,96,165,203,16
2,2,221,56,73,240,5,202,16,815
4860 DATA 248,48,15,189,59,73,133,141,
165,204,201,91,176,4,201,65,2024
4870 DATA 176,2,56,96,56,233,65,24,101
,141,133,150,201,64,176,242,3530
4880 DATA 96,185,192,35,201,155,240,16
,133,203,200,185,192,35,201,155,5933
4890 DATA 240,6,133,204,132,205,24,96,
56,96,165,203,32,180,61,176,1565
4900 DATA 23,133,151,165,204,32,180,61
,176,14,138,24,101,151,170,202,2001
4910 DATA 224,64,176,4,134,151,24,96,5
6,96,162,9,221,62,73,240,348
4920 DATA 5,202,16,248,56,96,189,72,73
,24,96,166,181,32,238,60,400
4930 DATA 32,171,221,96,189,192,35,153
,192,35,201,155,240,4,200,232,5954
4940 DATA 208,242,96,166,193,189,192,3
5,157,194,35,202,228,138,240,245,8885
4950 DATA 176,243,230,193,230,193,96,1
69,35,133,244,169,192,133,243,96,7302
4960 DATA 251,61,35,62,32,182,221,162,
42,160,62,32,167,221,162,36,1096
4970 DATA 160,62,32,137,221,32,40,219,
162,42,160,62,32,152,221,32,9980
4980 DATA 102,218,162,191,160,74,32,15
2,221,32,219,218,96,48,62,179,2500
4990 DATA 69,169,0,133,139,168,170,134
,140,166,139,189,216,69,24,105,2425
5000 DATA 2,133,128,189,220,69,105,0,1
33,129,166,140,189,192,36,32,847
5010 DATA 0,44,228,148,208,2,73,128,14
5,128,232,200,192,38,208,236,6422
5020 DATA 160,0,230,139,165,139,201,3,
208,205,96,173,31,208,201,6,2996
5030 DATA 208,16,104,104,169,155,141,1
92,36,169,0,133,191,141,201,75,2578
5040 DATA 56,96,173,252,2,201,255,240,
226,32,163,42,166,148,96,32,2569
5050 DATA 145,46,169,0,133,148,32,48,6
2,32,106,62,166,148,164,191,489
5060 DATA 208,30,172,201,75,208,25,160
,4,217,180,72,240,5,136,16,262
5070 DATA 248,48,20,141,179,72,201,155
,208,70,140,179,72,76,255,62,3089
5080 DATA 201,155,208,3,76,255,62,201,
27,208,10,32,106,62,201,155,737
5090 DATA 240,46,76,222,62,160,12,217,
76,70,240,18,136,16,248,157,2092
5100 DATA 192,36,232,134,148,224,114,1
44,173,202,134,148,208,168,185,50,6088

5110 DATA 70,141,250,62,185,63,70,141,
251,62,32,255,255,76,149,62,3278
5120 DATA 160,114,136,185,192,36,201,3
2,240,248,169,155,200,153,192,36,6052
5130 DATA 132,193,169,0,133,191,24,96,
32,145,46,169,0,133,148,96,8792
5140 DATA 166,148,240,8,202,169,32,157
,192,36,134,148,96,166,148,240,4828
5150 DATA 251,202,134,148,96,166,148,2
24,113,176,241,232,134,148,96,166,7225
5160 DATA 148,189,193,36,157,192,36,23
2,224,122,208,245,96,162,122,189,7288
5170 DATA 192,36,157,193,36,202,228,14
8,16,245,232,169,32,157,192,36,4093
5180 DATA 141,50,37,96,165,148,56,233,
38,16,3,24,105,114,133,148,8032
5190 DATA 96,165,148,24,105,38,201,114
,144,244,56,233,114,76,109,63,1565
5200 DATA 162,0,165,148,201,38,144,8,1
62,38,201,76,144,2,162,76,9261
5210 DATA 134,148,96,32,127,63,169,37,
133,141,32,76,63,198,141,16,8381
5220 DATA 249,96,32,127,63,169,37,133,
141,32,62,63,198,141,16,249,716
5230 DATA 96,169,4,141,124,37,169,1,14
1,201,75,32,142,62,176,47,8867
5240 DATA 32,156,46,162,0,169,89,133,1
28,169,70,133,129,160,0,177,1028
5250 DATA 128,240,222,217,192,36,208,3
4,200,192,3,208,242,189,111,70,5122
5260 DATA 141,236,63,189,118,70,141,23
7,63,32,146,42,32,255,255,169,4223
5270 DATA 0,141,124,37,141,201,75,76,2
01,37,232,165,128,24,105,3,9450
5280 DATA 133,128,144,201,230,129,208,
197,165,155,162,0,56,233,26,48,1826
5290 DATA 3,232,208,248,72,138,24,105,
161,141,19,32,104,24,105,187,9276
5300 DATA 141,20,32,166,158,232,138,16
2,0,56,233,10,48,3,232,208,1402
5310 DATA 248,72,138,24,105,144,141,21
,32,104,24,105,154,141,22,32,6426
5320 DATA 96,32,146,42,169,80,141,199,
70,169,58,141,200,70,169,155,3235
5330 DATA 141,201,70,169,72,162,3,160,
0,140,72,71,32,48,69,32,5071
5340 DATA 119,69,32,104,69,32,142,69,3
2,104,69,165,159,133,138,165,898
5350 DATA 160,56,229,159,48,7,165,164,
56,229,163,16,8,160,141,32,9444
5360 DATA 140,52,76,196,46,162,32,32,1
29,40,162,32,169,3,157,66,7675
5370 DATA 3,169,70,157,69,3,169,199,15
7,68,3,169,0,157,75,3,7212
5380 DATA 169,8,157,74,3,32,86,228,48,
213,166,138,189,202,75,133,3547
5390 DATA 149,165,163,32,151,69,176,93
,160,1,177,130,133,144,160,3,781
5400 DATA 177,130,16,3,76,98,65,41,3,2
40,3,76,115,65,160,2,5857
5410 DATA 177,130,168,32,249,59,169,15
5,153,192,36,136,48,0,177,130,1297
5420 DATA 153,192,36,76,218,64,169,192
,133,132,169,36,133,133,32,25,9841
5430 DATA 54,32,49,65,230,138,165,138,
197,160,240,174,144,172,165,159,7592
5440 DATA 133,138,32,26,65,230,163,165
,163,197,164,240,157,144,155,162,7310
5450 DATA 32,32,129,40,96,32,87,65,76,
240,64,169,11,162,32,157,9113
5460 DATA 66,3,169,0,157,72,3,157,73,3
,169,155,32,86,228,48,8747
5470 DATA 34,96,169,11,162,32,157,66,3
,165,149,157,72,3,169,0,7589
5480 DATA 157,73,3,169,5,157,69,3,169,
128,157,68,3,32,86,228,8711
5490 DATA 48,1,96,104,104,76,215,48,16
2,127,169,32,157,128,5,202,1220
5500 DATA 16,250,96,32,87,65,160,4,185
,220,70,153,128,5,136,16,8910
5510 DATA 247,76,240,64,201,1,208,19,3
2,249,59,166,130,164,131,32,1386
5520 DATA 137,221,32,230,216,32,201,52
,76,240,64,160,2,177,130,168,3060
5530 DATA 32,249,59,152,56,233,6,24,10
1,130,170,165,131,105,0,168,677
5540 DATA 76,126,65,32,50,50,32,115,50

,32,156,50,32,119,54,144,6167
 5550 DATA 3,76,79,49,32,160,54,144,3,7
 6,184,46,32,146,42,169,7945
 5560 DATA 72,162,34,76,86,64,32,146,42
 ,169,71,162,138,160,0,140,9635
 5570 DATA 70,71,140,72,71,32,48,69,32,
 119,69,32,104,69,32,142,5616
 5580 DATA 69,32,104,69,32,137,67,32,10
 4,69,165,160,197,159,176,3,844
 5590 DATA 76,124,64,165,164,197,163,14
 4,247,165,161,197,159,208,6,165,6778
 5600 DATA 165,197,163,240,235,165,161,
 197,160,208,6,165,165,197,164,240,9245
 5610 DATA 223,165,160,56,229,159,133,1
 71,133,169,165,164,56,229,163,133,6442
 5620 DATA 172,56,165,161,229,159,133,2
 02,56,165,165,229,163,133,201,32,5914
 5630 DATA 50,67,10,170,189,131,67,141,
 215,66,189,132,67,141,216,66,3251
 5640 DATA 32,104,42,165,159,133,173,16
 5,161,133,174,165,165,32,25,45,1151
 5650 DATA 160,0,177,130,201,255,240,19
 ,197,174,240,8,176,13,32,242,4299
 5660 DATA 59,76,79,66,166,174,165,165,
 32,165,44,165,130,133,132,165,3295
 5670 DATA 131,133,133,165,163,32,25,45
 ,160,0,177,130,201,255,240,87,4790
 5680 DATA 197,173,240,8,176,81,32,242,
 59,76,119,66,169,35,133,135,735
 5690 DATA 169,192,133,134,160,2,177,13
 0,24,105,4,168,136,133,154,132,1647
 5700 DATA 149,177,130,145,134,136,192,
 255,208,247,165,174,141,192,35,165,797
 4
 5710 DATA 132,133,130,165,133,133,131,
 173,70,71,240,3,32,169,67,164,1349
 5720 DATA 149,200,132,154,32,56,43,165
 ,165,32,101,44,164,149,177,134,1859
 5730 DATA 145,132,136,192,255,208,247,
 32,255,255,176,3,76,74,66,32,2166
 5740 DATA 46,68,32,1,45,32,104,42,24,9
 6,230,174,230,173,198,171,4415
 5750 DATA 165,171,201,255,208,26,230,1
 65,230,163,198,172,165,172,201,255,159
 6
 5760 DATA 208,2,56,96,165,159,133,173,
 165,161,133,174,165,169,133,171,6506
 5770 DATA 24,96,230,174,230,173,198,17
 1,165,171,201,255,208,242,198,165,2482
 5780 DATA 198,163,76,249,66,198,174,19
 8,173,198,171,165,171,201,255,208,1436
 5790 DATA 223,240,195,165,165,197,163,
 208,45,165,161,197,159,208,5,104,5458
 5800 DATA 104,76,124,64,144,28,165,160
 ,56,229,159,24,101,161,133,161,2976
 5810 DATA 201,64,176,235,165,160,72,16
 5,159,133,160,104,133,159,169,1,3491
 5820 DATA 24,96,169,0,24,96,144,250,16
 5,164,56,229,163,24,101,165,3346
 5830 DATA 133,165,201,64,176,201,165,1
 64,72,165,163,133,164,104,133,163,5583
 5840 DATA 169,2,24,96,233,66,36,67,17,
 67,169,2,133,176,169,71,9388
 5850 DATA 162,239,160,3,76,125,69,32,1
 46,42,169,1,133,176,141,70,9884
 5860 DATA 71,169,71,162,168,160,0,76,2
 09,65,160,4,185,192,35,201,2711
 5870 DATA 61,240,1,96,200,196,149,176,
 250,185,192,35,201,68,176,244,8166
 5880 DATA 201,65,144,240,140,73,71,32,
 54,61,144,5,172,73,71,208,9885
 5890 DATA 227,165,150,24,101,202,133,1
 50,201,64,176,215,165,151,24,101,4041
 5900 DATA 201,133,151,201,64,176,229,1
 72,73,71,162,2,165,150,221,59,3656
 5910 DATA 73,176,3,202,16,248,72,189,5
 6,73,153,192,35,104,56,253,2756
 5920 DATA 59,73,24,105,65,200,153,192,
 35,230,151,165,151,162,9,221,4907
 5930 DATA 72,73,176,3,202,16,248,72,18
 9,62,73,200,153,192,35,104,2496
 5940 DATA 56,253,72,73,170,189,62,73,2
 00,153,192,35,76,179,67,169,3077
 5950 DATA 0,133,151,133,150,173,69,71,
 208,29,32,111,68,230,150,165,2880
 5960 DATA 150,201,64,144,245,169,0,133
 ,150,230,151,165,151,201,64,144,5915

5970 DATA 233,32,1,45,76,104,42,32,111
 ,68,230,151,165,151,201,64,2323
 5980 DATA 144,245,169,0,133,151,230,15
 0,165,150,201,64,144,233,176,225,8840
 5990 DATA 165,151,166,150,133,186,134,
 187,32,151,69,144,3,76,203,68,1291
 6000 DATA 165,130,133,188,165,131,133,
 189,160,3,177,130,48,239,201,2,3624
 6010 DATA 208,235,136,177,130,56,233,6
 ,168,169,155,153,192,36,32,249,4676
 6020 DATA 59,136,48,8,177,130,153,192,
 36,76,160,68,32,231,54,165,1559
 6030 DATA 188,133,130,165,189,133,131,
 160,2,177,130,56,233,2,24,101,397
 6040 DATA 130,133,252,165,131,105,0,13
 3,253,32,171,221,165,186,133,151,6491
 6050 DATA 165,187,133,150,96,32,146,42
 ,169,71,162,106,160,0,140,72,235
 6060 DATA 71,32,48,69,32,119,69,32,104
 ,69,32,142,69,32,104,69,5919
 6070 DATA 165,160,197,159,144,6,165,16
 4,197,163,176,3,76,124,64,165,2704
 6080 DATA 159,133,161,166,161,165,163,
 32,151,69,176,5,165,163,32,165,2436
 6090 DATA 44,230,161,165,161,197,160,2
 40,234,144,232,230,163,165,163,197,164
 8
 6100 DATA 164,240,220,144,218,32,146,4
 2,32,210,44,32,1,45,76,104,7323
 6110 DATA 42,133,133,134,132,132,177,1
 85,216,69,24,105,2,133,128,185,2450
 6120 DATA 220,69,105,0,133,129,160,0,1
 77,132,201,155,240,5,145,128,3749
 6130 DATA 200,208,245,96,32,163,42,201
 ,155,208,2,24,96,201,27,208,2485
 6140 DATA 2,56,96,32,225,38,76,83,69,1
 74,72,71,165,155,149,159,2318
 6150 DATA 165,158,149,163,238,72,71,96
 ,169,71,162,198,160,1,132,177,3912
 6160 DATA 32,48,69,32,83,69,176,1,96,1
 04,104,32,146,42,96,169,8754
 6170 DATA 71,162,218,160,2,76,125,69,1
 34,150,32,25,45,160,0,177,8767
 6180 DATA 130,201,255,240,6,197,150,24
 0,10,144,2,56,96,32,242,59,1113
 6190 DATA 76,156,69,24,96,216,69,70,71
 ,32,72,112,152,35,35,35,6346
 6200 DATA 35,128,162,141,163,225,236,2
 27,128,136,227,137,128,145,153,152,824
 8
 6210 DATA 151,128,128,128,161,161,144,
 145,128,128,128,180,185,176,165,154,72
 60
 6220 DATA 128,128,128,128,128,128,128,
 128,128,155,128,162,141,163,225,236,81
 48
 6230 DATA 227,128,226,249,128,162,225,
 242,242,249,128,171,239,236,226,229,51
 03
 6240 DATA 128,134,128,162,242,249,225,
 238,128,179,227,232,225,240,240,229,56
 23
 6250 DATA 236,128,155,23,31,62,76,99,1
 12,44,52,146,161,43,43,43,6950
 6260 DATA 63,63,63,63,63,63,63,63,63,6
 3,63,63,63,125,126,254,9697
 6270 DATA 255,28,29,30,31,157,156,127,
 158,159,68,73,82,71,79,84,9218
 6280 DATA 83,65,86,76,79,65,83,69,84,8
 7,73,68,76,73,83,0,5473
 6290 DATA 155,254,65,98,66,173,162,48,
 49,49,49,47,50,65,28,29,4904
 6300 DATA 30,31,8,20,11,4,3,13,5,6,7,1
 5,18,16,1,172,287
 6310 DATA 11,100,232,184,68,74,212,197
 ,150,68,54,43,92,116,64,62,9709
 6320 DATA 39,40,39,38,46,50,50,68,65,6
 7,51,52,52,52,52,64,3780
 6330 DATA 50,255,254,253,159,158,157,1
 56,155,127,126,125,31,30,29,28,11
 6340 DATA 27,68,49,58,42,46,42,155,68,
 49,58,32,32,32,32,2882
 6350 DATA 32,32,32,32,32,32,155,37,
 114,114,111,114,69,114,114,8527
 6360 DATA 111,114,37,114,114,111,114,2
 6,0,0,0,48,114,101,115,5083
 6370 DATA 115,0,97,0,107,101,121,38,10

```

5,108,101,26,0,0,0,0,2516
6380 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,38
,114,101,238
6390 DATA 101,26,50,101,99,97,108,99,2
6,47,38,38,0,0,0,0,1300
6400 DATA 0,0,0,0,0,0,0,0,0,98,121,2
6,50,47,55,1553
6410 DATA 0,0,0,47,38,38,47,46,0,50,47
,55,0,0,0,35,9950
6420 DATA 47,44,53,45,46,0,0,0,74,71,1
84,72,180,229,248,244,4558
6430 DATA 128,128,128,128,174,245,237,
226,229,242,128,128,166,245,238,227,37
32
6440 DATA 244,233,239,238,128,128,128,
128,128,128,128,128,36,37,44,37,9761
6450 DATA 52,37,26,0,48,111,115,105,11
6,105,111,110,0,35,117,114,7909
6460 DATA 115,111,114,12,0,50,37,52,53
,50,46,155,35,47,48,57,4250
6470 DATA 26,0,48,111,115,105,116,105,
111,110,0,35,117,114,115,111,9078
6480 DATA 114,12,0,50,37,52,53,50,46,1
55,45,47,54,37,26,0,2719
6490 DATA 48,111,115,105,116,105,111,1
10,0,35,117,114,115,111,114,12,8348
6500 DATA 0,50,37,52,53,50,46,155,53,1
12,112,101,114,0,108,101,7805
6510 DATA 102,116,0,111,102,0,34,108,1
11,99,107,155,44,111,119,101,9453
6520 DATA 114,0,114,105,103,104,116,0,
111,102,0,34,108,111,99,107,7929
6530 DATA 155,53,112,112,101,114,0,108
,101,102,116,0,111,102,0,36,6280

```

```

6540 DATA 101,115,116,14,155,48,50,41,
46,52,26,0,48,111,115,105,5819
6550 DATA 116,105,111,110,0,35,117,114
,115,111,114,12,0,50,37,52,5220
6560 DATA 53,50,46,155,44,41,51,52,26,
0,48,111,115,105,116,105,7189
6570 DATA 111,110,0,35,117,114,115,111
,114,12,0,50,37,52,53,50,4553
6580 DATA 46,155,38,47,50,45,33,52,0,0
,0,42,117,115,116,105,5460
6590 DATA 102,121,0,44,36,111,108,108,
97,114,0,4,8,36,9,0,2380
6600 DATA 46,35,111,109,109,97,115,0,0
,8,12,9,0,46,36,101,2537
6610 DATA 99,14,0,112,108,97,99,101,11
5,0,16,46,57,44,50,35,4238
6620 DATA 16,17,18,19,20,21,22,23,24,2
5,0,40,18,38,105,108,2379
6630 DATA 101,0,101,120,105,115,116,11
5,12,0,50,101,112,108,97,99,8338
6640 DATA 101,31,0,57,15,46,155,50,31,
30,26,24,29,27,51,53,2388
6650 DATA 48,0,0,0,0,0,155,28,29,30,31
,187,72,187,72,0,5787
6660 DATA 53,73,86,73,11,2,20,65,66,67
,0,26,52,48,49,50,2595
6670 DATA 51,52,53,54,55,56,57,0,10,20
,30,40,50,60,70,80,3130
6680 DATA 90,94,42,47,43,45,179,74,200
,74,64,1,0,0,0,0,2858
6690 DATA 64,2,0,0,0,0,63,80,0,0,0,0,3
3,33,16,17,9242
6700 DATA 226,2,227,2,144,37,0,0,0,0,0
,0,0,0,0,0,8561

```

LISTING 2: ASSEMBLY

```

0100 ;SAVEHD:BCALC.PT1
0110 .OPT NO LIST
0120 *=-
0130 PASS := PASS+1
0140 .IF PASS=1
0150 .INCLUDE HD:BCALC.PT2
0160 .ENDIF
0170 ;@TOS
0180 ;save N & res N in HSAU
0190 .MACRO CLOSE
0200 LDH # <X1*16
0210 JSR CLOSE
0220 .EMDH
0230 ;
0240 .OPT NO LIST
0250 *=- $2000
0260 SCMEM .DS 40*20
0270 THTMIN .DS 160
0280 BF1 .DS $80
0290 BF2 .DS $80
0300 STRING .DS $80
0310 HDLST .BYTE $70,$70,$50,$42
0320 .WORD SCMEM
0330 .BYTE $90,2,2,2,2,2,2
0340 .BYTE 2,2,2,2,2,2,2,2
0350 .BYTE 2,2,2,2,2,2,$90,$42
0360 MFLP .WORD THTMIN
0370 .BYTE 2,2,2,2,$41
0380 .WORD HDLST
0390 ;@I
0400 DLI PHA
0410 THA
0420 PHA
0430 LDH DLIH
0440 LDA DLIC,X
0450 STA MSYNC
0460 STA COLPF2
0470 INC DLIH
0480 PLA
0490 TAX
0500 PLA
0510 RTI
0520 ;
0530 DLIH .BYTE 0
0540 DLIC .BYTE $82,$80
0550 ;
0560 ;@VBI
0570 VBI LDA #0
0580 STA DLIH
0590 LDH #4
0600 VBI LDA COLOR0,X
0610 STA COLPF0,X
0620 DEH
0630 BPL VBI
0640 JMP SYSUBV
0650 ;@CREDIT
0660 BEGIN JSR IMIT
0670 JSR CLRSCM
0680 JSR DRAMH
0690 JSR SCRH
0700 JSR CLRTXT
0710 LDA # >HDLST
0720 STA HDLSTL+1
0730 LDA # <HDLST
0740 STA HDLSTL
0750 LDA # <SCMEM
0760 STA L
0770 LDA # >SCMEM
0780 STA L+1
0790 LDA # >CREDITF
0800 LDH # <CREDITF
0810 LDY #0
0820 JSR DPRINT
0830 JSR GNKEY
0840 LDA # >CREDIT
0850 LDH # <CREDIT
0860 LDY #0
0870 JSR DPRINT
0880 JSR IVCRS
0890 BFLP JSR CLRTXT
0900 JSR SHMREF
0910 HIMP JSR CLRSTR
0920 LDA CONSLR
0930 .JSEL
0940 BNE TOPT
0950 JMP COMMODE
0960 TOPT CMP #3
0970 BNE TKEY
0980 JSR SHDMST
0990 JMP BFLP
1000 TKEY LDA CH
1010 CMP #5FF
1020 BEQ HIMP
1030 JSR GKEY
1040 LDH #16
1050 STA DM0
1060 AND #57F
1070 TSKY CMP #SPCKEY,X
1080 BEQ SPCAL
1090 DEH
1100 BPL TSKY
1110 LDA DM0
1120 JSR INPUT
1130 BCS BFLP
1140 JMP GTIN
1150 SPCAL LDA SPECL,X
1160 STA SPJMP+1
1170 LDA SPECH,X
1180 STA SPJMP+2
1190 LDA #5FF
1200 STA CH
1210 SPJMP JSR #FFFF
1220 JSR SHMT2
1230 JMP BFLP
1240 GTIN LDA STRING
1250 CMP #EOL
1260 BNE GTN1
1270 JSR SHOCEL
1280 JMP HIMP
1290 GTN1 LDA STR
1300 BEQ HIMP
1310 LDA FRCTXT
1320 BNE TRYTX
1330 LDA STRING
1340 CMP #'='
1350 BEQ AFNM
1360 CMP #'E'
1370 BEQ AFNM
1380 JMP TRYNU
1390 AFNM LDA STR
1400 STA STR2
1410 JSR LOZUP
1420 JSR TRYFNC
1430 BCC HOOK
1440 LDA #582
1450 BNE FHOK
1460 HOOK LDA #2
1470 FHOK STA DTYP
1480 LDA STR2
1490 STA STR
1500 CLC
1510 ADC # <STRING
1520 STA FLPTR
1530 LDA # >STRING
1540 ADC #0
1550 STA FLPTR+1
1560 LDH #0
1570 JSR FST0P
1580 LDA STR
1590 CLC
1600 ADC #5
1610 STA STR
1620 TAX
1630 LDA #EOL
1640 STA STRING,X
1650 JSR INSERT
1660 JSR CLRSTR
1670 LDH #BF2+1
1680 JSR STFLPTR
1690 JSR FLDBP
1700 JSR HM2STR
1710 JMP GTR
1720 TRYNU CMP #'-' ;-.,0-9
1730 BCC TRYTX
1740 BNE #1
1750 LDA STRING+1 ;nx chr #?
1760 CMP #'!'
1770 BCS TRYTX
1780 CMP #'/'
1790 BEQ TRYTX
1800 CMP #'.'
1810 BCC TRYTX
1820 #1 CMP #'/'
1830 BEQ TRYTX
1840 CMP #'!'
1850 BCS TRYTX
1860 JSR SBNUM
1870 JMP GTR
1880 TRYTX LDA #0
1890 STA DTYP
1900 STA FRCTXT
1910 JSR INSERT
1920 GTR LDA CR5Y
1930 STA UCNT
1940 LDA SROM
1950 JSR REFROM
1960 JSR IVCRS
1970 LDA DTYP
1980 JSR SHMTYP
1990 LDA ONF
2000 BEQ NORC
2010 JSR RECALC
2020 NORC LDA CDIR
2030 BEQ MOCR5
2040 JSR MOCR5
2050 MOCR5 JMP BFLP
2060 ;move cps rout
2070 MOCR5 CMP #5IF
2080 BEQ CR
2090 JMP CLA
2100 CR LDY SCOL
2110 CPY #63
2120 BCS MORT
2130 LDA CRSEN
2140 CLC
2150 ADC COLM+1,Y
2160 CMP #41
2170 BCS SCRR2
2180 PHA
2190 LDA CRSEN
2200 STA CR5H
2210 PLA
2220 STA CRSEN
2230 INC SCOL
2240 LDA CURRH
2250 STA UCNT
2260 LDA SROM
2270 JSR REFROM
2280 ENCL JSR IVCRS
2290 LDY SCOL
2300 LDA FPCOL,Y
2310 STA FORMAT
2320 MORT RTS
2330 SCRR2 INC SCOL
2340 LDA #2 ;find 1st col
2350 LDY SCOL
2360 FFLP CLC
2370 ADC COLM,Y
2380 CMP #40
2390 BCS FNF
2400 DEY
2410 JMP FFLP
2420 FNF INY
2430 STY FCOL

```

```

2440 JSR DRAWH
2450 JSR REFSCR
2460 ;find st/end of cursor
2470 LDA #2
2480 LDY FCOL
2490 CPY SCOL
2500 BEQ FSF
2510 FSE CLC
2520 ADC COLM,Y
2530 INY
2540 CPY SCOL
2550 BNE FSE
2560 FSP STA CR5K
2570 CLC
2580 ADC COLM,Y
2590 STA CR5EM
2600 LDA SCOL
2610 CMP #63
2620 BNE FFJ
2630 LDA CR5K
2640 CLC
2650 ADC COLM+63
2660 STA CR5EM
2670 FFJ JMP ENC1
2680 ;left
2690 CLA CMP #51E
2700 BNE CIA
2710 CL LDA CR5K
2720 CMP #2
2730 BNE PA
2740 LDA FCOL
2750 BNE SCRLL
2760 RTS
2770 PA DEC SCOL
2780 LDA CURRM
2790 STA VCNT
2800 LDA SROW
2810 JSR REFROM
2820 LDA CR5K
2830 PHA
2840 SEC
2850 LDY SCOL
2860 SBC COLM,Y
2870 STA CR5K
2880 PLA
2890 STA CR5EM
2900 JMP ENC1
2910 SCRLL DEC SCOL
2920 LDA SCOL
2930 STA FCOL
2940 JSR DRAWH
2950 JSR REFSCR
2960 LDA #2
2970 STA CR5K
2980 LDY FCOL
2990 CLC
3000 ADC COLM,Y
3010 STA CR5EM
3020 JMP ENC1
3030 ;up
3040 CIA CMP #51C
3050 BNE CBJ
3060 CU LDA CURRM
3070 BNE R3
3080 LDA FROM
3090 BNE R4
3100 RTS
3110 R3 JSR IVCRS
3120 DEC CURRM
3130 DEC CR5Y
3140 DEC SROW
3150 JMP IVCRS
3160 R4 JSR IVCRS
3170 DEC FROM
3180 DEC SROW
3190 ;
3200 LDA # <[SCMEM+720]
3210 STA JL
3220 LDA # >[SCMEM+720]
3230 STA JL+1
3240 LDA # <[SCMEM+760]
3250 STA L
3260 LDA # >[SCMEM+760]
3270 STA L+1
3280 LDH #16
3290 MUAG JSR MOV5CM
3300 SEC
3310 SBC #40
3320 STA JL
3330 LDA JL+1
3340 SBC #0
3350 STA JL+1
3360 DEH
3370 BPL MUAG
3380 ;
3390 LDH #39
3400 LDA #0
3410 UU STA SCMEM+80,X
3420 DEH
3430 BPL UU
3440 STA VCNT
3450 LDA #2
3460 STA SPK
3470 LDA FROM
3480 JSR REFROM
3490 JSR DRCOL
3500 JMP IVCRS
3510 CDJ CMP #51D ;down
3520 BEQ CD
3530 RTS
3540 CD LDA FROM
3550 CLC
3560 ADC CURRM
3570 CMP #63
3580 BCC R1
3590 RTS
3600 R1 LDA CURRM
3610 CMP #17
3620 BCS R2
3630 JSR IVCRS
3640 INC CURRM
3650 INC CR5Y
3660 INC SROW
3670 JMP IVCRS
3680 R2 JSR IVCRS
3690 INC FROM
3700 INC SROW
3710 ;
3720 LDA # <[SCMEM+120]
3730 STA JL
3740 LDA # >[SCMEM+120]
3750 STA JL+1
3760 LDA # <[SCMEM+80]
3770 STA L
3780 LDA # >[SCMEM+80]
3790 STA L+1
3800 LDH #16
3810 MDAG JSR MOV5CM

```

```

3820 CLC
3830 ADC #40
3840 STA JL
3850 LDA JL+1
3860 ADC #0
3870 STA JL+1
3880 DEH
3890 BPL MDAG
3900 LDA #0
3910 LDH #39
3920 DDZ STA SCMEM+760,X
3930 DEH
3940 BPL DDZ
3950 LDA #2
3960 STA SPK
3970 LDA #17
3980 STA VCNT
3990 CLC
4000 ADC FROM
4010 JSR REFROM
4020 JSR DRCOL
4030 JMP IVCRS
4040 ;mov scrm
4050 MOV5CM LDY #39
4060 M52 LDA (JL),Y
4070 STA (L),Y
4080 DEY
4090 BPL M52
4100 LDA JL+1
4110 STA L+1
4120 LDA JL
4130 STA L
4140 RTS
4150 ;close
4160 CLOSE STX #5AU
4170 LDA #12
4180 STA ICCOM,X
4190 JSR CIOU
4200 LDH #5AU
4210 RTS
4220 ;draw screen
4230 SCRVR JSR DRCOL
4240 JMP CLEAR
4250 SCRVR JSR DRAWH
4260 CLEAR JSR CLR5CM
4270 RTS
4280 ;draw screen, key, v, status
4290 DPRINT STA IL+1
4300 STX IL
4310 DOP LDA (IL),Y
4320 CMP #EOL
4330 BEQ ENDP
4340 STA (L),Y
4350 INY
4360 BNE DOP
4370 ENDP RTS
4380 ;draw color
4390 DRCOL LDY #0
4400 STY VCNT
4410 LDA FROM
4420 STA BCD1
4430 INC BCD1
4440 FG LDA BCD1
4450 LDH #0
4460 FE CMP #10
4470 BCC FF
4480 SEC
4490 SBC #10
4500 INX
4510 BNE FE
4520 FF PHA
4530 THA
4540 PHA
4550 LDY VCNT
4560 LDA Y40L,Y
4570 STA L
4580 LDA Y40H,Y
4590 STA L+1
4600 PLA
4610 ORA #590
4620 LDY #0
4630 STA (L),Y
4640 PLA
4650 INY
4660 ORA #590
4670 STA (L),Y
4680 INC BCD1
4690 INC VCNT
4700 LDA VCNT
4710 CMP #10
4720 BNE FG
4730 RTS
4740 ;initia
4750 INIT CLD
4760 LDA # <[SCMEM+80]
4770 STA Y40L
4780 STA L
4790 LDA # >[SCMEM+80]
4800 STA Y40H
4810 STA L+1
4820 LDY #1
4830 I1 LDA L
4840 CLC
4850 ADC #40
4860 STA L
4870 STA Y40L,Y
4880 LDA L+1
4890 ADC #0
4900 STA L+1
4910 STA Y40H,Y
4920 INY
4930 CPY #10
4940 BNE I1
4950 ;
4960 LDA # <DLI
4970 STA UP5L5T
4980 LDA # <DLI
4990 STA UP5L5T+1
5000 LDA #2
5010 STA COLOR4
5020 LDA #0
5030 STA COLOR2
5040 LDA #10
5050 STA COLOR1
5060 LDA #6
5070 LDH # >VBI
5080 LDY # <VBI
5090 JSR SETUBV
5100 LDA #5C0
5110 STA MHIEH
5120 ;
5130 LDY #63
5140 IJ LDA #8
5150 STA COLM,Y
5160 LDA #520
5170 STA FNCOL,Y
5180 DEY
5190 BPL IJ

```

```

5200 STA FORMAT
5210 LDA #0
5220 STA SPFLG
5230 LDY #501
5240 ZPLP STA #80,X
5250 DEH
5260 BMI ZPLP
5270 STA CR5Y
5280 STA RCFLG
5290 STA ONF
5300 LDA #2
5310 STA CR5K
5320 CLC
5330 ADC COLM
5340 STA CR5EM
5350 ;key
5360 LDA #79
5370 ORA #7A
5380 BEQ BUZZ
5390 LDA #79
5400 STA GTU+1
5410 LDA #7A
5420 STA GTU+2
5430 ;
5440 BUZZ LDA #57F
5450 TAY
5460 TAX
5470 KLP THA
5480 STA AIT,Y
5490 EOR #580
5500 STA AIT+580,Y
5510 DEH
5520 DEY
5530 BMI KD
5540 CPY #55F
5550 BNE KA
5560 LDH #53F
5570 BNE KLP
5580 KCA CPY #51F
5590 BNE KLP
5600 LDH #55F
5610 BNE KLP
5620 ;dec stack
5630 KD LDA # <R01
5640 STA L
5650 LDA # >R01
5660 STA L+1
5670 LDY #0
5680 CET LDA L
5690 STA RAD,Y
5700 LDA L+1
5710 INY
5720 STA RAD,Y
5730 LDA L
5740 CLC
5750 ADC #1
5760 STA L
5770 LDA L+1
5780 ADC #0
5790 STA L+1
5800 INY
5810 CPY #580
5820 BNE CET
5830 LDY #63
5840 LDA #5FF
5850 CES STA R01,Y
5860 DEY
5870 BPL CES
5880 LDA # <ENPROG
5890 STA ENDAT
5900 LDA # >ENPROG
5910 STA ENDAT+1
5920 ;fp table addresses
5930 LDA # <FPSTACK
5940 STA L
5950 STA STLO
5960 LDA # >FPSTACK
5970 STA L+1
5980 STA STHI
5990 LDY #1
6000 FPZ LDA L
6010 CLC
6020 ADC #6
6030 STA L
6040 STA STLO,Y
6050 LDA L+1
6060 ADC #0
6070 STA L+1
6080 STA STHI,Y
6090 INY
6100 CPY #57
6110 BNE FPZ
6120 RTS
6130 ;draw box
6140 DRAWH LDA # <[SCMEM+40]
6150 STA L
6160 LDA # >[SCMEM+40]
6170 STA L+1
6180 LDY #39
6190 DN LDA #580
6200 STA (L),Y
6210 DEY
6220 BPL DN
6230 ;
6240 LDA #2
6250 STA TM
6260 LDA FCOL
6270 STA TM1
6280 COA LDY TM1
6290 CPY #64
6300 BEQ CDO
6310 LDA COLM,Y
6320 PHA
6330 LSR A
6340 CLC
6350 ADC TM
6360 STA TM3
6370 DEC TM3
6380 PLA
6390 CLC
6400 ADC TM
6410 STA TM2
6420 CMP #41
6430 BCC COK
6440 CDO RTS
6450 COK LDA #33
6460 STA TM4
6470 LDA TM1
6480 CMP #26
6490 BCC T2
6500 INC TM4
6510 SEC
6520 SBC #26
6530 CMP #26
6540 BCC T2
6550 INC TM4
6560 SEC
6570 SBC #26

```



```

6580 T2 PHA
6590 LDY TM3
6600 LDA TM4
6610 EOR #580
6620 STA (L),Y
6630 PLA
6640 CLC
6650 ADC #521
6660 EOR #580
6670 INY
6680 STA (L),Y
6690 INC TM
6700 LDA TM2
6710 STA TM
6720 JMP COA
6730 ;inverse cursor
6740 IUCRS JSR LOCCRS
6750 LDY CRSEM
6760 DEY
6770 IV5 LDA (L),Y
6780 EOR #580
6790 STA (L),Y
6800 DEY
6810 CPY CR5X
6820 BPL TV
6830 LDY SCOL
6840 LDA FMCOL,Y
6850 STA FORMAT
6860 JSR SHMCRI
6870 RTS
6880 ;find cursor
6890 LOCCRS LDY CR5Y
6900 LDA V40L,Y
6910 STA L
6920 LDA V40H,Y
6930 STA L+1
6940 RTS
6950 ;clear
6960 CLR1K1 LDY #0
6970 TYA
6980 CT STA THTHIM,Y
6990 INY
7000 CPY #160
7010 BNE CT
7020 LDA #580
7030 STA THTHIM+2
7040 RTS
7050 ;get reg. key
7060 GKEY JSR GKEY
7070 LDX #5FF
7080 STX CH
7090 RTS
7100 ;get a key
7110 GKEY LDA CH
7120 CMP #5FF
7130 BEQ GKEY
7140 STA OLDC
7150 K0
7160 CPY #5C0
7170 BCC GTV
7180 LDY #59A
7190 GTV LDA $FEFE,Y
7200 STA OLDC
7210 KGB CMP #582
7220 BNE KII
7230 KIM LDA CAP5
7240 EOR #540
7250 STA CAP5
7260 KIJ LDA #5FF
7270 STA CH
7280 BNE GKEY
7290 KII CMP #581
7300 BNE K3
7310 LDA INVFLG
7320 EOR #580
7330 STA INVFLG
7340 JMP KIJ
7350 K3 CMP #583
7360 BEQ KIM
7370 CMP #584
7380 BEQ GKEY
7390 CMP #585
7400 BEQ GKEY
7410 LDA OLDC
7420 CMP #540
7430 BCS K6
7440 LDA OLDC
7450 CMP #97
7460 BCC K6
7470 CMP #123
7480 BCS K6
7490 LDA CAP5
7500 BEQ K6
7510 ORA OLDC
7520 JMP K0
7530 K6 LDA OLDC
7540 LDM #15
7550 KCMP CMP #5KEY,K
7560 BEQ KOUT
7570 DEX
7580 BPL KCMP
7590 EOR INVFLG
7600 KOUT RTS
7610 ;limit print
7620 LIM1 LDY #1
7630 BNE TP5
7640 LIM2 LDY #2
7650 BNE TP5
7660 LIM3 LDY #3
7670 TP5 LDA THIML,Y
7680 STA L
7690 LDA THIMH,Y
7700 STA L+1
7710 LDY #1
7720 RTS
7730 ;out of screen
7740 MERR LDY #2
7750 JSR DSKERR
7760 JMP HOME
7770 ;move up
7780 MOVUP LDA ENDAT
7790 STA MFR+1
7800 CLC
7810 ADC LENG
7820 STA MTO+1
7830 LDA ENDAT+1
7840 STA MFR+2
7850 ADC #0
7860 STA MTO+2
7870 CMP #06
7880 BCS MERR
7890 LDA ENDAT
7900 SEC
7910 SBC LL
7920 STA NBYT
7930 LDA ENDAT+1
7940 SBC LL+1
7950 STA NBYT+1
7960 LDA ENDAT
7970 CLC
7980 ADC LENG
7990 STA ENDAT
8000 LDA ENDAT+1
8010 ADC #0
8020 STA ENDAT+1
8030 MAG JSR MFR
8040 BCC MML
8050 RTS
8060 MML DEC MFR+1
8070 LDA MFR+1
8080 CMP #5FF
8090 BNE M5
8100 DEC MFR+2
8110 M5 DEC MTO+1
8120 LDA MTO+1
8130 CMP #5FF
8140 BNE MT
8150 DEC MTO+2
8160 MT JMP MAG
8170 MFR LDA $FFFF
8180 MTO DEC $FFFF
8190 M4 LDA NBYT
8200 CMP #5FF
8210 BNE CNR
8220 DEC NBYT+1
8230 LDA NBYT+1
8240 CMP #5FF
8250 BNE CNR
8260 SEC
8270 RTS
8280 CNR CLC
8290 RTS
8300 ;move in
8310 MOVIN LDA LL
8320 STA MTO+1
8330 CLC
8340 ADC LENG
8350 STA MFR+1
8360 LDA LL+1
8370 STA MTO+2
8380 ADC #0
8390 STA MFR+2
8400 STA ENDAT
8410 LDA ENDAT
8420 SEC
8430 SBC MFR+1
8440 STA NBYT
8450 LDA ENDAT+1
8460 SBC MFR+2
8470 STA NBYT+1
8480 LDA ENDAT
8490 SEC
8500 SBC LENG
8510 STA ENDAT
8520 LDA ENDAT+1
8530 SBC #0
8540 STA ENDAT+1
8550 DD1 JSR MFR
8560 BCC DD4
8570 RTS
8580 DD4 INC MFR+1
8590 BNE DD2
8600 INC MFR+2
8610 DD2 INC MTO+1
8620 BNE DD3
8630 INC MTO+2
8640 DD3 JMP DD1
8650 ;set key
8660 AS2IC STX #5AV
8670 TAX
8680 LDA AIT,X
8690 LDX #5AV
8700 RTS
8710 ;insert
8720 ;b#0 = col # b#1 = format
8730 ;b#2 = length b#3 = type
8740 ;types: 0=text 1=# 2=function
8750 INSERT LDA SCOL
8760 STA COLNU
8770 LDA FROM
8780 CLC
8790 ADC CURR
8800 STA ROMMU
8810 ASL A
8820 TAY
8830 LDA RAD,Y
8840 STA LL
8850 INY
8860 LDA RAD,Y
8870 STA LL+1
8880 IU LDY #0
8890 LDA (LL),Y
8900 BPL FCCL
8910 IS LDA STR
8920 CLC
8930 ADC #DTLEN
8940 STA LENG
8950 JSR MOVUP
8960 LDY #0
8970 LDA COLNU
8980 STA (LL),Y
8990 LDA FORMAT
9000 LDY #1
9010 STA (LL),Y
9020 INY
9030 LDA STR
9040 STA (LL),Y
9050 INY
9060 LDA #0
9070 ORA DTYP
9080 STA (LL),Y
9090 JSR ADDDT
9100 LDY #0
9110 IT LDA STRING,Y
9120 STA (LL),Y
9130 INY
9140 CPY STR
9150 BNE IT
9160 LDA STR
9170 CLC
9180 ADC #DTLEN
9190 STA LENG
9200 LDA ROMMU
9210 JSR UPAD
9220 RTS
9230 ;w/ RAD tab up
9240 UPAD ASL A
9250 TAY
9260 INY
9270 INY
9280 JMP IQ
9290 IR LDA RAD,Y
9300 CLC
9310 ADC LENG
9320 STA RAD,Y
9330 INY
9340 LDA RAD,Y
9350 ADC #0
9360 STA RAD,Y
9370 CPY LL
9380 BNE IQ
9390 BNE IR
9400 RTS
9410 FCCL CMP COLNU
9420 BNE M0D
9430 LDA ROMMU
9440 JSR DELTCL
9450 SEC
9460 M0D BCS IS
9470 LDY #2
9480 LDA (LL),Y
9490 CLC
9500 ADC LL
9510 STA LL
9520 LDA LL+1
9530 ADC #0
9540 STA LL+1
9550 JSR ADDDT
9560 JMP IU
9570 ;delete col, w/row
9580 DELTCL PHA
9590 LDY #2
9600 LDA (LL),Y
9610 CLC
9620 ADC #DTLEN
9630 STA LENG
9640 JSR MOVDM
9650 PLA
9660 ASL A
9670 TAY
9680 INY
9690 INY
9700 DR LDA RAD,Y
9710 SEC
9720 SBC LENG
9730 STA RAD,Y
9740 INY
9750 LDA RAD,Y
9760 SBC #0
9770 STA RAD,Y
9780 INY
9790 DR CPY #580
9800 BNE DR
9810 RTS
9820 ;clear screen
9830 CLRSCN LDX #17
9840 CTD JSR CLRLLIM
9850 DEX
9860 BPL CTD
9870 RTS
9880 ;get w/w/row
9890 CLRLLIM LDA V40L,X
9900 STA L
9910 LDA V40H,X
9920 STA L+1
9930 LDA #0
9940 LDY #2
9950 C5LP STA (L),Y
9960 INY
9970 CPY #40
9980 BNE C5LP
9990 RTS
10000 ;find last col
10010 L5COL LDY FCOL
10020 OB CLC
10030 ADC COLM,Y
10040 CMP #40
10050 BCS OA
10060 INY
10070 BNE OB
10080 OA STY TM2
10090 RTS
10100 ;refresh screen
10110 REF5CR LDA #0
10120 STA UCNT
10130 LDA FROM
10140 STA TH1
10150 ;rows
10160 ;
10170 OH LDA TM1
10180 JSR REFROM
10190 INC TH1
10200 INC UCNT
10210 LDA UCNT
10220 CMP #18
10230 BNE OH
10240 RTS
10250 ;get RAD
10260 STRAD ASL A
10270 TAY
10280 LDA RAD,Y
10290 STA LL
10300 INY
10310 LDA RAD,Y
10320 STA LL+1
10330 RTS
10340 ;set row enter w/w/row
10350 REFROM PHA
10360 LDA FCOL
10370 STA TM
10380 LDA #2
10390 STA SPX
10400 JSR L5COL
10410 LDX UCNT
10420 JSR CLRLLIM
10430 PLA
10440 JSR STRAD
10450 VE LDY #0
10460 LDA (LL),Y
10470 CMP #5FF
10480 BEQ VA
10490 CMP TM
10500 BEQ VB
10510 BCC VC
10520 VB LDY TM
10530 LDA COLM,Y
10540 CLC
10550 ADC SPX
10560 STA SPX
10570 INC TM
10580 LDA TM
10590 CMP TM2
10600 BCC VE
10610 VA RTS
10620 VC JSR UPICOL
10630 JMP VE
10640 VB LDA LL
10650 CLC
10660 ADC #DTLEN
10670 STA IL
10680 LDA LL+1
10690 ADC #0
10700 STA IL+1
10710 LDY UCNT

```

```

010720 LDA Y40L,Y
010730 STA JL
010740 LDA Y40H,Y
010750 STA JL+1
010760 LDA SPX
010770 STA TH3
010780 LDY #1
010790 LDA (LL),Y
010800 STA FORMAT
010810 LDY #3
010820 LDA (LL),Y
010830 BPL MNER
010840 JMP MNER
010850 MNER AND #3
010860 BEQ NMH
010870 CMP #1
010880 BEQ ISN ;a #
010890 LDY #2 ;a F
010900 LDA (LL),Y
010910 SEC
010920 SBC #6
010930 CLC
010940 ADC IL
010950 TAX ;low
010960 LDA IL+1
010970 ADC #0
010980 TAY
010990 JMP ISM2
011000 ;put on screen
011010 NMH JSR CLRSTR
011020 LDY #2
011030 LDA (LL),Y
011040 TAY
011050 LDA MEOL
011060 STA STRING,Y
011070 DEY
011080 YTA LDA (IL),Y
011090 STA STRING,Y
011100 DEY
011110 BPL YTA
011120 LDA # <STRING
011130 STA IL
011140 LDA # >STRING
011150 STA IL+1
011160 JSR FRMTXT ;result in IL
011170 ;put on screen
011180 NMH1 LDY TH
011190 LDA COLM,Y
011200 STA GMX
011210 LDY #0
011220 STY TH4
011230 YLP LDY TH4
011240 LDA LBUF,Y
011250 JSR AS2IC
011260 LDY TH3
011270 STA (JL),Y
011280 INC TH4
011290 INC TH3
011300 LDA TH4
011310 CMP GMX
011320 BNE VLP
011330 JSR CLRSTR
011340 JMP UD
011350 ;begin if it has #
011360 ISN LDY IL ;str't of #
011370 LDY IL+1
011380 ISM2 JSR FLDBR
011390 JSR SETIMB
011400 JSR FASC
011410 JSR FRMHUM
011420 JMP NMH1
011430 ;numbers
011440 SBNUM LDA #1
011450 STA DTYP
011460 JSR SETFP
011470 JSR ZFR0
011480 JSR AFP
011490 LDY # <STRING
011500 LDY # >STRING
011510 JSR FST0R
011520 LDA #6
011530 STA STR
011540 JSR INSERT
011550 RTS
011560 ;control in cell
011570 MNER LDY #0
011580 LDA #5
011590 STA GMX
011600 LDY TH
011610 LDA COLM,Y
011620 CMP #4
011630 BNE ADD
011640 DEC GMX
011650 ADD LDY TH3
011660 ADP LDA ERMES,X
011670 STA (JL),Y
011680 INY
011690 INX
011700 CPX GMX
011710 BNE ADP
011720 JMP UD ;inx col
011730 ;begin if it has #
011740 SETFP LDA # >STRING
011750 STA INBUF+1
011760 LDA # <STRING
011770 STA INBUF
011780 LDA #0
011790 STA CIX
011800 RTS
011810 ;begin inbuf
011820 SETIMB LDA # >LBUF
011830 STA INBUF+1
011840 LDA # (LBUF
011850 STA INBUF
011860 LDA #0
011870 STA CIX
011880 RTS
011890 ;to string
011900 NM2STR JSR SETIMB
011910 JSR FASC
011920 LDY #0
011930 ML3 LDA (INBUF),Y
011940 BHI ML2
011950 STA STRING,Y
011960 INY
011970 BNE NLP
011980 ML2 AND #57F
011990 STA STRING,Y
012000 INY
012010 STY STR
012020 CPY #1
012030 BEQ NRT
012040 LDA STRING
012050 CMP #0
012060 BNE NRT
012070 LDY #0
012080 ML3 LDA STRING+1,Y
012090 STA STRING,Y

```

```

012100 INY
012110 CPY STR
012120 BNE ML3
012130 DEC STR
012140 NRT RTS
012150 ;low spec ctrl cmds
012160 ;clear string for FP
012170 CLRSTR LDY #57F
012180 LDA #520
012190 CCTL STA STRING,X
012200 DEM
012210 BPL CCTL
012220 RTS
012230 ;convert low to upp case
012240 LOZUP LDY #0
012250 LOLP LDA STRING,Y
012260 CMP MEOL
012270 BEQ LODDM
012280 AND #57F
012290 CMP #2+1
012300 BCS NLO
012310 CMP #a
012320 BCC NLO
012330 EOR #520
012340 NLO STA STRING,Y
012350 INY
012360 BNE LOLP
012370 LODDM RTS
012380 .INCLUDE #D:BCALC.PT3
012390 .INCLUDE #D:BCALC.PT4
012400 .INCLUDE #D:BCALC.PT5
012410 .INCLUDE #D:BCALC.PT6
012420 ENPRG = #
012430 * = RUNAD
012440 .WORD BEGIN
012450 .END

```

LISTING 3: ASSEMBLY

```

0100 ;SAVEHD:BCREQU
0110 ;
0120 CIOV = $E456
0130 ICCOM = $0342
0140 ICBAL = $0344
0150 ICBAH = $0345
0160 ICBLI = $0348
0170 ICBLH = $0349
0180 AUX1 = $034A
0190 AUX2 = $034B
0200 RUNAD = $02E0
0210 INITAD = $02E2
0220
0230 SDSLST = $0230
0240 SDMCTL = $022F
0250 VDSLST = $0200
0260 MHLEN = $040E
0270 M5VNC = $040A
0280 SETUPV = $E45C
0290 SYSUBV = $E45F
0300 COLOR0 = $02C4
0310 COLOR1 = $02C5
0320 COLOR2 = $02C6
0330 COLOR4 = $02C8
0340 COLPF0 = $0B16
0350 COLPF2 = $0B18
0360 KEYDEF = $79
0370 INVFLG = $02B6
0380 ;fp equates
0390 CIX = $F2
0400 INBUF = $F3
0410 FR0 = $04
0420 FR1 = $E0
0430 FLPTR = $FC
0440 LBUF = $0500
0450 AFP = $0B00
0460 FASC = $0B0E
0470 IFP = $0B0A
0480 ZFR0 = $0B44
0490 ZF1 = $0B46
0500 FSUB = $0B60
0510 FADD = $0B66
0520 FMUL = $0B6B
0530 FDSV = $0B20
0540 FST0R = $0B47
0550 FST0P = $0B4B
0560 FLDBR = $0B89
0570 FLD1R = $0B90
0580 FLD0P = $0B8D
0590 FLD1P = $0B9C
0600 FMV0V = $0B06
0610 EXP10 = $0BDC
0620 LOG10 = $0BDD
0630 ;
0640 COMSOL = $0B1F
0650 CH = $02FC
0660 EOL = $0B
0670 DOSVEC = $0A
0680 RAMTOP = $6A
0690 SEOL = $0B
0700 CAP5 = $02BE
0710 DTLEN = 4 ;data length
0720 NUMB = $0500
0730 ;
0740 ;page zero
0750 * = $00
0760 L .D5 2
0770 LL .D5 2
0780 IL .D5 2
0790 JL .D5 2
0800 FPLD .D5 2
0810 TH .D5 1
0820 TH1 .D5 1
0830 TH2 .D5 1
0840 TH3 .D5 1
0850 TH4 .D5 1
0860 FCOL .D5 1
0870 FORMAT .D5 1
0880 VCNT .D5 1
0890 CR5X .D5 1
0900 CR5Y .D5 1
0910 WPOS .D5 1
0920 GMX .D5 1
0930 COLNU .D5 1
0940 ROMNU .D5 1
0950 NBYT .D5 2
0960 LENG .D5 1
0970 SCOL .D5 1
0980 FROM .D5 1
0990 CURRH .D5 1
1000 SROW .D5 1
1010 COLF .D5 1
1020 COL5 .D5 1
1030 COLT .D5 1

```

```

1040 COLZ .D5 1
1050 ROMF .D5 1
1060 ROM5 .D5 1
1070 ROMT .D5 1
1080 ROMZ .D5 1
1090 DM0 .D5 1
1100 DM1 .D5 1
1110 DM2 .D5 1
1120 DM3 .D5 1
1130 DM4 .D5 1
1140 DM5 .D5 1
1150 DM6 .D5 1
1160 DM7 .D5 1
1170 HSAV .D5 1
1180 ASAV .D5 1
1190 YSAV .D5 1
1200 NARG .D5 1
1210 SUY .D5 1
1220 FPPTR .D5 1
1230 FTOK .D5 1
1240 STOK .D5 1
1250 OPERAT .D5 1
1260 OLDG .D5 1
1270 OLDAC .D5 1
1280 TA1 .D5 1
1290 TA2 .D5 1
1300 TA3 .D5 1
1310 TA4 .D5 1
1320 BCD1 .D5 1
1330 EDIT .D5 1
1340 SPH .D5 1
1350 STR .D5 1
1360 DTYP .D5 1
1370 FPN1 .D5 3
1380 ARG1 .D5 3
1390 ROF5T .D5 1
1400 COF5T .D5 1
1410 HLDY .D5 1
1420 HLD5 .D5 1
1430 HLDY .D5 1
1440 STR2 .D5 1
1450 LEFTP .D5 1
1460 RIGHTP .D5 1
1470 .OPT LIST
1480 ENZ = #
1490 .OPT NO LIST

```

LISTING 4: ASSEMBLY

```

0100 ;SAVEHD:BCALC.PT3
0110 ;HOME:BCALC.PT3
0120 HOME JSR CLRSCM
0130 JSR R5TCR5
0140 HM2 JSR DRAMH
0150 JSR DRCOL
0160 HM3 JSR REF5CR
0170 JSR SHMT2
0180 JMP TUCR5
0190 ;reset cursor
0200 R5TCR5 LDA #0
0210 STA FCOL
0220 STA FROM
0230 STA SROM
0240 STA SCOL
0250 CR52 STA CURRM
0260 STA CR5Y
0270 LDA #2
0280 STA CR5X
0290 LDY FCOL
0300 CLC
0310 ADC COLM,Y
0320 STA CR5EN
0330 RTS
0340 ;show stats
0350 SHOWST JSR CLRTRT
0360 LDY #26
0370 SJA LDA STY,Y
0380 STA THTMIN+2,Y
0390 DEY
0400 BPL SJA
0410 LDY #32
0420 SJA LDA SEG,Y
0430 STA THTMIN+42,Y
0440 DEY
0450 BPL SJA
0460 ;free mem
0470 LDA #0
0480 SEC
0490 SBC ENDAT
0500 STA FR0
0510 LDA #06
0520 SBC ENDAT+1
0530 STA FR0+1
0540 JSR IFP
0550 JSR FASC
0560 LDA #0
0570 FMLP LDA (INBUF),Y
0580 PHA
0590 AND #57F
0600 JSR AS2IC
0610 STA THTMIN+29,Y
0620 INY
0630 PLA
0640 BPL FMLP
0650 ;file name
0660 LDY #0
0670 FSLP LDA FNAME,Y
0680 CMP MEOL
0690 BEQ FSL1
0700 JSR AS2IC
0710 STA THTMIN+7,Y
0720 INY
0730 CPY #16
0740 BNE FSLP
0750 FSL1 JSR GCH
0760 JMP CLRTRT
0770 ;set cross ref
0780 SETREF JSR ST2BF1
0790 JSR FINARG
0800 BCS S02
0810 JSR GTCLRM
0820 BCS S02
0830 JSR FINARG
0840 LDY #0
0850 S9LP LDA BF1,Y
0860 STA BF1,X
0870 CMP MEOL
0880 BEQ S8P
0890 INY
0900 INX
0910 BNE S9LP
0920 S8P LDY #3
0930 SFR LDA BF1,X
0940 JSR AS2IC

```

```

0950 STA REFTAB,X
0960 DEM
0970 BPL SFR
0980 LDA COLNU
0990 STA SETCOL
1000 LDA ROMMU
1010 STA SETROM
1020 S02 RTS
1030 ;show news
1040 SHMREF JSR CLLIN3
1050 LDA #1
1060 SMO LDA REFTAB+2,X
1070 STA THTMIN+141,X
1080 LDA REFTAB,X
1090 STA THTMIN+121,X
1100 DEM
1110 BPL SMO
1120 LDA FROM
1130 CLC
1140 ADC CURRM
1150 JSR STRAD
1160 MRA LDY #0
1170 LDA (LL),Y
1180 CMP #5FF
1190 BEQ NRF
1200 CMP SETCOL
1210 BEQ GRF
1220 BCC URF
1230 NRF JMP S07 ;try col
1240 URF JSR UPICOL
1250 JMP MRA
1260 GRF LDY #3
1270 LDA (LL),Y
1280 AND #3
1290 BEQ TXT ;text
1300 CMP #1
1310 BNE NRF
1320 JSR LL2FR0
1330 JSR FASC
1340 LDY #0
1350 S06 LDA (IMBUF),Y
1360 PHA
1370 AND #57F
1380 JSR AS2IC
1390 STA THTMIN+124,Y
1400 PLA
1410 BMI S07
1420 INY
1430 CPY #14
1440 BNE S06
1450 TKT LDY #2
1460 LDA (LL),Y
1470 STA ASAV
1480 JSR ADDDT
1490 LDY #0
1500 S08 LDA (LL),Y
1510 JSR AS2IC
1520 STA THTMIN+124,Y
1530 INY
1540 CPY ASAV
1550 BEQ S07
1560 CPY #14
1570 BNE S08
1580 ;now show col
1590 S07 LDA SETROM
1600 JSR STRAD
1610 LDA SCOL
1620 STA TH
1630 MCL LDA #0
1640 LDA (LL),Y
1650 CMP #5FF
1660 BEQ NCOL
1670 CMP TH
1680 BEQ GCOL
1690 BCC UCOL
1700 NCOL RTS
1710 MCOL JSR UPICOL
1720 JMP MCL
1730 GCOL LDY #3
1740 LDA (LL),Y
1750 AND #3
1760 BEQ TTK ;text
1770 CMP #1
1780 BNE NCOL
1790 JSR LL2FR0
1800 JSR FASC
1810 LDY #0
1820 SK2 LDA (IMBUF),Y
1830 PHA
1840 AND #57F
1850 JSR AS2IC
1860 STA THTMIN+144,Y
1870 PLA
1880 BMI SK3
1890 INY
1900 CPY #14
1910 BNE SK2
1920 TTK LDY #2
1930 LDA (LL),Y
1940 STA ASAV
1950 JSR ADDDT
1960 LDY #0
1970 SK4 LDA (LL),Y
1980 JSR AS2IC
1990 STA THTMIN+144,Y
2000 INY
2010 CPY ASAV
2020 BEQ SK3
2030 CPY #14
2040 BNE SK4
2050 SK3 RTS
2060 ;show data type
2070 SHMT2 LDY SCOL
2080 LDA SROM
2090 JSR LOCCEL
2100 BCC SKB
2110 LDA #3
2120 STA DTYP
2130 BNE SHMTYP
2140 SKB LDY #3
2150 LDA (LL),Y
2160 AND #3
2170 STA DTYP
2180 JMP SHMTYP
2190 ;ll # 2 fr0
2200 LL2FR0 JSR ADDDT
2210 LDY LL
2220 LDY LL+1
2230 JMP FLDBR ;auto ret
2240 ;clear line3
2250 CLLIN3 LDY #119
2260 LDA #0
2270 LP3 STA THTMIN,Y
2280 INY
2290 CPY #160
2300 BNE LP3
2310 RTS
2320 ;show ent. type
2330 ;0-t,1-n,2-f
2340 SHMTYP LDA DTYP
2350 ASL A
2360 ASL A
2370 ASL A
2380 TAX
2390 LDY #31
2400 STG LDA TEXTYP,X
2410 STA SCHEM,Y
2420 INX
2430 INY
2440 CPY #39
2450 BNE STG
2460 RTS
2470 ;direct
2480 DIR JSR CLRSCM
2490 LDA #1
2500 STA DIRTXT+1
2510 JSR ST2BF1
2520 JSR FINARG
2530 BCS NDARG
2540 STA DIRTXT+1
2550 NDARG CLOSE 2
2560 LDA #1
2570 STA UCNT
2580 LDA #3
2590 LDY #520
2600 STA ICCOM,X
2610 LDA # <DIRTXT
2620 STA ICBAL,X
2630 LDA # >DIRTXT
2640 STA ICBAN,X
2650 LDA #6
2660 STA AUX1,X
2670 LDA #8
2680 STA AUX2,X
2690 JSR CIOU
2700 BPL DIROK
2710 GENER JSR DSKERR ;error
2720 JMP HM3
2730 DIROK LDY #520
2740 LDA #5
2750 STA ICCOM,X
2760 LDA # <LBUF
2770 STA ICBAL,X
2780 LDA # >LBUF
2790 STA ICBAN,X
2800 LDA #8
2810 STA ICBLL,X
2820 LDA #514
2830 STA ICBLL,X
2840 JSR CIOU
2850 BPL PRDIR
2860 JSR GCH
2870 CLOSE 2
2880 JMP HM3
2890 PRDIR JSR SCMP05
2900 LDY #0
2910 LDY #11
2920 DIRLP LDA LBUF,X
2930 CMP #EOL
2940 BEQ NHDR
2950 JSR AS2IC
2960 STA (L),Y
2970 INY
2980 INX
2990 JMP DIRLP
3000 NHDR INC UCNT
3010 LDA UCNT
3020 CMP #17
3030 BNE DIROK
3040 JSR GCH
3050 JSR CLRSCM
3060 LDA #8
3070 STA UCNT
3080 BEQ DIROK
3090 ;get scrn pos
3100 SCMP05 LDY UCNT
3110 LDA Y40L,Y
3120 STA L
3130 LDA Y40H,Y
3140 STA L+1
3150 RTS
3160 ;save
3170 SAVE JSR ST2BF1
3180 JSR FINARG
3190 JSR GFFILE
3200 JSR CHKDEU
3210 BCC FILE?
3220 DEVERR LDY #165
3230 JMP GENER
3240 FILE? JSR FILEHST
3250 BCC DEVOK
3260 JMP HOME
3270 DEVOK LDA #6 ;write
3280 LDY #11 ;get rec
3290 BNE DOIT
3300 LOAD JSR ST2BF1
3310 JSR FINARG
3320 JSR GFFILE
3330 JSR CHKDEU
3340 BCS DEVERR
3350 LDA #4 ;read
3360 LDY #7 ;input
3370 DOIT STA DM6 ;keep em
3380 STY DM7
3390 CLOSE 2
3400 LDY #520
3410 LDA DM6
3420 STA AUX1,X
3430 LDA #3
3440 STA ICCOM,X
3450 LDA # (FNAME
3460 STA ICBAL,X
3470 LDA # >FNAME
3480 STA ICBAN,X
3490 LDA #0
3500 STA AUX2,X
3510 JSR CIOU
3520 JSR RNOK
3530 JMP GENER ;error
3540 RNOK LDY #520
3550 LDA DM7
3560 TAY
3570 STA ICCOM,X
3580 LDA # <COLM
3590 STA ICBAL,X
3600 LDA # >COLM
3610 STA ICBAN,X
3620 CPY #7 ;inp?
3630 BNE RITE
3640 LDA #5FF
3650 STA ICBLL,X
3660 STA ICBLL,X
3670 BNE SAMP
3680 RITE LDA ENDAT
3690 SEC
3700 SBC # <COLM
3710 STA ICBLL,X
3720 LDA ENDAT+1
3730 SBC # >COLM
3740 STA ICBLL,X
3750 SIMP JSR CIOU
3760 BPL IOOK
3770 CPY #136 ;EOF?
3780 BEQ IOOK
3790 JMP GENER ;error
3800 IOOK CLOSE 2
3810 LDA DM6
3820 CMP #6
3830 BEQ NRF
3840 LDY #520
3850 LDA # <COLM
3860 CLC
3870 ADC ICBLL,X
3880 STA ENDAT
3890 LDA # >COLM
3900 ADC ICBLL,X
3910 STA ENDAT+1
3920 NRF JMP HOME
3930 ;GOO
3940 GOTO JSR ST2BF1
3950 JSR FINARG
3960 BCS OAB
3970 JSR GTCLRM
3980 BCS OAB
3990 LDA ROMMU
4000 CMP #47
4010 BCS OAB
4020 LDA COLNU
4030 CMP #55
4040 BCS OAB
4050 JSR IFR5
4060 JSR CLRSCM
4070 LDA COLNU
4080 STA FCOL
4090 STA SCOL
4100 LDA ROMMU
4110 STA FROM
4120 STA SROM
4130 LDA #0 ;4 crs2
4140 JSR CR52
4150 JMP HM2
4160 OAB RTS
4170 ;EXTRA
4180 ST2BF1 LDY #114
4190 S01 LDA STRING,Y
4200 STA BF1,Y
4210 DEY
4220 BPL S01
4230 RTS
4240 ;REGALAND
4250 ARECL JSR RECALC
4260 JMP HOME
4270 ;force text
4280 FORCE LDA #1
4290 STA FRCTXT
4300 RTS
4310 ;kill cell
4320 KILLCL LDA SCOL
4330 TAX
4340 LDA FROM
4350 CLC
4360 ADC CURRM
4370 STA TH
4380 JSR LOCCEL
4390 BCC LCELL ;none
4400 RTS
4410 LCELL LDA TH
4420 JSR DETCL
4430 JSR IUCRS
4440 LDA #2
4450 STA SPH
4460 LDA CURRM
4470 STA UCNT
4480 LDA SROM
4490 JSR IFR05
4500 JSR IUCRS
4510 RTS
4520 ;find and
4530 FINARG LDY #0
4540 F8LP LDA BF1,Y
4550 CMP #EOL
4560 BEQ NOARG
4570 CMP #520
4580 BEQ FSPCE
4590 INY
4600 CPY #520
4610 BNE F8LP
4620 NOARG SEC
4630 RTS
4640 FSPCE INY
4650 FARG LDA BF1,Y
4660 CMP #EOL
4670 BEQ NOARG
4680 CMP #520 ;spac
4690 BNE GTARG
4700 INY
4710 CPY #520
4720 BNE FARG
4730 BEQ NOARG
4740 GTARG CLC
4750 RTS
4760 ;get file name
4770 GFFILE LDY #0
4780 F9LP LDA BF1,Y
4790 STA FNAME,X
4800 CMP #EOL
4810 BEQ FGD
4820 INY
4830 INX
4840 BNE F9LP
4850 FGD RTS
4860 ;chg col width
4870 MIDTH JSR ST2BF1
4880 JSR FINARG
4890 BCS NOM
4900 JSR GTCLRM
4910 BCS NOM
4920 INC ROMMU
4930 LDA ROMMU
4940 CMP #2
4950 BCC NOM
4960 CMP #33
4970 BCS NOM
4980 LDY COLMU
4990 STA COLM,Y
5000 JSR CLRSCM
5010 JSR DRAMH
5020 JSR REFSCR
5030 LDY FCOL
5040 STY SCOL
5050 LDA #2
5060 STA CR5X
5070 CLC
5080 ADC COLM,Y

```

```

5090 STA CRSEN
5100 JSR IUCRS
5110 NOW RTS
5120 ;show a cell's content
5130 SHOCEL LDA SCOL
5140 LDA SROM
5150 JSR LOCCEL
5160 BCC UAA
5170 RTS ;none
5180 UAA LDY #1 ;format
5190 LDA (LL),Y
5200 STA $00
5210 INY
5220 LDA (LL),Y ;length
5230 STA LENG
5240 INY
5250 LDA (LL),Y
5260 AND #3
5270 BNE UAD
5280 UAG JSR ADDDT
5290 LDY LENG ;text
5300 DEY
5310 UAH LDA (LL),Y
5320 STA STRING,Y
5330 DEY
5340 BPL UAH
5350 UAE LDA $5FF
5360 STA KP05
5370 JSR DRWIN
5380 RTS ;done
5390 UAB CMP #2
5400 BEQ UAF
5410 JSR ADDDT
5420 LDH LL
5430 LDY LL+1
5440 JSR FLDBR
5450 JSR FASC
5460 LDY #0
5470 UAD LDA (INBUF),Y
5480 PHA
5490 AND $57F
5500 STA STRING,Y
5510 PLA
5520 BMI UAE
5530 INY
5540 BNE UAD
5550 UAF LDA LENG
5560 SEC
5570 SBC #6
5580 STA LENG
5590 JMP UAG
5600 ;edit mode
5610 EDITH LDA #1
5620 STA EDIT
5630 JSR SHOCEL
5640 JSR INPUT2
5650 PHR
5660 LDA #0
5670 STA EDIT
5680 PLP
5690 BCC E00K
5700 RTS
5710 E00K PLA
5720 PLA
5730 LDA $00
5740 STA FORMAT
5750 JMP GTIN
5760 ;format a cell or col
5770 FORM JSR CLRTXT
5780 LDY SCOL
5790 LDA FMCOL,Y
5800 STA FORMAT
5810 LDH #18
5820 FPB DEH
5830 BMI FPX
5840 FPA LDA FMES,X
5850 STA TTXMIN+2,X
5860 CPB #5
5870 BCC FPB
5880 LDA FMA-5,X
5890 STA TTXMIN+42,X
5900 LDA FMB-5,X
5910 STA TTXMIN+82,X
5920 LDA FMC-5,X
5930 STA TTXMIN+122,X
5940 JMP FPB
5950 FPX JMP SHOFRM
5960 ;
5970 FPC LDA CH
5980 CMP $5FF
5990 BEQ FPC
6000 LDH $5FF
6010 STX CH
6020 CMP $51C
6030 BNE FPD ;done
6040 RTS
6050 FPD LDH #2
6060 FPE CMP JSKEY,X
6070 BEQ FPF
6080 DEX
6090 BPL FPE
6100 BMI FPG
6110 LDA FORMAT
6120 AND #252
6130 STA ASAV
6140 THA
6150 ORA ASAV
6160 STA FORMAT
6170 JMP SHOFRM
6180 FPG CMP $53A ;D
6190 BNE FPH
6200 LDA FORMAT
6210 EOR #8
6220 STA FORMAT
6230 JMP SHOFRM
6240 FPH CMP #20
6250 BNE FPJ ;,
6260 LDA FORMAT
6270 EOR #4
6280 STA FORMAT
6290 JMP SHOFRM
6300 FPJ LDH #9
6310 FPK CMP NKEY,X ;dec places
6320 BEQ FQD
6330 DEX
6340 BPL FPK
6350 BMI FPC
6360 FQD THA
6370 PHA
6380 LDA FORMAT
6390 AND $50F
6400 STA FORMAT
6410 PLA
6420 ASL A
6430 ASL A
6440 ASL A
6450 ASL A
6460 ORA FORMAT

```

```

6470 STA FORMAT
6480 ;
6490 SHOFRM LDA FORMAT
6500 AND #3
6510 TAX
6520 LDA LCR,X
6530 STA TTXMIN+19
6540 LDA FORMAT
6550 AND #8
6560 LSR A
6570 LSR A
6580 LSR A
6590 TAX
6600 LDA YN,X
6610 STA TTXMIN+59
6620 LDA FORMAT
6630 AND #4
6640 LSR A
6650 LSR A
6660 TAX
6670 LDA YN,X
6680 STA TTXMIN+99
6690 LDA FORMAT
6700 AND $5F0
6710 LSR A
6720 LSR A
6730 LSR A
6740 LSR A
6750 TAX
6760 LDA SDEC,X
6770 STA TTXMIN+139
6780 JMP FPC
6790 ;global format
6800 GLOBAL JSR FORM
6810 LDY SCOL
6820 LDA FORMAT
6830 STA FMCOL,Y
6840 RTS
6850 ;format a cell
6860 FORM LDA SROM
6870 LDH SCOL
6880 JSR LOCCEL
6890 BCC ECL
6900 RTS
6910 ECL JSR FORM
6920 LDY #1
6930 LDA FORMAT
6940 STA (LL),Y
6950 LDH CURRH
6960 STX UCNT
6970 LDA SROM
6980 JSR REFROM
6990 LDY SCOL ;restore
7000 LDA FMCOL,Y ;glob. formt
7010 STA FORMAT
7020 JMP IUCRS
7030 ;recalc flag
7040 RONOF LDA ONF
7050 EOR #3
7060 STA ONF
7070 TAX
7080 LDY #0
7090 RPP LDA ONOFH,X
7100 STA SEG+7,Y
7110 INX
7120 INY
7130 CPY #3
7140 BNE RPP
7150 RTS
7160 ;flip row/col flag
7170 RCFLP LDA RCFLG
7180 EOR #6
7190 STA RCFLG
7200 TAX
7210 LDY #0
7220 RHK LDA RCHEM,X
7230 STA SEG+27,Y
7240 INX
7250 INY
7260 CPY #6
7270 BNE RHK
7280 RTS
7290 ;disk error
7300 DSKERR STY YSAU
7310 CLOSE Z
7320 LDY YSAU
7330 STY FR0
7340 LDA #0
7350 STA FR0+1
7360 JSR IFP
7370 JSR FASC
7380 LDY $5FF
7390 ERLPA INY
7400 LDA (INBUF),Y
7410 PHA
7420 AND $57F
7430 JSR A52IC
7440 STA DERM+6,Y
7450 PLA
7460 BPL ERLPA
7470 JSR CLLIN3
7480 LDY #20
7490 ERLPB LDA DERM,Y
7500 STA TTXMIN+124,Y
7510 DEY
7520 BPL ERLPB
7530 LDA $5FF
7540 STA CH
7550 JSR GAKEY
7560 RTS
7570 ;Number Format Subroutine
7580 ;for B-CALC
7590 ;FORMATTING
7600 ;bit 4-7: Places after Decimal
7610 ;bit 3: Dollar Sign Flag
7620 ;bit 2: Comma Flag
7630 ;bit 0-1: L/C/R Justify
7640 ;
7650 FRMMUM LDY $5FF
7660 IFM0 INY
7670 LDA (INBUF),Y
7680 STA NUMB,Y
7690 BPL IFM0
7700 LDY $57F
7710 LDA #'0
7720 FM.1 STA STRING,Y
7730 DEY
7740 BPL FM.1
7750 LDA #'$
7760 STA STRING
7770 LDA # (STRING+1)
7780 STA IL
7790 LDA # (STRING+1)
7800 STA IL+1
7810 ;
7820 LDA FORMAT ;get FM byte
7830 PHA
7840 AND $5F0

```

```

7850 LSR A
7860 LSR A
7870 LSR A
7880 LSR A
7890 STA TORITE
7900 PLA
7910
7920 AND #4 ;commas?
7930 BNE FNCOM ;yes
7940 LDY $5FF
7950 ICPL INY
7960 LDA (INBUF),Y
7970 STA (IL),Y
7980 BPL ICPL
7990 IFM1 JMP :DSGN
8000 ;insert commas
8010 FNCOM LDY $5FF
8020 FM:1 INY
8030 LDA NUMB,Y
8040 BMI FM:0
8050 CMP #'.'
8060 BNE FM:1
8070 DEY
8080 FM:0 STY SUY
8090 LDA #29
8100 ILP1 LDA #2
8110 STA FCNT
8120 ILP2 LDA NUMB,Y
8130 AND $57F
8140 STA LBUF,X
8150 BEM
8160 DEY
8170 BMI ILV
8180 DEC FCNT
8190 BPL ILP2
8200 LDA #'.'
8210 STA LBUF,X
8220 BEM
8230 BPL ILP1
8240 ILV INX
8250 LDY #1
8260 ILP3 LDA LBUF,X
8270 STA STRING,Y
8280 INY
8290 INX
8300 CPX #30
8310 BNE ILP3
8320 LDH SUY
8330 LDA NUMB,X
8340 BPL INNV
8350 DEY
8360 LDA STRING,Y
8370 ORA $580
8380 STA STRING,Y
8390 BNE :DSGN
8400 ;
8410 INNV INX
8420 LDA NUMB,X
8430 STA STRING,Y
8440 BMI :DSGN
8450 INY
8460 BNE :MNV
8470 ;
8480 :DSGN LDA FORMAT
8490 AND #8
8500 BEQ :AFTER
8510 LDA # (STRING
8520 STA IL
8530 LDA # (STRING
8540 STA IL+1
8550 ;handle places after decimal
8560 ;search for #
8570 :AFTER LDA #0
8580 STA SCI ;flag
8590 LDY $5FF
8600 HXA INY
8610 LDA STRING,Y
8620 BMI :AF2 ;none
8630 CMP #'E
8640 BNE HXA ;not yet
8650 ;got exp
8660 INC SCI ;set flg
8670 DEY ;invs prev byt
8680 LDA STRING,Y
8690 ORA $580
8700 STA STRING,Y
8710 LDH $5FF ;bck to E
8720 HXB INX
8730 INY
8740 LDA STRING,Y
8750 PHA
8760 AND $57F
8770 STA SVEK,X
8780 LDA #'0
8790 STA STRING,Y
8800 PLA
8810 BPL HXB
8820 ;show end
8830 INX
8840 STX SCI
8850 LDA #EOL
8860 STA SVEK,X
8870 :AF2 LDY $5FF
8880 :AF INY
8890 LDA STRING,Y
8900 BPL :TP
8910 AND $57F
8920 STA STRING,Y
8930 INY
8940 LDA #'.'
8950 STA STRING,Y
8960 STY SUY
8970 JMP :JM
8980 :TP CMP #'.'
8990 BNE :AF
9000 STY SUY
9010 :A2 INY
9020 LDA STRING,Y
9030 BPL :A2
9040 AND $57F
9050 STA STRING,Y
9060 :JM LDA SUY
9070 CLC
9080 ADC TORITE
9090 TAY
9100 LDA TORITE
9110 BNE :EL
9120 DEY
9130 :EL LDA #EOL
9140 STA STRING+1,Y
9150 ;chk for E
9160 LDA SCI
9170 BEQ CKMIN
9180 LDH #0
9190 HXC LDA SVEK,X
9200 STA STRING+1,Y
9210 CMP #EOL
9220 BEQ CKMIN
9230 INX

```

```

9240 INY
9250 BNE HMC
9260 ;change string
9270 CKMIN LDA STRING+1
9280 CMP #'-
9290 BNE FRMTXT
9300 LDA STRING+2
9310 CMP #' ,
9320 BNE FRMTXT
9330 LDY #0
9340 MLPL LDA STRING+3,Y
9350 STA STRING+2,Y
9360 CMP #EOL
9370 BEQ FRMTXT
9380 INY
9390 BNE MLPL
9400 ;re-align of number
9410 FRMTXT LDY #5FF
9420 :L1 INY
9430 LDA (IL),Y
9440 CMP #EOL
9450 BNE :L1
9460 STY SUY
9470 ;left/center/right justify
9480 :LCR LDX #57F
9490 LDA #520
9500 :L2 STA LBUF,X
9510 DEX
9520 BPL :L2
9530 ;
9540 LDA FORMAT
9550 AND #3
9560 TAX
9570 BEQ :HAV
9580 ;right justify
9590 DEX
9600 BNE :CEN
9610 LDY TH
9620 LDA COLM,Y
9630 SEC
9640 SBC SUY
9650 BPL :HV
9660 YUA LDX #0
9670 BEQ :LC
9680 ;center
9690 :CEN LDY TH
9700 LDA COLM,Y
9710 SEC
9720 SBC SUY
9730 BPL :C1
9740 EOR #5FF
9750 LSR A
9760 TAY
9770 JMP YUA
9780 :C1 LSR A
9790 :HV TAX
9800 ;copy the number
9810 :HAV LDY #5FF
9820 :LC INY
9830 LDA (IL),Y
9840 CMP #EOL
9850 BEQ :LB
9860 STA LBUF,X
9870 INX
9880 BNE :LC
9890 :LB LDA # <LBUF
9900 STA IL
9910 LDA # >LBUF
9920 STA IL+1
9930 RTS
9940 SCI .BYTE 0
9950 SVEK .DS 4
9960 .BYTE EOL
9970 ;check for DH
9980 CHKDEV LDA FNAME
9990 CMP #'D
010000 BNE WRDEV
010010 LDA FNAME+1
010020 DUNUH CMP #'1
010030 BEQ AYOK
010040 CMP #'S
010050 BEQ MYOK
010060 CMP #'I
010070 BCC WRDEV
010080 CMP #'4
010090 BCS WRDEV
010100 BCC MYOK
010110 AYOK CLC
010120 RTS
010130 WRDEV SEC
010140 RTS
010150 MYOK LDA FNAME+2
010160 CMP #'2
010170 BEQ AYOK
010180 BNE WRDEV
010190 ;check if file exist
010200 FILEXST LDA #0
010210 STA THTM+2
010220 CLOSE 2
010230 LDA #56 ;unloc
010240 STA ICCOM,X
010250 LDA # >FNAME
010260 STA ICBAH,X
010270 LDA # <FNAME
010280 STA ICBAL,X
010290 JSR CIOU
010300 LDA ICCOH+1,X ;stat?
010310 PHA
010320 CLOSE 2
010330 PLA
010340 CMP #1
010350 BEQ FYES
010360 OVMR CLC
010370 RTS
010380 FYES LDA # >REPME5
010390 LDX # <REPME5
010400 LDY #1
010410 JSR PRNTHM
010420 YNK LDA CH
010430 LDX #57F
010440 CMP #520 ;yes
010450 BEQ OVMR
010460 CMP #523
010470 BNE YNK
010480 STX CH
010490 SEC ;no
010500 RTS
0140 STA FPPTR
0150 JSR ZFR0
0160 PARS LDX #0
0170 STX STR
0180 PLP LDA STRING,X
0190 STA BF1,X
0200 CMP #EOL
0210 BEQ PDON
0220 INX
0230 BNE PLP
0240 PDON STX STR
0250 ;pass remove 0
0260 ;replace with #
0270 :EAB5(-----,---)
0280 ; TH +TM1 TM2
0290 ;fpnl-3 fp tokens
0300 FNDFUN LDX #1
0310 FFK LDA BF1,X
0320 CMP #0
0330 BEQ FFM
0340 INX
0350 CPX STR
0360 BNE FFK
0370 JMP PAB52 ;none
0380 STX TH
0390 LDA # >FTAB
0400 STA LL+1
0410 LDA # <FTAB
0420 STA LL
0430 INX
0440 STX TH
0450 FFS LDY #0
0460 FFR LDA (LL),Y
0470 BEQ FFF ;no mat
0480 BMJ FFP ;1st chr
0490 CMP BF1,X
0500 BNE FFF ;next
0510 INX
0520 INY
0530 BNE FFR
0540 FFO LDA #4
0550 JSR INCLL
0560 LDX TH1
0570 BNE FFS
0580 FFP AND #57F
0590 CMP BF1,X
0600 BNE FFO
0610 LDY #3
0620 LDA (LL),Y ;FX
0630 STA ASAV
0640 ;
0650 STX TH1 ;?
0660 ASL A
0670 TAX
0680 LDA JTAB,X
0690 STA JPOFF+1
0700 LDA JTAB+1,X
0710 STA JPOFF+2
0720 ;now get arguments
0730 ;if 1 arg. in fr0
0740 LDA #0
0750 STA NARG
0760 LDY TH1
0770 LDA BF1,Y
0780 CMP #'(
0790 BEQ FFD
0800 FFX SEC ;error
0810 RTS
0820 FFD INY
0830 LDA BF1,Y ;ck 4 cell
0840 CMP #'D
0850 BCS FFX ;error
0860 CMP #'A
0870 BCS FFX ;cell
0880 STX CXH ;#
0890 STY TH1
0900 JSR FP2BF1
0910 JSR AFP
0920 BCS FFX
0930 JSR SPFR25T
0940 BCS FFX
0950 LDY CXH
0960 FFF LDA BF1,Y ; , or )
0970 CMP #' )
0980 BNE FFC
0990 JMP FFI
1000 FFC CMP #' ,
1010 BEQ FFD
1020 BNE FFX
1030 FFW STY TH1
1040 JSR GTCLRW
1050 FFY BCS FFX ;error
1060 LDY NARG
1070 LDA COLMU
1080 STA COLM,Y
1090 LDA ROMNU
1100 STA ROMF,Y
1110 LDA ASAV
1120 CMP #7 ;cnting only
1130 BEQ FVA
1140 LDX COLNU
1150 LDX ROMNU
1160 JSR CL2FR0
1170 BCS FFY
1180 FVA LDY NARG
1190 LDA #580
1200 STA ARGY,Y
1210 JSR SPFR25T
1220 BCS FFY
1230 FVB LDA TH1
1240 CLC
1250 ADC #4
1260 STA TH1
1270 TAY
1280 JMP FFH
1290 ;collapse & do fp
1300 FFI INY
1310 STY TM2
1320 LDY TH
1330 LDX TH2
1340 JSR COLAPS
1350 LDX #0
1360 FFO LDA BF1,X
1370 CMP #EOL
1380 BEQ FFL
1390 INX
1400 BNE FFO
1410 FFL STX STR
1420 JSR ADD25P
1430 LDY TH
1440 LDA #'#
1450 STA BF1,Y
1460 LDA FPM1
1470 STA BF1+1,Y
1480 JPOFF JSR $FFFF
1490 ;back to begin
1500 BCC JJJ
1510 RTS
1520 JJJ JMP FNDFUN
1530 SPFR25T LDA FPPTR
1540 LDY NARG
1550 STA FPM1,Y
1560 INX NARG
1570 INY
1580 COPY #3
1590 BCS FFG
1600 TAX
1610 JSR FROX
1620 INX FPPTR
1630 CLC
1640 RTS
1650 FFG SEC
1660 RTS
1670 ;get a cell's #
1680 ;K=col A=row FRO=#
1690 CL2FR0 STX COLMU
1700 JSR STRAD
1710 EE LDY #0
1720 LDA (LL),Y
1730 CMP #5FF
1740 BEQ ED
1750 CMP COLMU
1760 BEQ EC
1770 BCS EC
1780 ED SEC ;err
1790 RTS
1800 EC JSR UPICOL
1810 JMP EE
1820 RTS
1830 EB LDY #3 ;c is set
1840 LDA (LL),Y ;data typ
1850 AND #3
1860 BEQ EN ;txt!
1870 CMP #1
1880 BEQ EF
1890 LDY #2 ;fn
1900 LDA (LL),Y
1910 SEC
1920 SBC #6
1930 JSR INCLL
1940 LDA #4
1950 JSR INCLL
1960 LDX #0
1970 LDY LL+1
1980 JSR FLD0R ;2 fro
1990 EM CLC
2000 RTS
2010 ;trunc tab pad 3 spaces + tok
2020 FTAB .BYTE "SQU",0
2030 .BYTE "ABS",1
2040 .BYTE "SQR",2
2050 .BYTE "RND",3
2060 .BYTE "EXP",4
2070 .BYTE "LOG",5
2080 .BYTE "TAN",6
2090 .BYTE "COS",7
2100 .BYTE "SIN",8
2110 .BYTE "AVE",9
2120 .BYTE 0
2130 ;map tab
2140 JTAB .WORD SQU
2150 .WORD ABS
2160 .WORD SQR
2170 .WORD RND
2180 .WORD EXP
2190 .WORD LOG
2200 .WORD INT
2210 .WORD CNT
2220 .WORD SUM
2230 .WORD AVE
2240 ;absolute val
2250 ABS JSR CHKONE
2260 AND LDA FRO
2270 AND #57F
2280 STA FRO
2290 EXK LDX FPM1
2300 JSR FROX
2310 CLC
2320 RTS
2330 ;int
2340 INT LDA #0 ;default
2350 STA TRCF
2360 JSR CHKONE
2370 MF JSR SETIMB
2380 JSR FASC
2390 JSR TRUNC
2400 LDA FRO
2410 PHA
2420 JSR AFP
2430 PLA
2440 BPL EX2 ;trunc flag
2450 LDA TRCF
2460 BEQ EX2
2470 LDX # <ONE
2480 LDY # >ONE
2490 JSR FLD1R
2500 JSR F5B
2510 EX2 JMP EXK
2520 ;SQU
2530 JSR CHKONE
2540 LDX # <5FFP
2550 LDY # >5FFP
2560 JSR F50R
2570 JSR LOG10
2580 LDX # <TWO
2590 LDY # >TWO
2600 SQI2 JSR FLD1R
2610 JSR FDIY
2620 BCC SQ0 ;xit
2630 RTS
2640 SQ0 JSR EXP10
2650 EX3 BCS BDEX
2660 JSR RND5QR
2670 EX0 JMP EX2
2680 BDEX RTS
2690 ;square
2700 SQU JSR CHKONE
2710 JSR LOG10
2720 LDX # <TWO
2730 LDY # >TWO
2740 JSR FLD1R
2750 JSR FMUL
2760 BCC SQ7
2770 RTS
2780 SQ7 JSR EXP10
2790 BCC EX0
2800 RTS
2810 ;RND
2820 RND JSR CHKONE
2830 LDX # <HALF
2840 LDY # >HALF
2850 JSR FLD1R
2860 JSR FADD
2870 BCC RP
2880 RTS
2890 RP JSR FASC
2900 JSR TRUNC
2910 JSR SETIMB

```

LISTING 5: ASSEMBLY

```

3740 INC DM7
3750 CMA INC DM2 ;nxt col
3760 LDA DM2
3770 CMP DM1
3780 BEQ CNB
3790 BCC CNB
3800 INC DM4 ;nxt row
3810 LDA DM7
3820 CMP DMS
3830 BEQ CNC
3840 BCC CNC
3850 JSR ZFR0
3860 LDA DM6 ;gt cnt
3870 STA FR0
3880 LDA DM7
3890 STA FR0+1
3900 JSR IFP
3910 CLC
3920 RTS
3930 ;behc 4 1 arg
3940 CHKONE LDA MARG
3950 BEQ CZ
3960 CMP #1
3970 BEQ CY
3980 CZ PLA ;pull ret
3990 PLA
4000 SEC
4010 CY RTS
4020 ;bhc 2 arg
4030 CHKTM LDA MARG
4040 CMP #2
4050 BNE CZ ;err
4060 RTS
4070 ;getn's cell's
4080 CHKCLD LDY #1
4090 CMP LDA ARCT,Y
4100 BPL CZ ;err
4110 DEY
4120 BPL CMP
4130 RTS
4140 ;bhc 3 arg
4150 CHKSLD LDA ROWS
4160 CMP ROWF
4170 BCC CZ ;err
4180 LDA COLS
4190 CMP COLF
4200 BCC CZ ;err
4210 RTS
4220 ;dup colf-rowz
4230 DUPDAT LDY #7
4240 DDT LDA COLF,X
4250 STA DM0,X
4260 DEY
4270 BPL DDT
4280 RTS
4290 ;truncate
4300 TRUNC LDY #0
4310 TX LDA (INBUF),Y
4320 BHI TEX
4330 CMP #1
4340 BEQ TJAA
4350 INY
4360 BNE TX
4370 TJAA STA TRCF ;flag
4380 TJ LDA NEOL
4390 STA (INBUF),Y
4400 RTS ;NEI
4410 TEN AND #57F
4420 STA (INBUF),Y
4430 INY
4440 BNE TJ
4450 ;save fp
4460 SAUFPM LDY FPPTR
4470 LDA STHI,X
4480 TAY
4490 LDA STLO,X
4500 TAX
4510 JSR FSTOR
4520 RTS
4530 ;return ptr to fnl
4540 RETFPM LDY FPPTR
4550 LDA STHI,X
4560 TAY
4570 LDA STLO,X
4580 TAX
4590 JSR FLDIR
4600 RTS
4610 ;pass 2 find cells
4620 ;replace with ::
4630 PASS2 LDY #1
4640 LCEL LDA BF1,Y
4650 CMP #'D
4660 BCS MXY
4670 CMP #'A
4680 BCS AHA
4690 MXY INY
4700 CPY STR
4710 BNE LCEL
4720 JMP PASS3 ;do #'s next
4730 ANA STY TH ;for collps
4740 TYA ;strt
4750 CLC
4760 ADC #4
4770 STA TH2 ;end
4780 JSR GTCLRM
4790 BCS ACD
4800 LDA ROMMU
4810 JSR STRAD
4820 ACC LDY #0
4830 LDA (LL),Y
4840 CMP #FFF ;end?
4850 BEQ ACD ;error
4860 CMP COLMU
4870 BEQ ACD ;yup
4880 BCC ACB
4890 ACD RTS
4900 ACB JSR UPICOL
4910 JMP ACC
4920 AGA LDY #1 ;gt cell
4930 LDA (LL),Y
4940 AND #3 ;dt type
4950 BEQ ACD ;text!
4960 CMP #1 ;#?
4970 BEQ ACE
4980 LDY #2
4990 LDA (LL),Y
5000 SEC
5010 SBC #6
5020 JSR INCLL
5030 ;def fault
5040 ACE LDA #4
5050 JSR INCLL
5060 LDY LL
5070 LDY LL+1
5080 JSR FLDOR
5090 LDA #4
5100 STA TH1
5110 JSR PUTTOK
5120 LDY TH
5130 JMP MXY
5140 ;put tok in2 bfi
5150 PUTTOK LDY TH
5160 LDY TH2
5170 JSR COLAP5
5180 LDA STR
5190 SEC
5200 SBC TH1
5210 STA STR
5220 JSR ADD25P
5230 LDY TH
5240 LDA #'#
5250 STA BF1,Y
5260 LDA FPPTR
5270 INY
5280 STA BF1,Y
5290 STY TH
5300 ;move fp to stck
5310 TAY
5320 LDA STHI,X
5330 TAY
5340 LDA STLO,X
5350 TAY
5360 JSR FSTOR
5370 INC FPPTR
5380 RTS
5390 ;pass 3 find #'s
5400 ;replace with tokens
5410 PASS3 LDY #1 ;find # loop
5420 FHL LDA BF1,Y
5430 CMP NEOL
5440 BEQ P55
5450 CMP #'(
5460 BEQ P5M
5470 CMP #' )
5480 BEQ P5M
5490 CMP #'$
5500 BNE P5I
5510 INY
5520 INY
5530 JMP P55
5540 P5M INY FNLP ;look again
5550 JSR FNLP ;get #
5560 PSI STY CIX ;list byte #
5570 STY TH
5580 JSR FP2BF1 ;get fp#
5590 JSR AFP
5600 BCS P5L
5610 LDY CIX
5620 TAY
5630 STA TH2
5640 SEC
5650 SBC TH
5660 STA TH1 ;# less
5670 JSR PUTTOK ;get pos. back
5680 LDY TH
5690 MKEN INY
5700 CPY STR
5710 BEQ PASS4
5720 P55 LDA BF1,Y
5730 CMP NEOL
5740 BEQ PASS4
5750 PSK LDY #4
5760 PSJ CMP OPTAB,X
5770 DEN MKA
5780 DEN
5790 BPL PSJ
5800 CMP #' )
5810 BEQ MKEN
5820 P5L SEC
5830 RTS
5840 MKA INY
5850 JMP FNLP
5860 ;pass 4 find nested ()
5870 ;calculate & quit !
5880 PASS4 LDY #0
5890 P4A LDA BF1,Y
5900 CMP NEOL
5910 BEQ NOLF
5920 CMP #'(
5930 BEQ FLP
5940 INY
5950 BNE P4A
5960 NOLF LDY #0
5970 P4B LDA BF1,Y
5980 CMP NEOL
5990 BEQ NOPRM
6000 CMP #' )
6010 BEQ P4ER
6020 INY
6030 BNE P4B ;error
6040 BEQ P4ER
6050 NOPRM LDY #0
6060 P5M LDA BF1+1,X
6070 STA BF2,H
6080 CMP NEOL
6090 BEQ P5V
6100 INX
6110 INY
6120 JMP P5M
6130 P5V JSR CALCIT
6140 BCS P4ER
6150 RTS
6160 FLP INY
6170 LDA BF1,Y
6180 CMP #' )
6190 BEQ FRP
6200 CMP NEOL
6210 BEQ P4ER
6220 BNE FLP
6230 P4ER SEC
6240 RTS
6250 FRP STY RIGHTP
6260 P4C DEY
6270 LDA BF1,Y
6280 CMP #'(
6290 BNE P4C
6300 STY LEFTP
6310 LDY #FFF
6320 LDY LEFTP
6330 PSR INX
6340 INY
6350 CPY RIGHTP
6360 BEQ P5T
6370 LDA BF1,Y
6380 STA BF2,H
6390 JSR P5R
6400 PST LDA NEOL
6410 STA BF2,Y
6420 JSR CALCIT
6430 BCS P4ER
6440 LDA RIGHTP
6450 SEC
6460 SBC LEFTP
6470 CMP #1
6480 BEQ P4ER
6490 LDY LEFTP
6500 LDA BF2
6510 STA BF1,X
6520 LDA BF2+1
6530 INX
6540 STA BF1,X
6550 LDY RIGHTP
6560 PSU INY
6570 LDA BF1,Y
6580 JMP PAS54
6590 ;calculate bfi
6600 ;exponents first!
6610 CALCIT
6620 CALCEM LDY #0
6630 CCM STY TM4 ;save for clipse
6640 JSR GT2TOK
6650 CMP NEOL
6660 BEQ CALMD ;try #,/
6670 BCS CCA
6680 CCB RTS ;SEC=error
6690 CCA JSR MU2FPM
6700 LDA OPERAT
6710 CMP #'A
6720 BNE CCE
6730 JSR FIGEXP
6740 BCS CCB ;error
6750 JSR CLP5BF2 ;collapse
6760 JSR FR02ST
6770 JMP CALCEM
6780 CCE LDA TM4
6790 CLC
6800 #3
6810 TAY
6820 JMP CCM ;look 4 more
6830 ;calculate * & /
6840 CALMD LDY #0
6850 CDC STY TM4
6860 JSR GT2TOK
6870 CMP NEOL
6880 BEQ CALAS
6890 BCC CDA
6900 CDB RTS ;error
6910 CDA JSR MU2FPM
6920 LDA OPERAT
6930 CMP #'*
6940 BNE CDD
6950 JSR FMUL
6960 BCS CDB ;error
6970 BCC CDE
6980 CDD CMP #' /
6990 BNE CDF
7000 JSR FDIU
7010 BCS CDB ;error
7020 CDE JSR CLP5BF2
7030 JSR FR02ST
7040 LDY TM4
7050 JMP CDC
7060 CDF LDA TM4
7070 CLC
7080 ADC #3
7090 TAY
7100 JMP CDC
7110 ;calculate %
7120 CALAS LDY #0
7130 CEG STY TM4
7140 JSR GT2TOK
7150 CMP NEOL
7160 BEQ CDONE
7170 BCC CEA
7180 CEB RTS ;SEC =error
7190 CEA JSR MU2FPM
7200 LDA OPERAT
7210 CMP #' +
7220 BNE CEC
7230 JSR FADD
7240 BCS CEB
7250 BCC CED
7260 CEC CMP #' -
7270 BNE CEB ;error
7280 JSR FSUB
7290 BCS CEB ;error
7300 CED JSR CLP5BF2
7310 JSR FR02ST
7320 LDY TM4
7330 JMP CEG
7340 CDONE CLC ;ok
7350 RTS
7360 ;get 2 tokens & operation
7370 GT2TOK LDA BF2,Y
7380 CMP #'#
7390 BEQ GBA
7400 CMP NEOL ;done?
7410 BEQ GBB
7420 GBC SEC ;error
7430 RTS
7440 GBA INY
7450 LDA BF2,Y
7460 CMP NEOL
7470 BEQ GBC ;error
7480 STA FTOK
7490 INY
7500 LDA BF2,Y
7510 STA OPERAT
7520 CMP NEOL
7530 BEQ GBB ;done
7540 INY
7550 LDA BF2,Y
7560 CMP NEOL ;error
7570 STA STOK
7580 GBB CLC ;A=ool if done
7590 RTS
7600 ;move tok #1 to fp row
7610 MU2FPM LDY FTOK
7620 JSR STFLPTR
7630 JSR FLDOP
7640 LDY STOK

```

```

7880 JSR STFLPTR
7890 JSR FLDPTR
7900 RTS
7910 ;set pointers
7920 STFLPTR LDA STLO,X
7930 STA FLPTR
7940 LDA STHI,X
7950 STA FLPTR+1
7960 RTS
7970 ;collapse bf2
7980 CLP5BF2 LDA TM4
7990 CLC
8000 ADC #2
8010 TAY
8020 CLC
8030 ADC #3
8040 TAY
8050 CF1 LDA BF2,X
8060 STA BF2,Y
8070 CMP HEOL
8080 BEQ CF2
8090 INY
8100 INX
8110 JMP CF1
8120 CF2 RTS
8130 ;figure exp
8140 FIGEXP LDX FTOK
8150 JSR STFLPTR
8160 JSR FLDPTR
8170 JSR LOG10
8180 BCC FIA
8190 FIB SEC
8200 RTS
8210 FIA LDX STOK
8220 JSR STFLPTR
8230 JSR FLDPTR
8240 JSR FMUL
8250 BCS FIB
8260 JSR EMP10
8270 BCS FIB
8280 CLC
8290 RTS
8300 ;get a cell
8310 ;column,row,now have col,row
8320 GTCLRN JSR GETZCH
8330 BCS GRN
8340 JSR FINCMU
8350 BCC GRM
8360 GRM RTS ;error
8370 GRN LDY HLDY
8380 INY
8390 JSR GETZCH
8400 BCS GRM
8410 JSR FINCMU
8420 LDY HLDY
8430 RTS
8440 ;find the column #
8450 FINCMU LDA HLDY
8460 LDX #2
8470 FCA CMP A$26,X
8480 BEQ FCC
8490 DEX
8500 BPL FCA
8510 BHI FCER
8520 LDA H$26,X
8530 STA TH$
8540 LDA HLD$
8550 CMP #'I
8560 BCS FCER
8570 CMP #'A
8580 BCS FCB
8590 FCER SEC ;error
8600 RTS
8610 FCB SEC
8620 SBC #'A
8630 CLC
8640 ADC TH$ ;col #
8650 STA COLMU ;0-63
8660 CMP #64
8670 BCS FCER
8680 RTS
8690 ;get 2 chr's from bf1
8700 GETZCH LDA BF1,Y
8710 CMP HEOL
8720 BEQ GERR
8730 STA HLDY
8740 INY
8750 LDA BF1,Y
8760 CMP HEOL
8770 BEQ GERR
8780 STA HLD$
8790 STY HLDY
8800 CLC
8810 RTS
8820 GERR SEC
8830 RTS
8840 ;get the row #
8850 FINRMU LDA HLDY
8860 JSR GETDEC
8870 BCS FRER
8880 STA ROMMU
8890 LDA HLD$
8900 JSR GETDEC
8910 BCS FRER
8920 TNA
8930 CLC
8940 ADC ROMMU
8950 TAY
8960 DEX
8970 CPX #64
8980 BCS FRER
8990 STX ROMMU
9000 CLC
9010 RTS
9020 FRER SEC
9030 RTS
9040 ;get decimal # (row)
9050 GETDEC LDX #9
9060 DECB CMP DECB,X
9070 BEQ DECA
9080 DEX
9090 BPL DECB
9100 SEC
9110 RTS
9120 DECA LDA DECB,X
9130 CLC
9140 RTS
9150 ;move 10 to fastack
9160 FR025T LDX FTOK
9170 FR08X JSR STFLPTR
9180 JSR FSTOP
9190 RTS
9200 ;collapse ;x=beg y=des
9210 ;uses tm3
9220 COLAPS LDA BF1,X
9230 STA BF1,Y
9240 CMP HEOL
9250 BEQ CPF
9260 INY

```

```

9270 INX
9280 BNE COLAPS
9290 CPF
9300 ;room for fp token
9310 ADD25P LDX STR
9320 ASLP LDA BF1,X
9330 STA BF1+2,X
9340 DEX
9350 CPX TH
9360 BCS ASLP
9370 BCS ASLP
9380 INC STR
9390 INC STR
9400 RTS
9410 ;point to bf1
9420 FP2BF1 LDA #>BF1
9430 STA INBUF+1
9440 LDA #<BF1
9450 STA INBUF
9460 RTS
9470 TRCF .DS 1
9480 ;round off scale
9490 RND50R JSR FMOVE ;rt to fr1
9500 LDX #<SROOT ;save root
9510 LDY #>SROOT
9520 JSR FSTOR
9530 LDX #<SFPN
9540 LDY #>SFPN
9550 JSR FLDOOR ;in to fr0
9560 JSR FDIU ;in/rt
9570 LDX #<SROOT
9580 LDY #>SROOT
9590 JSR FLDIR ;res. root
9600 JSR FADD ;add rts
9610 LDX #<HALF
9620 LDY #>HALF
9630 JSR FLDIR
9640 JSR FMUL ;x .5
9650 RTS
9660 SFPN .DS 6
9670 SROOT .DS 6

```

LISTING 6: ASSEMBLY

```

0100 ;save B:BCALC.PT5
0110 ;draw text on screen
0120 DRXIN LDA #0
0130 STA TM1
0140 TAY
0150 TAY
0160 DIT STX TM2
0170 LDX TM1
0180 LDA TM1N,X
0190 CLC
0200 ADC #2
0210 STA L
0220 LDA TM1N,X
0230 ADC #0
0240 STA L+1
0250 LDX TM2
0260 DRPL LDA STRING,X
0270 JSR AS2IC
0280 CPX XPOS
0290 BNE DRSC
0300 EOR #500
0310 DRSC STA (L),Y
0320 INX
0330 INY
0340 CPY #38
0350 BNE DRPL
0360 LDY #0
0370 INC TM1
0380 LDA TM1
0390 CMP #3
0400 BNE DIT
0410 RTS
0420 ;get a row #/column
0430 GAKELY LDA CONSOL
0440 CMP #6
0450 BNE GCH
0460 PLA
0470 PLA
0480 LDA HEOL
0490 STA STRING
0500 LDA #0
0510 STA EDIT
0520 STA SPFLG
0530 SEC
0540 RTS
0550 GCH LDA CH
0560 CMP #FF
0570 BEQ GAKELY
0580 JSR GNKEY
0590 LDX XPOS
0600 RTS
0610 ;input rowing
0620 INPUT JSR CLRSTR
0630 INPUT2 LDA #0
0640 STA XPOS
0650 INMAIN JSR DRXIN
0660 JSR GAKELY
0670 LDX XPOS ;edit mode?
0680 LDY EDIT ;yes
0690 BNE ENTS ;yes
0700 LDY SPFLG ;cmd mode?
0710 BNE ENTS ;yes
0720 LDY #4 ;ch 4 eol &
0730 IELP CMP ENKEY,Y ;cursor keys
0740 BEQ INEN
0750 DEX
0760 BPL IELP
0770 BHI SE5C
0780 INEN STA CDIR
0790 CMP HEOL
0800 BNE INDOV
0810 STY CDIEY
0820 JMP INDOV
0830 ENTS CMP HEOL
0840 BNE SE5C
0850 JMP INDOV
0860 SE5C CMP #27 ;ESC?
0870 BNE INEXT
0880 JSR GAKELY
0890 CMP HEOL
0900 BNE INDOV
0910 JMP INDOV
0920 INEXT LDY #12 ;test special
0930 INILP CMP INSPC,Y
0940 BEQ INPRC
0950 DEX
0960 BPL INILP
0970 INOVR STA STRING,X
0980 INX
0990 STX XPOS
1000 CPX #114
1010 BCC INMAIN

```

```

1020 DEX
1030 STX XPOS
1040 BNE INMAIN
1050 ;
1060 INPRC LDA INTABL,Y
1070 STA INJSR+1
1080 LDA INTABH,Y
1090 STA INJSR+2
1100 INJSR JSR STFFF
1110 JMP INMAIN
1120 ;
1130 INDOV LDY #114
1140 INOUT DEY
1150 LDA STRING,Y
1160 CMP #520
1170 BEQ INOUT
1180 LDA HEOL
1190 INY
1200 STA STRING,Y
1210 STY STR
1220 LDA #0
1230 STA EDIT
1240 CLC
1250 RTS
1260 ;kill input
1270 ZAPST JSR CLRSTR
1280 LDA #0
1290 STA XPOS
1300 RTS
1310 ;del char
1320 INDEL LDX XPOS
1330 BEQ INDO
1340 DEX
1350 LDA #520
1360 STA STRING,X
1370 STX XPOS
1380 INDO RTS
1390 ;input left
1400 INLEFT LDX XPOS
1410 BEQ INDO
1420 DEX
1430 STX XPOS
1440 RTS
1450 ;input rt
1460 INRT LDX XPOS
1470 CPX #113
1480 BCS INDO
1490 INX
1500 STX XPOS
1510 RTS
1520 ;control del
1530 INCTDL LDX XPOS
1540 INCTD LDA STRING+1,X
1550 STA STRING,X
1560 INX
1570 CPX #57A
1580 BNE INCTD
1590 RTS
1600 ;control end
1610 INHNS LDA #57A
1620 ININD LDA STRING,X
1630 STA STRING+1,X
1640 DEX
1650 CPX XPOS
1660 BPL ININD
1670 INX
1680 LDA #520
1690 STA STRING,X
1700 STA STRING+114
1710 RTS
1720 ;input up
1730 INUP LDA XPOS
1740 SEC
1750 SBC #38
1760 BPL INST
1770 CLC
1780 ADC #14
1790 INST STA XPOS
1800 RTS
1810 ;input down
1820 INDOWN LDA XPOS
1830 CLC
1840 ADC #38
1850 CMP #114
1860 BCC INST
1870 SEC
1880 SBC #14
1890 JMP INST
1900 ;xpos to start of line
1910 INMOD LDX #0
1920 LDA XPOS
1930 CMP #33
1940 BCC INMD
1950 LDX #38
1960 CMP #76
1970 BCC INMD
1980 LDX #76
1990 INMD STA XPOS
2000 RTS
2010 ;insert line
2020 ININLN JSR INMOD
2030 LDA #37
2040 STA TH$
2050 INLNL JSR ININS
2060 DEC TH$
2070 BPL INLNL
2080 RTS
2090 ;del a line
2100 INDLLN JSR INMOD
2110 LDA #37
2120 STA TH$
2130 INLND JSR INCTDL
2140 DEC TH$
2150 BPL INLND
2160 RTS
2170 ;command mode
2180 COMMODE LDA #4
2190 STA DLIC+1
2200 LDA #1
2210 STA SPFLG
2220 JSR INPUT
2230 BCS CHOUT
2240 JSR LOZUP
2250 CHAOK LDX #0 ;Command #
2260 LDA #<CHTAB
2270 STA L
2280 LDA #>CHTAB
2290 STA L+1
2300 CHRG LDY #0
2310 CH$5 LDA (L),Y
2320 BEQ COMMODE
2330 CMP STRING,Y
2340 BNE CUI3
2350 INY
2360 CPX #3
2370 BNE CM5T
2380 ;
2390 LDA COML,X
2400 STA CMJP+1

```

```

2410 LDA COMH,X
2420 STA CHJP+2
2430 JSR CLRTRT
2440 CMJJP JSR $FFFF
2450 CMOUT LDA #0
2460 STA DLI+1
2470 STA SPFLG
2480 JMP BFLP
2490
2500 CUJ INX
2510 LDA L
2520 CLC
2530 ADC #3
2540 STA L
2550 BCC CMBG
2560 INC L+1
2570 BNE CMBG
2580 ;show col/row
2590 SHMCR1 LDA SCOL
2600 LDH #0
2610 KGZ SEC
2620 SBC #26
2630 BMI KGV
2640 INX
2650 BNE KGZ
2660 KGV PHA
2670 TKA
2680 CLC
2690 ADC #5A1
2700 STA SCHEM+19
2710 PLA
2720 CLC
2730 ADC #5BB
2740 STA SCHEM+20
2750 LDH SROM
2760 INX
2770 TKA
2780 LDH #0
2790 HGV SEC
2800 SBC #10
2810 BMI HGV
2820 INX
2830 BNE HGV
2840 HGV PHA
2850 TKA
2860 CLC
2870 ADC #590
2880 STA SCHEM+21
2890 PLA
2900 CLC
2910 ADC #59A
2920 STA SCHEM+22
2930 RTS
2940 ;print sheet
2950 PRNT JSR CLRTRT
2960 LDA #P
2970 STA FNAME
2980 LDA #1
2990 STA FNAME+1
3000 LDA #EOL
3010 STA FNAME+2
3020 LDA # >PRITH5
3030 LDH # <PRITH5
3040 PRNTIT LDY #0
3050 STY CP05 ;curs. index
3060 JSR PRNTM
3070 JSR PRMS1
3080 JSR STCR5
3090 JSR PRMS2
3100 JSR STCR5
3110 LDA COLF
3120 STA TH ;1st col
3130 LDA COL5 ;chk cell range
3140 SEC
3150 SBC COLF
3160 BMI PERRR
3170 LDA ROM5
3180 SEC
3190 SBC ROMF
3200 BPL PROK
3210 PERRR LDY #141 ;range err
3220 PERRR JSR P5KERR
3230 JMP H#3
3240 ;open #2 on #2
3250 PROK CLOSE 2
3260 LDH #520
3270 LDA #3
3280 STA ICCOM,X
3290 LDA # >FNAME
3300 STA ICBAH,X
3310 LDA # <FNAME
3320 STA ICBAL,X
3330 LDA #0
3340 STA AUX2,X
3350 LDA #0
3360 STA AUX1,X
3370 JSR CIOU
3380 BMI PERRA
3390 PHD LDH TH ;curr col
3400 LDA COLM,X ;get width
3410 STA GHM
3420 LDA ROMF
3430 JSR LOCCEL
3440 BCS CMTB ;empty
3450 LDY #1
3460 LDA (LL),Y
3470 STA FORMAT
3480 LDY #3
3490 LDA (LL),Y
3500 BPL PHF
3510 JMP ISERR ;error
3520 PHF AND #3
3530 BEQ PHA ;text
3540 JMP ITSAN ;M/F
3550 PHA LDY #2
3560 LDA (LL),Y
3570 TAY
3580 JSR ADDDT ;up 4
3590 LDA REOL
3600 STA STRING,Y
3610 PNC DEY
3620 BMI PHB
3630 LDA (LL),Y
3640 STA STRING,Y
3650 JMP PNC
3660 PHD LDA # <STRING
3670 STA IL
3680 LDA # >STRING
3690 STA IL+1
3700 JSR FRMTXT
3710 PHE JSR SHDOUT
3720 INC TH ;do nxt col
3730 LDA TH
3740 CMP COL5
3750 BEQ PHD
3760 BCC PHD
3770 LDA COLF
3780 STA TH
3790 JSR SHDEOL ;snd eol

```

```

3800 INC ROWF ;next row
3810 LDA ROWF
3820 CMP ROM5
3830 BEQ PHD
3840 BCC PHD
3850 CLOSE 2 ;done
3860 RTS
3870 ;send out blanks
3880 CMTB JSR SPLBF
3890 JMP PHE
3900 ;line feed
3910 SHDEOL LDA #11
3920 LDH #520
3930 STA ICCOM,X
3940 LDA #0
3950 STA ICBLI,X
3960 STA ICBLH,X
3970 LDA #59B
3980 JSR CIOU
3990 BMI LSERR
4000 RTS
4010 ;send out cell
4020 SHDOUT LDA #11
4030 LDH #520
4040 STA ICCOM,X
4050 LDA GHM
4060 STA ICBLI,X
4070 LDA #0
4080 STA ICBLH,X
4090 LDA # >LBUF
4100 STA ICBAH,X
4110 LDA # <LBUF
4120 STA ICBAL,X
4130 JSR CIOU
4140 BMI LSERR
4150 RTS
4160 ;1st/print err
4170 LSERR PLA
4180 PLA
4190 JMP GENER
4200 ;space out lbuf
4210 SPLBF LDH #57F
4220 LDH #520
4230 SL9 STA LBUF,X
4240 DEH
4250 BPL SL9
4260 RTS
4270 ;err in cell
4280 ISERR JSR SPLBF
4290 LDY #4
4300 ISA LDA CERR,Y
4310 STA LBUF,Y
4320 DEY
4330 BPL ISA
4340 JMP PHE
4350 ;its a Hor F
4360 ITSAN CMP #1
4370 BNE ITSAF
4380 JSR ADDDT
4390 LDH LL
4400 LDY LL+1 ;??
4410 IT5G JSR FLD0R
4420 JSR FASC
4430 JSR FRMMH
4440 JMP PHE
4450 ;must be a F
4460 ITSAF LDY #2
4470 LDA (LL),Y
4480 TAY
4490 JSR ADDDT
4500 TAY
4510 SEC
4520 SBC #6
4530 CLC
4540 ADC LL
4550 TAY
4560 LDA LL+1
4570 ADC #0
4580 TAY
4590 JMP IT5G
4600 ;list to disk
4610 LIST JSR ST2BF1
4620 JSR FINARG
4630 JSR GTFLE
4640 JSR CHKDEV
4650 BCC LDKK
4660 JMP DEVERR
4670 LDKK JSR FILEXST
4680 BCC FLDK
4690 JMP HOME
4700 FILEK JSR CLRTRT
4710 LDA # >LSMES
4720 LDH # <LSMES
4730 JMP PRNTIT
4740 ;copy & move
4750 ;cmfig 0-copy 1=mov
4760 ;copy 2=block verbatim
4770 COPY JSR CLRTRT
4780 LDA # <CPYMES
4790 LDH # <CPYMES
4800 LDY #0
4810 STY CMFLG
4820 JMCY STY CP05
4830 JSR PRNTM
4840 JSR PRMS1
4850 JSR STCR5
4860 JSR PRMS2
4870 JSR STCR5
4880 JSR PRMS3
4890 JSR STCR5
4900 ;show disk ends
4910 LDA COL5
4920 CMP COLF
4930 BCS CLOK
4940 CSER JMP PERRR
4950 CLOK LDA ROM5
4960 CMP ROMF
4970 BCC CSER
4980 ;chk if dest = corners
4990 LDA COLT
5000 CMP COLF
5010 BNE LOMR
5020 LDA ROMT
5030 CMP ROMF
5040 BEQ CSER
5050 LOMR LDA COLT
5060 CMP COL5
5070 BNE CKOK
5080 LDA ROMT
5090 CMP ROM5
5100 BEQ CSER
5110 ;set cntns
5120 CKOK LDA COL5
5130 SEC
5140 SBC COLF
5150 STA DM4
5160 STA DM2
5170 LDA ROM5
5180 SEC

```

```

5190 SBC ROMF
5200 STA DM5
5210 ;offsets
5220 SEC
5230 LDA COLT
5240 SBC COLF
5250 STA COFST
5260 SEC
5270 LDA ROMT
5280 SBC ROMF
5290 STA ROFST
5300 JSR CNKFSB ;A =typ inc
5310 ASL A
5320 TAY
5330 LDA INCTAB,X
5340 STA INCRES+1
5350 LDA INCTAB+1,X
5360 STA INCRES+2
5370 JSR IUCRS
5380 CPY1 LDA COLF
5390 STA DM7
5400 LDA COLT
5410 STA DM7
5420 CPY5 LDA ROMT ;dest
5430 JSR STRAD ;get add
5440 CPY2 LDY #0 ;see if MT
5450 LDA (LL),Y
5460 CMP #5FF ;MT
5470 BEQ DM7 ;col?
5480 CMP DM7
5490 BEQ FUL
5500 BCS NOEM
5510 JSR UPICOL
5520 LDH CPY2
5530 FUL LDA DM7
5540 JSR DELTCL
5550 NOEN LDA LL
5560 STA IL
5570 LDA LL+1
5580 STA IL+1
5590 LDA ROMF
5600 JSR STRAD
5610 LDY #0 ;chk source
5620 LDA (LL),Y
5630 CMP #5FF
5640 BEQ INCRES ;no cell
5650 CMP DM6 ;col?
5660 BEQ G11
5670 BCS INCRES ;no cell
5680 JSR UPICOL
5690 JMP CPY3
5700 ;move cell to G11
5710 G11 LDA # >BF1
5720 STA JL+1
5730 LDA # <BF1
5740 STA JL
5750 LDY #2
5760 LDA (LL),Y
5770 CLC
5780 ADC #DTLEN
5790 TAY
5800 DEY
5810 STA LENG ;4 move
5820 STY GHM
5830 CPY4 LDA (LL),Y
5840 STA (JL),Y
5850 DEY
5860 CPY #5FF
5870 BNE CPY4
5880 LDA DM7 ;new col #
5890 STA BF1
5900 LDA IL
5910 STA LL
5920 LDA IL+1
5930 STA IL+1
5940 LDA CMFLG
5950 BEQ NOCHG
5960 JSR HWPARS
5970 NOCHG LDY GHM
5980 INY
6000 STY LENG
6010 JSR MOVUP
6020 LDA ROMT
6030 JSR UPAD
6040 LDY GHM
6050 CPY5 LDA (JL),Y
6060 STA (JL),Y
6070 DEY
6080 CPY #5FF
6090 BNE CPY6
6100 INCRES JSR $FFFF
6110 BCS CFIN
6120 JMP CPY5
6130 CFIN JSR RECALC
6140 JSR REFSOR
6150 JSR IUCRS
6160 CLC
6170 RTS
6180 ;increase dm7 routes
6190 RTDM INC DM7
6200 INC DM6 ;next col
6210 DEC DM4
6220 LDA DM4
6230 CMP #5FF
6240 BNE RBX
6250 RBZ INC ROMT ;next row
6260 INC ROWF
6270 RRR DEC DM5
6280 LDA DM5
6290 CMP #5FF
6300 BNE RBY
6310 SEC ;done
6320 RTS
6330 RBY LDA COLF
6340 STA DM6
6350 LDA COLT
6360 STA DM7
6370 LDA DM2
6380 STA DM4
6390 RBX CLC
6400 RTS ;keep going
6410 ;
6420 RTUP INC DM7
6430 INC DM6
6440 DEC DM4
6450 LDA DM4
6460 CMP #5FF
6470 BNE RBX
6480 DEC ROMT
6490 DEC ROMF
6500 JMP RRR
6510 ;
6520 LFDN DEC DM7
6530 DEC DM6
6540 LDA DM4
6550 LDA DM4
6560 CMP #5FF
6570 BNE RBX

```



```

6580 BEQ #BZ
6590 ;check source, end & dest. cell
6600 ;if in sheet
6610 CHKFS5 LDA ROMT
6620 CMP ROMF
6630 BNE TBFH
6640 LDA COLT ;A,E
6650 CMP COLF
6660 BNE VAE
6670 VER PLA ;error
6680 PLA
6690 JMP PERRR
6700 VAE BCC VEE ;E
6710 LDA COLS ;A
6720 SEC
6730 SBC COLF
6740 CLC
6750 ADC COLT
6760 STA COLT
6770 CMP #64 ;0-63
6780 BCS VER
6790 LDA COLS ;switch
6800 PHA
6810 LDA COLF
6820 STA COLS
6830 PLA
6840 STA COLF
6850 LDA #1 ;ifdn
6860 CLC
6870 RTS
6880 VEE LDA #0 ;rtdn
6890 CLC
6900 RTS
6910 TBFH BCC VEE
6920 LDA ROMS
6930 SEC
6940 SBC ROMF
6950 CLC
6960 ADC ROMT
6970 STA ROMT
6980 CMP #64
6990 BCS VER
7000 LDA ROMS ;switch
7010 PHA
7020 LDA ROMF
7030 STA ROMS
7040 PLA
7050 STA ROMF
7060 LDA #2 ;rtup
7070 CLC
7080 RTS
7090 ;
7100 INCTAB .WORD RTDM
7110 .WORD LFDN
7120 .WORD RTUP
7130 ;
7140 ;jump table
PRM53 LDA #2
7150 STA ASAV
7160 LDA # >CMES3
7170 LDX # <CMES3
7180 LDY #3
7190 JMP PRMA
7200 ;move block
7210 MOVEB JSR CLRTRT
7220 LDA #1
7230 STA ASAV
7240 STA CMFLG
7250 LDA # >MOVME5
7260 LDX # <MOVME5
7270 LDY #0
7280 JMP JMPCPY ;jmp in2 cpy
7290 ;reassign cell values
7300 ;if error = changed to eol
7310 ;dmu, dml used
7320 MUPARS LDY #4
7330 LDA BFI,Y
7340 CMP #'=
7350 BEQ JAA
7360 JAB RTS ;quit
7370 JAA INY
7380 CPY GKH ;max len
7390 BCS JAB ;end!
7400 LDA BFI,Y
7410 CMP #'D
7420 BCS JAA ;nxt
7430 CMP #'A
7440 BCC JAA
7450 STY Y9 ;get cell
7460 JSR GTCLRM
7470 BCC JAC
7480 JAD LDY Y9
7490 BNE JAA
7500 JAC LDA COLMU
7510 CLC
7520 ADC C0F5T ;col offset
7530 STA COLMU
7540 CMP #64
7550 BCS JAB ;err
7560 LDA ROMMU
7570 CLC
7580 ADC R0F5T ;row offset
7590 STA ROMMU
7600 CMP #64
7610 BCS JAD ;err
7620 LDY Y9
7630 ;now change
7640 LDX #2
7650 LDA COLMU
7660 JAE CMP #26,X
7670 BCS JAF
7680 DEX
7690 BPL JAE
7700 JAF PHA
7710 LDA AS26,X
7720 STA BFI,Y
7730 PLA
7740 SEC
7750 SBC #26,X
7760 CLC
7770 ADC #55
7780 INY
7790 STA BFI,Y
7800 INC ROMMU
7810 LDA ROMMU
7820 LDX #3
7830 CMP DECN,X
7840 BCS JAC
7850 DEX
7860 BPL JAH
7870 JAC PHA
7880 LDA DECN,X
7890 INY
7900 STA BFI,Y
7910 PLA
7920 SEC
7930 SBC DECN,X
7940 TAX
7950 LDA DECN,X
7960 INY

```

```

7970 STA BFI,Y
7980 JMP JAA
7990 ;recalc
8000 RECALC LDA #0
8010 STA ROMMU
8020 STA COLMU
8030 LDA RCLF,G
8040 BNE BYC
8050 ;by rows
8060 IG JSR RARITH
8070 INC COLMU
8080 LDA COLMU
8090 CMP #64
8100 BCC IG
8110 LDA #0
8120 STA COLMU
8130 INC ROMMU
8140 LDA ROMMU
8150 CMP #64
8160 BCC IG
8170 BYR JSR REFSCR
8180 JMP IVCRS
8190 ;by cols
8200 BYC JSR RARITH
8210 INC ROMMU
8220 LDA ROMMU
8230 CMP #64
8240 BCC BYC
8250 LDA #0
8260 STA ROMMU
8270 INC COLMU
8280 LDA COLMU
8290 CMP #64
8300 BCC BYC
8310 BCS BYR ;done
8320 ;arith done here
8330 RARITH LDA ROMMU
8340 LDX COLMU
8350 STA TA1
8360 STX TA2
8370 JSR LOCCEL
8380 BCC RBD
8390 RBFF JMP RBGH ;none
8400 RBD LDA LL
8410 STA TA3
8420 LDA LL+1
8430 STA TA4
8440 LDY #3
8450 LDA (LL),Y
8460 BMI RBFF ;err
8470 CMP #2 ;func?
8480 BNE RBFF ;no
8490 DEY
8500 LDA (LL),Y
8510 SEC
8520 SBC #6
8530 LDA #EOL ;end of F
8540 LDA #EOL
8550 STA STRING,Y
8560 JSR ADDDT ;up 4
8570 RRR DEY
8580 BMI RRB
8590 LDA (LL),Y
8600 STA STRING,Y
8610 JMP RRR
8620 RRB JSR TRVFC
8630 LDA TA3
8640 STA LL
8650 LDA TA4
8660 STA LL+1
8670 LDY #2
8680 LDA (LL),Y
8690 SEC
8700 SBC #2
8710 CLC
8720 ADC LL
8730 STA FLPTR
8740 LDA LL+1
8750 ADC #0
8760 STA FLPTR+1
8770 JSR FSTOP
8780 RBGH LDA TA1
8790 STA ROMMU
8800 LDA TA2
8810 STA COLMU
8820 RTS
8830 ;delete block
8840 DELETE JSR CLRTRT
8850 LDA # >DELM5
8860 LDX # <DELM5
8870 LDY #0
8880 STY CPOS
8890 JSR PRMTTH
8900 JSR PRM51
8910 JSR STCR5
8920 JSR PRM52
8930 JSR STCR5
8940 LDA COLS
8950 CMP COLF
8960 BCC KER
8970 LDA ROMS
8980 CMP ROMF
8990 BCS DLC
9000 KER JMP PERRR ;ERR 141
9010 ;now delete
9020 DLC LDA COLF
9030 STA COLT
9040 DLB LDX COLT
9050 LDA ROMF
9060 JSR LOCCEL
9070 BCS DLA
9080 LDA ROMF
9090 JSR DELTCL
9100 DLA INC COLT
9110 LDA COLT
9120 CMP COLS
9130 BEQ DLB
9140 BCC DLB
9150 INC ROMF
9160 LDA ROMF
9170 CMP ROMS
9180 BEQ DLC
9190 BCC DLC
9200 JSR CLRTRT
9210 JSR CLRSCN
9220 JSR REFSCR
9230 JMP IVCRS ;done
9240 ;print in txtwind
9250 ;save pos when x=0 y=lin#
PRMTTH STA IL+1
9270 STX IL
9280 STY Y5AV
9290 LDA TMINL,Y
9300 CLC
9310 ADC #2
9320 STA L
9330 LDA TMINH,Y
9340 ADC #0
9350 STA L+1

```

LISTING 7: ASSEMBLY

```

9360 LDY #0
9370 PR1 LDA (IL),Y
9380 CMP NEOL
9390 BEQ PR2
9400 STA (L),Y
9410 INY
9420 BNE PR1
9430 PR2 RTS
9440 ;get RET/ESC
9450 RETESC JSR GNKEY
9460 CMP #59B
9470 BNE III
9480 CLC
9490 RTS
9500 III CMP #51B
9510 BNE IIT
9520 SEC
9530 RTS
9540 IIT JSR MUCRS ;move?
9550 JMP RETESC
9560 ;stor cns pos
9570 STCR5 LDX CPOS
9580 LDA SCOL
9590 STA COLF,X
9600 LDA SROM
9610 STA ROMF,X
9620 INC CPOS
9630 RTS
9640 ;print msg
PRM51 LDA # >CMES1
9650 LDX # <CMES1
9660 LDY #1
9670 PRMA STY Y5AV ;line#
9680 JSR PRMTTH
9690 JSR RETESC
9700 BCS EOUT
9710 RTS
9720 EOUT PLA ;exit
9730 PLA
9740 JSR CLRTRT
9750 RTS
9760 ;open #52
9770 PRM52 LDA # >CMES2
9780 LDX # <CMES2
9790 LDY #2
9800 JMP PRMA ;cheat
9810 ;chk parameters
9820 ;locat cell
9830 ;x=col, y=row, l=loc
LOCCEL STX COLMU
9850 JSR STRAB
9860 LDY #0
9870 LDA (LL),Y
9880 CMP #5FF
9890 BEQ FQE
9900 CMP COLMU
9910 BEQ FQF
9920 BCC FQG
9930 FQE SEC
9940 FQF JSR UPICOL
9950 JMP FU
9960 FQG JSR UPICOL
9970 JMP FU
9980 FQF CLC
9990 RTS

```

```

0100 ;SAVEHD:bcalc.pt6
0110 ;
0120 V40L .D5 18
0130 V40H .D5 18
0140 TMINL .BYTE <TMTMIN
0150 .BYTE <[TMTMIN+$20]
0160 .BYTE <[TMTMIN+$50]
0170 .BYTE <[TMTMIN+$78]
0180 TMINH .BYTE <TMTMIN
0190 .BYTE <[TMTMIN+$20]
0200 .BYTE <[TMTMIN+$50]
0210 .BYTE <[TMTMIN+$78]
CREDIT .SBYTE +$80," B-Calc (c)"
0230 .SBYTE +$80," 1987 AAB1 "
0240 .SBYTE +$80," TYPE: "
0250 .SBYTE +$80," "
0260 .BYTE EOL
0270 CREDIT .SBYTE +$80," B-Calc by "
0280 .SBYTE +$80," Barry Kolbe & "
0290 .SBYTE +$80," Bryan Schappel"
0300 .SBYTE +$80," "
0310 .BYTE EOL
0320 ;input tables
0330 INTABL .BYTE <ZAPST
0340 .BYTE <INDEL
0350 .BYTE <INCTDL
0360 .BYTE <ININS
0370 .BYTE <INUP
0380 .BYTE <INDWN
0390 .BYTE <INLEFT
0400 .BYTE <INRT
0410 .BYTE <INIMLM
0420 .BYTE <INDLLM
0430 .BYTE <INDO
0440 .BYTE <INDO
0450 .BYTE <INDO
0460 INTABH .BYTE <ZAPST
0470 .BYTE <INDEL
0480 .BYTE <INCTDL
0490 .BYTE <ININS
0500 .BYTE <INUP
0510 .BYTE <INDWN
0520 .BYTE <INLEFT
0530 .BYTE <INRT
0540 .BYTE <INIMLM
0550 .BYTE <INDLLM
0560 .BYTE <INDO
0570 .BYTE <INDO
0580 .BYTE <INDO
0590 ;
0600 INSPC .BYTE $7D,$7E,$FE,$FF
0610 .BYTE $1C,$1D,$1E,$1F
0620 .BYTE $9D,$9C,$7F,$9E
0630 .BYTE $9F
0640 ;
0650 ;command mode tables
0660 CHTAB .BYTE #0
0670 .BYTE "GOT" ;GOTO cell
0680 .BYTE "SAV" ;SAVE Sheet
0690 .BYTE "LOA" ;LOAD Sheet
0700 .BYTE "SET" ;CROSS reference
0710 .BYTE "MID" ;set col WIDTH
0720 .BYTE "LIS" ;LIST to disk
0730 .BYTE 0
0740 ;
0750 COML .BYTE <DIR, <GOTO
0760 .BYTE <SAVE, <LOAD
0770 .BYTE <SETREF, <WIDTH

```

BCALC

```

0780 .BYTE <LIST
0790
0800 COMH .BYTE >DIR, >GOTO
0810 .BYTE >SAVE, >LOAD
0820 .BYTE >SETREF, >WIDTH
0830 .BYTE >LIST
0840
0850 $PCKEY .BYTE $1C, $1D, $1E, $1F
0860 .BYTE $0B, $14, $0B
0870 .BYTE $04, $03, $0D
0880 .BYTE $05, $06, $07
0890 .BYTE $0F, $12, $10
0900 .BYTE $01
0910
0920 $SPECL .BYTE <CU, <CD, <CL, <CR
0930 .BYTE <HOME, <FORCE
0940 .BYTE <KILLCL, <DELETE
0950 .BYTE <COPY, <MOVEB
0960 .BYTE <EDITH, <FORMC
0970 .BYTE <GLOBAL, <RONOF
0980 .BYTE <RCFLP, <PRNT
0990 .BYTE <ARECL
1000
1010 $SPECH .BYTE >CU, >CD, >CL, >CR
1020 .BYTE >HOME, >FORCE
1030 .BYTE >KILLCL, >DELETE
1040 .BYTE >COPY, >MOVEB
1050 .BYTE >EDITH, >FORMC
1060 .BYTE >GLOBAL, >RONOF
1070 .BYTE >RCFLP, >PRNT
1080 .BYTE >ARECL
1090 $LKEY .BYTE $FF, $FE, $FD, $9F
1100 .BYTE $9E, $9D, $9C, $9B
1110 .BYTE $7F, $7E, $7D, $1F
1120 .BYTE $1E, $1D, $1C, $1B
1130
1140 ;Special Text
1150
1160 $IRTYT .BYTE "D1:*.M", EOL
1170 $NAME .BYTE "D1: " , EOL
1180 $RMES .SBYTE "Error"
1190 $ERR .SBYTE "Error"
1200 $DERM .SBYTE "Error: Press a ke
y"
1210
1220 $TT .SBYTE "File: "
1230 .SBYTE " Free:"
1240 $EG .SBYTE "Recalc:OFF "
1250 .SBYTE " by:ROM "
1260
1270 $NOFM .SBYTE "OFFON "
1280 $RMES .SBYTE "ROW COLUMN"
1290 $NF .BYTE 0
1300 $RFLG .BYTE 0
1310 $CMFLG .BYTE 0
1320 $KPY .DS 1 ;y sav
1330 $CPOS .DS 1
1340 $Y .DS 1
1350
1360 $TERTYP .SBYTE "Text "
1370 $NUMTYP .SBYTE "Number "
1380 $FUNTYP .SBYTE "Function"
1390 .SBYTE " "
1400
1410 $DELMS .SBYTE "DELETE: Position Cu
r sor, RETURN", SEOL
1420 $CPYMS .SBYTE "COPY: Position Cur
sor, RETURN", SEOL
1430 $MOVMS .SBYTE "MOVE: Position Cur
sor, RETURN", SEOL
1440 $CMES1 .SBYTE "Upper left of Block
", SEOL
1450 $CMES2 .SBYTE "Lower right of Bloc
k", SEOL
1460 $CMES3 .SBYTE "Upper left of Dest.
", SEOL
1470 $PRITHS .SBYTE "PRINT: Position Cu
r sor, RETURN", SEOL
1480 $LSMES .SBYTE "LIST: Position Curs
or, RETURN", SEOL
1490
1500 $FMES .SBYTE "FORMAT Justify L"
1510 $FMA .SBYTE "Dollar $(D) M"
1520 $FMB .SBYTE "Commas (,) M"
1530 $FMC .SBYTE "Dec. places 0"
1540 $YM .SBYTE "M"
1550 $LCR .SBYTE "LRC"
1560 $DEC .SBYTE "0123456789"
1570 ;raw L,C,R
1580 $JKEY .BYTE $00, $20, $12
1590 $REPMS .SBYTE "File exists, Repla
ce? Y/N", SEOL
1600 ;raw 0-9
1610 $NKEY .BYTE $32, $1F, $1E, $1A
1620 .BYTE $18, $1D, $1B, $33
1630 .BYTE $35, $30
1640 $FCNT .BYTE 0
1650 $TORITE .BYTE 0
1660 $SETRM .BYTE 0
1670 $SETCOL .BYTE 0
1680 $CDIR .BYTE 0 ;cursor directn
1690 $ENKEY .BYTE EOL, $1C, $1D, $1E, $1F
1700 $EOM .DS 2 ;end of memry
1710 $FRCTXT .BYTE 0
1720 $ORSEM .DS 1
1730 $ENDAT .DS 2
1740 $FX5 .DS 1
1750 $FXA .DS 1
1760 $STLO .DS 50
1770 $STHI .DS 50
1780 $CLCHO .BYTE $0B, $02, $14
1790 $AS26 .BYTE "ABC"
1800 $N26 .BYTE 0, 26, $2
1810 $DECM .BYTE "0123456789"
1820 $DECH .BYTE 0, 10, 20, 30, 40, 50
1830 .BYTE 60, 70, 80, 90
1840 $OPTAB .BYTE "A-Z"
1850 $FPSTACK .DS 6*50
1860 $ONE .FLOAT 1
1870 $TWO .FLOAT 2
1880 $HALF .FLOAT .5
1890 $REFTAB .SBYTE "AA"
1900 .SBYTE "01"
1910 $AIT .DS 26
1920 $SPFLG .DS 1
1930 $COLM .DS 64
1940 $FMCOL .DS 64
1950 $RAD .DS $00 ;row add tabl
1960 $ROI .DS $40 ;row data

```

TRAIN CRAZY

continued from page 11



There are four ways to lose a life: 1) by getting knocked off the train in the tunnel, 2) by hitting one of the obstacles, 3) by running out of energy and 4) by jumping too soon for the diamonds and knocking yourself out.

LISTING 1: BASIC

```

QK 1 REM *****
GT 2 REM ** TRAIN CRAZY (pt 1) **
NG 3 REM ** BY **
OA 4 REM ** COLIN FALLER **
ZD 5 REM **-----**
EZ 6 REM ** COPYRIGHT 1988 **
MZ 7 REM ** BY ANALOG COMPUTING **
QR 8 REM *****
FH 9 GRAPHICS 8:G0SUB 10:GOTO 100
TX 10 GRAPHICS 2+16:POKE 559,0:SETCOLOR 3
,1,14:SETCOLOR 4,13,8:SETCOLOR 1,0,0:5
ETCOLOR 0,6,2:SETCOLOR 2,9,4
GT 11 ? #6;"
YV 12 ? #6;"
UT 13 ? #6;"
GY 14 ? #6;"
WQ 15 ? #6;"
NX 16 ? #6;"
WS 17 ? #6;" characters 400"
LC 18 ? #6;"
EF 19 ? #6;"
YX 20 ? #6:? #6;"
LT 21 ? #6;" reading PART one":POKE 559,
34:RETURN
VX 100 RAMTOP=PEEK(106):RESTORE 2000:STAR
T=RAMTOP-8:CH=256*START
GW 109 REM CHARACTERS
PU 110 FOR I=CH TO CH+727 STEP 7:READ A1,
B1,C1,D1,E1,F1,G1:POKE I,A1:POKE I+1,B
1:POKE I+2,C1:POKE I+3,D1:POKE I+4,E1
5A 115 POKE I+5,F1:POKE I+6,G1:POSITION 1
3,6:? #6;39911-I;" ":NEXT I
UY 120 FOR I=CH+776 TO CH+994 STEP 7:READ
A2,B2,C2,D2,E2,F2,G2:POKE I,A2:POKE I
+1,B2:POKE I+2,C2:POKE I+3,D2
JA 122 POKE I+4,E2:POKE I+5,F2:POKE I+6,G
2:POSITION 13,6:? #6;39905-I;" ":NEXT
I
KA 130 TEMP=RAMTOP-20:DM=256*TEMP:FOR I=1
536 TO 1538:POKE I,112:NEXT I:BOTTOM=D
M+2816
FU 140 FOR V=2 TO 110 STEP 37:RESTORE 400
:FOR X=V TO V+36:POSITION 13,7:? #6;11
2-X;" ":READ A,B,C,D,E,F,G,H
JK 145 POKE DM+256*4+X,A:POKE DM+256*5+X,

```

```

B:POKE DM+256*6+X,C:POKE DM+256*7+X,D:
POKE DM+256*8+X,E+100
EK 147 POKE DM+256*9+X,F:POKE DM+256*10+X
,G:POKE DM+256*11+X,H
NF 150 POKE DM+256*4+X+111,A:POKE DM+256*
5+X+111,B:POKE DM+256*6+X+111,C:POKE D
M+256*7+X+111,D
WS 154 POKE DM+256*8+X+111,E+100:POKE DM+
256*9+X+111,F:POKE DM+256*10+X+111,G:P
OKE DM+256*11+X+111,H:NEXT X:NEXT U
IM 200 RESTORE 205:FOR V=1 TO 6:READ O:PO
KE DM+256*0+0,53:NEXT V
SE 205 DATA 20,58,97,133,168,206
JN 250 FOR C=224 TO 255:POKE DM+256*11+C,
39:NEXT C
RN 255 FOR O=39 TO 200 STEP 37:POKE DM+25
6*3+0,248:NEXT O:FOR O=38 TO 200 STEP
37:POKE DM+256*3+0,247:NEXT O
PN 300 RESTORE 500:FOR I=1578 TO 1637:REA
D A:POKE I,A:POSITION 13,8:? #6;1578+5
9-I;" ":NEXT I
OQ 320 RESTORE 330:FOR T=1 TO 14:READ OI:
POKE DM+256*0+OI,165:POKE DM+256*1+OI,
166:POKE DM+256*2+OI,149:NEXT T
UG 330 DATA 25,35,46,69,79,92,103,127,137
,161,175,182,197,222
DM 340 POKE DM+256*3+222,248
MT 350 RESTORE 365:FOR N=41 TO 223 STEP 3
7:READ OP:POKE DM+256*8+N,OP:POKE DM+2
56*8+N+17,OP:POKE DM+256*8+N+29,OP
EP 360 NEXT N:POKE 559,0
IR 365 DATA 159,160,161,162,163
LC 370 POKE 1575,65:POKE 1576,0:POKE 1577
,6:POKE 560,0:POKE 561,6:POKE 764,255
CS 375 POKE 53256,1:POKE 53277,2:GOSUB 10
XF 380 POSITION 2,11:? #6;"Loading PART T
WO":RUN "D:TRAIN2.BAS"
LC 399 REM *** TRAIN DATA ***
UV 400 DATA 2,3,3,3,-76,170,48,39,1,68,69
,69,33,43,1,49,1,70,71,65,58,44,1,50,1
,72,73,74,42,45,1,51,1,75,0,76,29,45
MB 401 DATA 1,39,1,75,0,76,29,46,1,39,1,7
5,0,76,29,47,1,39,1,77,78,79,29,43,1,3
9,1,80,81,82,29,43,1,49,1,75,0,76,29
ZJ 402 DATA 44,1,50,1,75,0,76,29,45,1,51,
1,75,0,76,29,25,0,39,1,77,78,79,29,1,2
5,39,1,80,81,82,29,1,25,39,1,75,0,76
IG 403 DATA 29,1,25,39,1,75,0,76,29,1,25,
39,1,75,0,76,29,1,25,39,1,77,78,79,29,
1,25,39,1,68,69,69,33,1,25,39,1,70
BC 404 DATA 71,65,58,1,25,39,1,72,73,74,4
2,1,25,39,1,75,0,76,29,1,25,39,1,75,0,
76,29,1,25,39,1,75,0,76,29,1,25,39,1
CC 405 DATA 77,78,79,29,1,25,39,1,80,81,8
2,29,25,0,39,1,75,0,76,29,43,1,49,1,75
,0,76,29,44,1,50,1,75,0,76,29,45,1,51
TV 406 DATA 1,77,78,79,29,45,1,39,1,68,69
,69,33,46,1,39,1,70,71,65,58,47,1,39,1
,72,73,74,42,43,1,39,1,75,0,76,29,43
QR 407 DATA 1,49,1,75,0,76,29,44,1,50,1,7
7,78,79,29,45,1,51,26,27,27,27,-72,169
,52,39
YQ 499 REM *** SCROLL DATA ***
JT 500 DATA 104,169,0,133,203,141,4,212,1
60,59,162,6,169,7,76,92
UU 501 DATA 228,198,203,165,203,141,4,212
,16,31,169,7,133,203
QP 502 DATA 141,4,212,238,4,6,173,4,6,201
,234,208,2,169,0,162
KS 503 DATA 0,157,4,6,232,232,232,224,39,
208,246,76,98,228
AL 599 REM *** CHARACTER DATA ***
JD 999 REM
GI 2000 DATA 0,0,0,0,0,0,0,0,255,255,255,
255,255,255,255,255,0,7,29,117,85,85,0
5,85,85,85,85,85,85,85,85,252,253
SV 2001 DATA 253,253,253,253,253,253,253,
253,253,253,253,253,253,0,255,255,
131,131,131,131,131,255,131,131,131
SG 2002 DATA 131,131,131,131,127,127,126,
126,126,126,126,127,126,126,126,126,12
6,126,126,126,126,126,126,126,126
NN 2003 DATA 127,127,127,255,24,24,24,24,
24,24,255,0,0,0,0,0,0,255,255,255,255,
127,127,127,127,127,255,127,127,127
NQ 2004 DATA 127,127,127,127,127,127,127,
127,127,127,255,255,255,255,255,254,25
4,254,254,254,255,254,254,254,254
MY 2005 DATA 254,254,254,254,254,254,254,

```

```

254,254,255,255,255,127,127,127,127,25
5,255,255,255,255,255,255,127,127,127
SF 2006 DATA 127,127,56,16,56,16,0,0,0,0,
0,255,255,255,255,255,255,255,0,127,12
7,127,127,127,127,85,85,85,85,85,85
AD 2007 DATA 85,85,127,255,255,255,255,25
5,0,0,0,0,224,184,174,170,170,170,170,
170,170,170,170,170,170,170,170
HP 2008 DATA 170,170,170,170,170,170,254,
255,195,153,145,137,153,195,255,255,23
1,199,231,231,231,129,255,255,195,153
ZG 2009 DATA 243,231,207,129,255,255,129,
243,231,243,153,195,255,255,243,227,19
5,147,129,243,255,255,129,159,131,249
ZO 2010 DATA 153,195,255,255,195,159,131,
153,153,195,255,0,0,0,0,0,0,255,254,
16,56,40,56,16,56,40,56,16,56,40
ZK 2011 DATA 56,16,56,40,0,0,0,0,255,0,25
5,255,16,40,68,130,68,40,16,0,0,224,63
,249,15,0,0,0,7,252,207,112,0,0,0
HR 2012 DATA 255,255,255,255,255,0,0,1,25
5,255,255,255,255,0,126,255,255,255,25
5,255,255,0,0,128,255,255,255,255,255
SV 2013 DATA 15,31,255,255,255,255,255,25
5,240,248,255,0,0,255,255,143,15,0,0,1
27,3,1,0,255,0,255,255,255,255,255
QV 2014 DATA 126,255,0,255,255,254,192,12
8,0,255,0,255,255,0,0,255,255,241,240,
0,0,254,16,124,16,56,16,56,16,0,255
KO 2015 DATA 255,255,255,255,128,0,0,0,0,
0,0,0,0,0,132,0,0,0,0,0,0,33,0,0,0,0
,0,0,124,254,16,56,124,254,124
JU 2059 DATA 56,16,0,127,127,127,127,127,
127,62,0
ZC 2060 DATA 0,63,127,127,127,127,127,127
FA 2061 DATA 0,255,255,255,255,255,255,0
BL 2062 DATA 0,240,252,254,254,255,255,25
5
PC 2063 DATA 0,0,0,0,255,255,255,255
EG 2064 DATA 127,127,127,127,254,254,252,
248
VT 2065 DATA 255,255,255,0,0,0,0,0
CP 2066 DATA 254,255,255,255,127,127,127,
127
EV 2067 DATA 0,7,31,63,63,127,127,127
BR 2068 DATA 255,255,255,127,127,127,127,
127
AQ 2069 DATA 127,255,255,255,255,255,255,
0
YR 2070 DATA 0,62,127,127,127,127,127,127
OD 2071 DATA 0,0,0,128,192,224,240,248
XN 2072 DATA 252,254,255,255,255,127,63,3
1
EM 2073 DATA 15,7,3,1,0,0,0,0
KZ 2074 DATA 127,127,255,255,255,255,255,
255
FC 2075 DATA 255,255,255,255,255,127,127,
127
WJ 2076 DATA 127,127,127,63,63,31,7,0
GL 2077 DATA 127,127,127,62,0,0,0,0
WV 2078 DATA 0,0,0,0,62,127,127,127
NE 2079 DATA 255,255,255,254,254,252,240,
0
JJ 2080 DATA 0,255,255,255,255,255,255,3
CQ 2081 DATA 0,254,255,255,255,255,254,25
2
DK 2082 DATA 3,7,15,15,31,63,63,127
ZR 2083 DATA 248,248,240,224,224,192,128,
128
SF 2084 DATA 0,1,1,3,7,7,15,31
NF 2085 DATA 255,254,252,252,248,240,240,
224
EQ 2086 DATA 31,63,127,127,127,127,63,0
BH 2087 DATA 192,255,255,255,255,255,255,
0
LW 2088 DATA 127,63,63,31,15,7,3,1
KM 2089 DATA 0,128,193,227,247,255,255,25
5
WK 2090 DATA 127,254,254,252,248,240,224,
192
EF 2091 DATA 198,124,124,198,198,192,192,
198
UN 2092 DATA 198,124,124,198,198,198,198,
198
LU 2093 DATA 192,254,192,192,192,192,192,
192
IH 2094 DATA 24,126,126,24,24,24,24,24
CK 2095 DATA 198,198,198,230,246,254,222,
206

```



```

PC 310 START=RAMTOP-8:CH=256*START:POKE 7
56,START
JQ 320 TEMP=RAMTOP-20:DM=256*TEMP:FOR I=1
536 TO 1538:POKE I,112:NEXT I:POKE 559
,0
JJ 321 POKE 623,2:POKE 53277,2:U=USR(1578
):?"K"
LU 323 POSITION 28,0:?"
";
DE 325 ? " " hehipq "
FE 326 POSITION 14,4:?"
udgvbe
CT 327 POSITION 37,7:?"
";
XE 328 ? " "
LH 330 FOR I=1539 TO 1572 STEP 3:POKE I,8
7:POKE I+1,0:POKE I+2,TEMP:TEMP=TEMP+1
:NEXT I
NH 333 RESTORE 335:FOR N=1 TO 20:READ OP:
POKE DM+256*3+OP,0:NEXT N:SOUND 1,0,0,
0
RM 335 DATA 53,87,156,140,100,118,170,191
,28,134,178,211,20,61,123,201,43,145,1
64,206
EM 340 SETCOLOR 1,1,14:SETCOLOR 4,13,8:SE
TCOLOR 0,0,0:SETCOLOR 3,13,8:SETCOLOR
2,9,4
FJ 365 RESTORE 366:POKE DL,76:POKE 560,35
:FOR A=3 TO 14:READ B:POKE DL+A,B:NEXT
A:POKE 559,43
DI 366 DATA 14,3,15,14,3,15,14,10,14,1,3,
6
DY 400 POKE 764,255:POKE 53249,90:TT=0:ZX
=0:POKE 752,1
JF 420 TT=0:Y=34:X=90:I=1:SC=SC+X
WX 421 POKE 53278,0:SOUND 1,0,0,0
ON 422 0=0+1:IF 0=7 THEN TT=TT+1:POSITION
34-TT+ZX,2:?" ":0=0:IF TT-ZX=28 THEN
3000
PG 423 S=STICK(0):IF S=14 THEN GOTO 600
MH 431 IF S=13 THEN I=96:X=X-1:GOTO 470
ZE 433 IF STRIG(0)=0 THEN X=X+1:GOTO 455
LR 444 GOTO 5000
BR 455 IF X>160 THEN X=160
VB 459 I=I+19:IF I>78 THEN SOUND 1,250,10
,8:I=1:SC=SC+12:SOUND 1,0,0,0
NJ 460 GOTO 500
PO 470 IF X<53 THEN GOTO 3000
RP 500 POKE 53249,X
UC 510 PM$(P1+Y,P1+Y+19)=P$(I,I+19)
LR 515 IF PEEK(53252)>10 THEN 20000
AL 516 IF PEEK(53253)>0 THEN 10000
OM 520 GOTO 422
SB 600 I=3:V=230:FOR Y=36 TO 22 STEP -2:V
=V-10:SOUND 1,V,10,5:IF PEEK(53253)>0
THEN 10000
IH 610 PM$(P1+Y,P1+Y+19)=P$(I,I+19):NEXT
Y
DD 620 I=3:V=120:FOR Y=22 TO 37 STEP 2:V
=V+10:SOUND 1,V,10,5:IF PEEK(53253)>0 T
HEN 3000
IL 630 PM$(P1+Y,P1+Y+19)=P$(I,I+19):NEXT
Y
VB 635 I=58:PM$(P1+34,P1+53)=P$(I,I+19)
IM 700 Y=34:I=58:SOUND 1,0,0,0:GOTO 422
UZ 900 POKE DM+256*3+222,0:I=3:FOR Y=37 T
O 30 STEP -1:SOUND 1,Y+Y+Y+Y+Y+Y+Y,10,
5
IK 910 PM$(P1+Y,P1+Y+19)=P$(I,I+19):NEXT
Y
KZ 920 I=22:FOR Y=30 TO 90 STEP 2:SOUND 1
,Y+Y,10,5
JG 922 I=I+19:IF I>78 THEN I=1
IO 930 PM$(P1+Y,P1+Y+19)=P$(I,I+19):NEXT
Y
UV 2000 GRAPHICS 2+16:POKE 559,0:SETCOLOR
0,1,14:SETCOLOR 4,13,8:SETCOLOR 1,0,0
:SETCOLOR 3,6,6:SETCOLOR 2,9,4
TX 2005 LEV=LEV+1:?" #6;" We'll Do "
AF 2010 ? #6;" "LeVeL ";LEV:?" #6;"
DT 2020 ? #6;" "SCoRe ";INT(5C)+215:?"
#6;"
XH 2022 ? #6;" "LiVeS ";LI-1

```

```

RF 2050 POKE 705,61:POKE 559,42
OU 2060 I=22:FOR Y=X TO 222:POKE 53249,Y:
SOUND 1,0,0,0
RR 2070 I=I+19:IF I>78 THEN SOUND 1,250,1
0,5:I=1
BZ 2080 PM$(P1+90,P1+90+19)=P$(I,I+19):PO
SITION 12,4:?" #6;INT(5C+Y):NEXT Y
UY 2090 I=115:FOR Y=10 TO 128 STEP 5
DY 2095 PM$(P1+Y,P1+Y+19)=P$(I,I+19):NEXT
Y:SOUND 1,0,0,0:POSITION 12,2:?" #6;LE
V+1
GN 2098 POSITION 7,8:?" #6;"GoD LuCk":POS
ITION 12,2:?" #6;LEV+1:CY=0
VS 2099 FOR Y=0 TO 100:CY=CY+86:SOUND 1,C
Y,10,6:NEXT Y:LL=LL+1:SOUND 1,0,0,0
CB 2130 IF LL=6 THEN LL=1
TA 2200 RESTORE 2250+LL:FOR N=1 TO 4:READ
OP:POKE DM+256*3+OP,185:NEXT N:POKE 5
59,0
BW 2210 GRAPHICS 0:POKE 559,0:RAMTOP=PEEK
(106)
ES 2220 START=RAMTOP-8:CH=256*START:POKE
756,START
DR 2225 TEMP=RAMTOP-20:DM=256*TEMP:FOR I=
1536 TO 1538:POKE I,112:NEXT I
PO 2230 FOR I=1539 TO 1572 STEP 3:POKE I,
87:POKE I+1,0:POKE I+2,TEMP:TEMP=TEMP+
1:NEXT I:GOSUB 2300:GOTO 340
YL 2251 DATA 53,87,156,140
IW 2252 DATA 100,118,170,191
QY 2253 DATA 28,134,178,211
LT 2254 DATA 20,61,123,201
QX 2255 DATA 43,145,164,206
UR 2300 POKE DM+256*3+222,248:POSITION 28
,0:?"
";
BX 2310 ? " " hehipq "
CY 2320 POSITION 14,4:?"
udgvb
eut H H H H H H H
JG 2321 POSITION 12+BN,6:?" #zyxw":POSITI
ON 6,6:?" cdsht ":POSITION 13,6:?" "
SH 2330 POSITION 37,7:?"
";
NH 2340 ? " "
":POKE 705,61:LP=0:BA=5:CK=
32
CJ 2350 RESTORE 2355:FOR V=1 TO 6:READ 0:
POKE DM+256*1+0,104:NEXT V:Q=0:GH=0
AJ 2355 DATA 20,58,97,133,168,206
HT 2360 FOR T=38 TO 200 STEP 37:POKE DM+2
56*3+T,247:POKE DM+256*3+T+1,248:NEXT
T:RETURN
IO 3000 FOR T=0 TO 255 STEP 15:POKE 705,T
:SOUND 1,T,10,8:NEXT T
CA 3001 FOR T=255 TO 0 STEP -15:POKE 705,
T:SOUND 1,T,10,8:NEXT T:SOUND 1,0,0,0:
POKE 53249,0:LI=LI-1
VE 3002 GRAPHICS 2+16:POKE 559,0:SETCOLOR
0,1,14:SETCOLOR 4,13,8:SETCOLOR 1,0,0
:SETCOLOR 3,6,6:SETCOLOR 2,9,4
NI 3005 ? #6;" " bEd LuCk":?" #6;"
DD 3010 ? #6;" "LeVeL ";LEV+1:?" #6;"
DU 3020 ? #6;" "SCoRe ";INT(5C)+215:?"
#6;"
YR 3025 ? #6;" "LiVeS ";LI
ZJ 3026 POKE 559,42:FOR U=1 TO 200:NEXT U
:IF LI=1 THEN POSITION 12,6:?" #6;"0":F
OR U=1 TO 200:NEXT U:GOTO 4000
UZ 3090 I=115:FOR Y=10 TO 128 STEP 5
HC 3095 PM$(P1+Y,P1+Y+19)=P$(I,I+19):NEXT
Y:POSITION 12,6:?" #6;LI-1
OD 3098 POSITION 7,8:?" #6;"GoD LuCk"
ZE 3099 FOR Y=2400 TO 0 STEP -25:SOUND 1,
Y+Y,10,6:NEXT Y:POKE 559,0:BN=BN-1:GOT
O 2210
ZD 4000 GRAPHICS 2+16:POKE 559,0:SETCOLOR
0,0,0:SETCOLOR 4,13,8:SETCOLOR 1,1,14
:SETCOLOR 3,6,6:SETCOLOR 2,9,4:V=0
YK 4001 POKE 559,43
KM 4005 FOR T=2 TO 12:V=V+128:FOR O=0 TO
8:NEXT O:SOUND 1,V,8,6:POSITION T,1:?"
#6;" "AmE":NEXT T:POSITION 13,1
WQ 4006 ? #6;" "":POSITION 9,0:?" #6;"
AmE"

```

Solar System Scaler

continued from page 13

Within the program you will find the following suboptions.

- "P" PRINT
- "C" CONVERT UOM
- "D" GENERATE DISTANCES
- "R" RERUN
- "M" MAIN MENU
- "SD" SCALE DISTANCES
- "AC" ACTUAL DISTANCES

As mentioned earlier, the menus are prompt-driven and you should have no difficulty traversing them. For some applications you may need to take output figures from one option and use them as input to another. Options S, E and SS provide single key transfer of the screen display to a printer, so you won't have to write down a lot of numbers between options. In addition, S and SS have menu option C (CONVERT UOM), enabling conversion of table data to other units of measure, such as inches to feet and feet to miles.

Let's take a typical scenario—say you want to determine the dimensions of the solar system with Earth at 12 inches in diameter. Start by selecting Option E (EARTH DIAMETER BASED TABLES) from the Main Menu.

Result: The following "Earth Based Ratios" menu appears.

```
EARTH BASED RATIOS
WHAT IS EARTH SCALE DIAMETER?
(Enter a number)
?
```

Type in the number 12. Press RETURN. The following prompt appears:

```
WHAT IS THE UNIT OF MEASURE (UOM)?
(For UOM enter MILES, FEET,
INCHES, MILLIMETERS or any UOM)
```

Type in "INCHES." Press RETURN. The following results print to the screen:

```
DIAMETERS OF SOLAR SYSTEM BODIES
WITH EARTH AT 12 INCHES DIAMETER.
```

DIAMETERS IN INCHES	
SOL.....	1317.064
MERCURY.....	4.59
VENUS.....	11.38
EARTH.....	12
LUNA.....	3.27
MARS.....	6.38
JUPITER.....	134.34
SATURN.....	113.46
URANUS.....	48.73
NEPTUNE.....	46.57
PLUTO.....	2.82

- P PRINT
- R RERUN
- M MAIN MENU

Now we know the diameters of the other major bodies in the solar system, in inches, when compared to a 12-inch Earth. At this point, you may want to print the screen. Press "P" (PRINT). After the data has finished printing, press "R" (RERUN).

The "Earth Based Ratio" menu reappears. For input this time type in a "1" for Earth diameter and "FEET" (or FOOT) for the UOM. The following results print to the screen:

```
DIAMETERS OF SOLAR SYSTEM BODIES
WITH EARTH AT 1 FEET DIAMETER.
```

DIAMETERS IN FEET	
SOL.....	109.755
MERCURY.....	0.38
VENUS.....	0.95
EARTH.....	1
LUNA.....	0.27
MARS.....	0.53
JUPITER.....	11.2
SATURN.....	9.45
URANUS.....	4.06
NEPTUNE.....	3.88
PLUTO.....	0.24

- P PRINT
- R RERUN
- M MAIN MENU

Now we know the diameter of the sun in feet when the Earth is 12 inches in diameter. (We will need solar diameter in feet for the next part of the sequence.) Type "M" and RETURN to return to the Main Menu. Enter Option "S" (Sun diameter based tables). The following "Sun Based Ratios" menu appears.

```
SOL BASED RATIOS
```

```
WHAT IS SCALE SUN DIAMETER?
(Enter a number)
```

?

Enter the number 109.755 (from the previous sequence). Press RETURN. The following prompt appears:

```
WHAT IS THE UNIT OF MEASURE (UOM)?
(For UOM enter MILES, FEET,
INCHES, MILLIMETERS or any UOM)
```

Type in FEET. Press RETURN. The following screen displays:

```
DIAMETERS ARE IN FEET AND ARE
BASED ON 109.755 FEET DIA. FOR SOL
```

DIAMETERS IN FEET	
MERCURY.....	0.3824
VENUS.....	0.9486
EARTH.....	1
MARS.....	0.532
JUPITER.....	11.195
SATURN.....	9.4546
URANUS.....	4.0609
NEPTUNE.....	3.8806
PLUTO.....	0.2352

- P PRINT
- R RERUN
- M MAIN MENU
- C CONVERT UOM
- D GENERATE DISTANCES

This screen shows us that Earth is one foot in diameter when the sun's diameter is 109.755 feet. That checks with what we obtained earlier from option E. Now pick option D to generate the distances of the modeled solar system. The following information prints to the screen:

```
DIAMETERS ARE IN FEET AND ARE
BASED ON 109.755 FEET DIA. FOR SOL
```

FEET DISTANT FROM SOL	
MERCURY.....	4478.92
VENUS.....	8453.16
EARTH.....	11733.5
MARS.....	17852.58
JUPITER.....	61001.6
SATURN.....	111972.9
URANUS.....	222557
NEPTUNE.....	352004
PLUTO.....	460886

- P PRINT
- R RERUN
- M MAIN MENU
- C CONVERT UOM

If you don't have a feel for the distances expressed in thousands of feet, press C (CONVERT UOM) to convert to a different unit of measure. The following submenu appears:

YOU CAN CONVERT OUTPUT FROM...	
FEET TO INCHES	(1)
INCHES TO MILLIMETERS	(2)
MILES TO FEET	(3)
FEET TO MILES	(4)
INCHES TO FEET	(5)
INCHES TO MILES	(6)
MILES TO INCHES	(7)

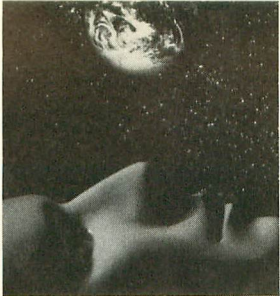
Enter "4" to convert feet to miles. The following results print to the screen:

CONVERTED TO MILES
DIAMETERS ARE IN FEET AND ARE
BASED ON 109,755 FEET DIA. FOR SOL

MILES DISTANT FROM SOL

MERCURY.....	0.85
VENUS.....	1.6
EARTH.....	2.22
MARS.....	3.38
JUPITER.....	11.6
SATURN.....	21.2
URANUS.....	42
NEPTUNE.....	66
PLUTO.....	87

P PRINT
R RERUN
M MAIN MENU



I have used this
program to build
models of the solar
system based on
diameters of hundreds
of miles to those of
only a few millimeters.

Now we have not only the sizes of bodies in the solar system (in inches and feet) when the Earth is 12 inches in diameter, but also the system's dimensions (in feet and miles).

You could take the distance figure for Pluto (87 miles) and double it to 174 miles to get the average diameter of Pluto's orbit (very average, since Pluto's orbit is far from circular, being the most highly elliptical of all the major components of the solar system). We can still use this number as input to the SS option (Solar System Based Tables) and find the distance to Alpha Centauri when the Earth is 12 inches in diameter. (It's 600,699 miles, or about 2.4 times the distance of the full-size moon from our 12-inch Earth.) On another scale, that of the sun being ten inches in diameter, Pluto would be 3,499 feet out (at 0.54 millimeters in diameter) with Alpha Centauri 4,575 miles away.

This screen sequence shows the general operational mode of the program and how information can be gathered while constructing a basic scenario. Option SS (Solar System Based Tables) enables you to generally size the solar system as a preliminary step to the other options. It also gives the distance to the nearest star. Option 12 (12-Inch Earth Globe Distance Conversions) is an added utility that comes in handy when calculating scale and real distance in a system based on a 12-inch Earth (most globes of the Earth found in the home are 12 inches in diameter).

I have used this program to build models of the solar system based on di-

ameters of hundreds of miles to those of only a few millimeters. Here are some results:

When I look at the 12-inch globe on my bookshelf five feet from my living-room easy chair, I can visualize a scale model of the Earth from a height of 39,630 miles. The Space Shuttle cruises only .23 inches above the globe's surface (150 miles). Geostationary satellites, orbiting the Earth at 22,000 miles, are out about 33.3 inches (2.78 feet) and lay sprinkled about halfway between me and the globe. The moon is a gray and dinky pockmarked baseball of 3.27 inches diameter, and is some 30 feet behind me, through the back door and out into the yard. The sun is an enormous, brilliant white ball of incandescent gas, 109.75 feet in diameter and 11,733 feet (2.22 miles) away. The gas giant

Jupiter is over 11 feet in diameter and 61,001 feet (11.6 miles) from the sun. Pluto is roughly 2.8 miles in diameter (an inch smaller than Earth's Moon!) and about 460,000 feet (87 miles away).

All of this was quickly generated using the E, S and 12 options with throughput based on a 12-inch Earth. We could, however, have generated similar results based on any size for the Earth, sun or the solar system.

In relation to the Voyager II flyby of Uranus last January, and staying with the 12-inch diameter Earth, the distance from Uranus to the sun turns out to be about 42 miles. Its diameter is 48.73 inches. That gives us some feel for the earlier question of how far two billion miles is: for a 12-inch Earth, about 44 miles.

Using the SS option, we can determine that if the solar system were a three-inch disk floating in front of us, the Alpha Centauri System would be another disk 863 feet away, or about the distance to the roadway on the other side of the strawberry field behind my house. Earth would be only .038 inches from the sun and would require a microscope for resolution. If the solar system were the size of a quarter (about one inch), the two systems would be separated by 287 feet. It's a lot easier imagining two quarters floating 287 feet apart than two multibillion-miles-wide planetary systems, composed of mainly empty space and separated by 25 trillion miles.

An infinite variety of layouts can easily be built using the *Solar System Scaler* based on manipulations of the three diameter parameters for the Earth, sun or solar system. And, with the 12 option, you can quickly calculate scale and real distances for a 12-inch Earth globe system.

Now do you have a better handle on how far two billion miles is?

Carey M. Furlong has been an amateur astronomer for the last 25 years and has operated an Atari for the last four (presently a 130XE). He holds a B.A. in English and is currently employed as a senior technical writer by MacDonnell Douglas Corporation's Information Systems Group, CAD/CAM Services in Cerritos, California.

LISTING 1: BASIC

```
HZ 100 REM SOLAR SYSTEM SIZE AND
HV 110 REM DISTANCE RATIO GENERATOR
UO 120 REM BY CAREY M. FURLONG
LV 125 REM COPYRIGHT 1988 BY ANALOG COMPU
TING
FB 130 GRAPHICS 17:SETCOLOR 2,9,7:SETCOLO
R 3,2,12:SETCOLOR 0,7,2
```

```
ZT 140 POSITION 4,9:? #6;"Solar system"
PH 150 POSITION 7,12:? #6;"Scaler"
TV 160 POSITION 0,1:? #6;"=====
=====
NT 170 POSITION 0,3:? #6;"ASTRONOMY SIMUL
ATION"
WL 180 POSITION 0,5:? #6;"=====
=====
MM 190 POSITION 0,16:? #6;"=====
=====
DG 200 POSITION 3,18:? #6;"CAREY M FURLON"
```

```

G"
HY 210 POSITION 1,19:? #6;"AND ANALOG MAG
AZINE"
BQ 220 POSITION 8,20:? #6;"1988"
KB 230 POSITION 0,22:? #6;"=====
=====
BD 240 FOR DELAY=1 TO 2500:NEXT DELAY
EW 250 DIM A$(10),E$(30),Z$(30)
OE 260 REM SYSTEM DIAMETER (MILES) & DIST
ANCE (MILLIONS OF MILES) DATA
OE 270 L=869921:REM DIAMETER OF SOL
IZ 280 B=3031:B1=35.5:REM MERCURY
EB 290 C=7518.6:C1=67:REM VENUS
HU 300 D=7926:D1=93:REM EARTH
DB 310 K=2160:REM LUNA
GF 320 E=4217:E1=141.5:REM MARS
AT 330 F=88732:F1=483.5:REM JUPITER
RF 340 G=74937.5:G1=887.5:REM SATURN
DW 350 H=32187:H1=1764:REM URANUS
AE 360 I=30757.9:I1=2790:REM NEPTUNE
HQ 370 J=1864:J1=3653:REM PLUTO
JP 380 K1=25222492:REM ALPHA CENTAURI
OR 382 REM OPEN IOCB TO SEND PROG INPUTS
TO E:
PH 385 OPEN #3,4,4,"E:"
RS 390 ? "K"
SJ 400 GRAPHICS 0
HI 410 SETCOLOR 2,0,4:SETCOLOR 1,0,12:SET
COLOR 4,8,10
JZ 415 POKE 752,1
TT 430 ? :? "SOLAR SYSTEM SCALER (
Rev. K)"
HY 440 ? "_____
ME 450 ? :? "With this program you can co
nstruct mathematically scaled models
of the Solar System."
SN 460 ? :? "Relative sizes for the plane
ts and the Sun and their respective
distances from";
HY 470 ? " one another are generat
ed when given a base measure
ment.";
AC 480 ? " Base measurements can be as
signed to either the Earth, Sun, or th
e Solar System."
IB 490 ? :? "_____
PS 500 ? "_____
XL 510 ? :? "PRESS RETURN TO CONTINUE"
BF 520 ? :INPUT #3,A$
KI 540 GRAPHICS 0:SETCOLOR 2,9,14:SETCOLO
R 1,0,4:SETCOLOR 4,8,10
KG 545 POKE 752,1
RM 550 ? "K"
MO 560 ? "
NP 570 ? "
PF 580 ? "
LO 590 ? :? :? "S SUN DIAMETER BASED T
ABLES"
ZW 600 ? :? "E EARTH DIAMETER BASED TA
BLES"
DK 610 ? :? "SS SOLAR SYSTEM DIAMETER B
ASED TABLES"
MA 620 ? :? "12 12 INCH EARTH-GLOBE DIS
TANCE CONVERSIONS"
FQ 630 ? :? "999 END PROGRAM"
HZ 640 POSITION 19,21:? "_____
OQ 650 TRAP 750
OZ 660 POSITION 1,19:? :? " ENTER SELECTI
ON ";
UA 670 INPUT #3,A$
JM 680 POKE 752,0
GA 690 GRAPHICS 0:SETCOLOR 2,0,12:SETCOLO
R 1,0,4:SETCOLOR 4,9,12
UO 700 IF A$="S" THEN 770
FC 710 IF A$="E" THEN 1760
QK 720 IF A$="12" THEN 2820
WX 730 IF A$="55" THEN 2280
PP 740 IF A$="999" THEN GRAPHICS 0:SETCOL
OR 2,9,4:SETCOLOR 4,0,0:? "K":? "END
SOLARSYS.AST (Rev. K - 11/9/86)":END
IC 750 SETCOLOR 2,2,0:SETCOLOR 4,3,0:SETC
OLOR 1,0,12:? :? "INSUFFICIENT RESPON
S E":GOTO 510
DZ 760 REM SOL TABLE GENERATOR
LU 770 ? "K":? :? " SOL BASED RA

```

```

HI 780 ? :? :? :? :? "_____
MP 790 ? :? "WHAT IS SCALE SUN DIAMETER?"
WE 800 ? :? " (Enter a number)"
HE 810 ? :INPUT 5
JC 820 VAR=1:W=0
SG 830 ? :? "WHAT IS UNIT OF MEASURE (UOM
)?"
IB 840 ? :? " (For UOM, enter MILES, FE
ET,"
LE 850 ? " INCHES, MILLIMETERS or any U
OM)"
GC 860 ? :INPUT Z$
FZ 870 ? "K":? "DIAMETERS ARE IN ";Z$;" A
ND ARE"
XR 880 SETCOLOR 4,8,2:? "BASED ON ";S;" "
;Z$;" DIA. FOR SOL"
YN 890 ? "_____
PW 900 ? "_____
WW 910 IF W<>0 THEN 930
XL 920 ? :? " DIAMETERS IN ";Z$:GOT
0 940
WH 930 ? :? " DIAMETERS IN ";E$
TW 940 X=5*B/L:X1=INT(X*10000+0.5)/10000:
? :? " MERCURY.....";X1*VAR
AV 960 X=5*C/L:X1=INT(X*10000+0.5)/10000:
? " VENUS.....";X1*VAR
CP 980 X=5*D/L:X1=INT(X*10000+0.5)/10000:
? " EARTH.....";X1*VAR
NJ 1000 X=5*E/L:X1=INT(X*10000+0.5)/10000
:? " MARS.....";X1*VAR
AO 1020 X=5*F/L:X1=INT(X*10000+0.5)/10000
:? " JUPITER.....";X1*VAR
FR 1040 X=5*G/L:X1=INT(X*10000+0.5)/10000
:? " SATURN.....";X1*VAR
HV 1060 X=5*H/L:X1=INT(X*10000+0.5)/10000
:? " URANUS.....";X1*VAR
VA 1080 X=5*I/L:X1=INT(X*10000+0.5)/10000
:? " NEPTUNE.....";X1*VAR
FM 1100 X=5*J/L:X1=INT(X*10000+0.5)/10000
:? " PLUTO.....";X1*VAR
LH 1120 POKE 838,175:POKE 839,242
SA 1130 IF A$="P" THEN GRAPHICS 0:SETCOLO
R 2,0,12:SETCOLOR 1,0,4:SETCOLOR 4,9,1
2:A$="M":? :? :GOTO 880
BX 1140 GOSUB 3620
ZS 1150 POKE 82,0:POKE 83,40
GU 1160 ? :? "_____
FF 1170 IF W<>0 THEN 1190
YE 1180 ? " P PRINT C CONVERT UO
M":GOTO 1200
HI 1190 ? " P PRINT"
ZU 1200 ? " R RERUN D GENERATE D
ISTANCES"
RH 1210 ? " M MAIN MENU"
WT 1220 ? " SEL
ECT ";
IJ 1230 POKE 82,2:POKE 83,39
DA 1240 INPUT A$
OG 1250 IF A$="P" THEN GOSUB 2780
QX 1260 IF A$="R" THEN 880
WZ 1270 IF A$="D" THEN 1340
WE 1280 IF A$="R" THEN A$="5":GOTO 690
JY 1290 IF A$="M" THEN 540
UF 1300 IF W<>0 THEN A$="5":GOTO 690
BV 1310 IF A$="C" THEN GOSUB 3450
EN 1320 GOSUB 3670
TY 1330 GOTO 880
IL 1340 GRAPHICS 0:SETCOLOR 2,0,12:SETCOL
OR 1,0,4:SETCOLOR 4,8,2
YR 1350 ? "K":? "_____
PE 1360 VAR=1:W=0
AL 1370 ? "DISTANCES ARE IN ";Z$;" AND AR
E"
WO 1380 SETCOLOR 4,8,2:? "BASED ON ";S;" "
;Z$;" DIA. FOR SOL"
QM 1390 IF W=0 AND A$="M" THEN ?
EJ 1400 ? "_____
LN 1410 R=0.869921
DD 1420 IF W<>0 THEN 1440
IT 1430 ? :? " ";Z$;" DISTANT FROM S

```



```

OL":GOTO 1450
VJ 1440 ? :? :? " "E$;" DISTANT FRO
M SOL"
VC 1450 X=INT((5*B1/R*VAR)*100+0.5)/100:?
:?" MERCURY .....";X
TD 1460 X=INT((5*C1/R*VAR)*100+0.5)/100:?
" VENUS .....";X
WF 1470 X=INT((5*D1/R*VAR)*100+0.5)/100:?
" EARTH .....";X
NX 1480 X=INT((5*E1/R*VAR)*100+0.5)/100:?
" MARS .....";X
UT 1490 X=INT((5*F1/R*VAR)*10+0.5)/10:? "
JUPITER .....";X
DC 1500 X=INT((5*G1/R*VAR)*10+0.5)/10:? "
SATURN .....";X
FE 1510 X=INT((5*H1/R*VAR)*10+0.5)/10:? "
URANUS .....";X
OZ 1520 X=INT((5*I1/R*VAR)*10+0.5)/10:? "
NEPTUNE .....";X
IX 1530 X=INT((5*J1/R*VAR)*10+0.5)/10:? "
PLUTO .....";X
LV 1540 POKE 838,175:POKE 839,242
RN 1550 IF A$="P" THEN GRAPHICS 0:SETCOLO
R 2,0,12:SETCOLOR 1,0,4:SETCOLOR 4,8,2
:A$="M":? :? :GOTO 1380
CL 1560 GOSUB 3620
AG 1570 POKE 82,0:POKE 83,40
HI 1580 ? :? "
"
DD 1590 IF W<>0 THEN 1610
IU 1600 ? " P PRINT C CONVERT UO
M":GOTO 1620
GU 1610 ? " P PRINT"
HJ 1620 ? " R RERUN"
RV 1630 ? " M MAIN MENU"
XH 1640 ? " SEL
ECT ";
IX 1650 POKE 82,2:POKE 83,39
DO 1660 INPUT A$
OU 1670 IF A$="P" THEN GOSUB 2780
GY 1680 IF A$="R" THEN 1380
WP 1690 IF A$="M" THEN A$="S":GOTO 690
JH 1700 IF A$="M" THEN 540
UQ 1710 IF W<>0 THEN A$="S":GOTO 690
CG 1720 IF A$="C" THEN GOSUB 3450
EY 1730 GOSUB 3670
SY 1740 GOTO 1380
XR 1750 REM EARTH TABLE GENERATOR
TJ 1760 ? "K":? :? " EARTH BASED
RATIOS"
TE 1770 ? :? :? :? :? "
"
UJ 1780 ? :? "WHAT IS SCALE EARTH DIAMETE
R?"
XV 1790 ? :? " (Enter a number)"
PT 1800 ? :INPUT 5
YP 1805 VAR=1
MM 1810 ? :? "WHAT IS UNIT OF MEASURE (UO
M)?"
HO 1820 ? :? " (For UOM, enter MILES, F
EET,"
OB 1830 ? " INCHES, MILLIMETERS or any
UOM)"
QK 1840 ? :INPUT Z$
MT 1850 ? "K":SETCOLOR 4,8,2:? "DIAMETERS
OF SOLAR SYSTEM BODIES"
DR 1860 ? "WITH EARTH AT ";S;" ";Z$;" DIA
METER"
FM 1870 ? "
"
PM 1890 ? " DIAMETERS IN ";Z$
YE 1900 X=5*L/D:X1=INT(X*1000+0.5)/1000:?
:?" SOL .....";X1*VAR
XV 1910 X=5*B/D:X1=INT(X*100+0.5)/100:? "
MERCURY .....";X1*VAR
TJ 1930 X=5*C/D:X1=INT(X*100+0.5)/100:? "
VENUS .....";X1*VAR
XM 1950 X=5*D/D:X1=INT(X*100+0.5)/100:? "
EARTH .....";X1*VAR
LD 1970 X=5*K/D:X1=INT(X*100+0.5)/100:? "
LUNA .....";X1*VAR
PN 1990 X=5*E/D:X1=INT(X*100+0.5)/100:? "
MARS .....";X1*VAR
PD 2010 X=5*F/D:X1=INT(X*100+0.5)/100:? "
JUPITER .....";X1*VAR
XA 2030 X=5*G/D:X1=INT(X*100+0.5)/100:? "
SATURN .....";X1*VAR

```

```

ZB 2050 X=5*H/D:X1=INT(X*100+0.5)/100:? "
URANUS .....";X1*VAR
KB 2070 X=5*I/D:X1=INT(X*100+0.5)/100:? "
NEPTUNE .....";X1*VAR
BA 2090 X=5*J/D:X1=INT(X*100+0.5)/100:? "
PLUTO .....";X1*VAR
LF 2110 POKE 838,175:POKE 839,242
YG 2120 IF A$="P" THEN GRAPHICS 0:SETCOLO
R 2,0,12:SETCOLOR 1,0,4:SETCOLOR 4,8,2
:A$="M":? :? :GOTO 1850
BV 2130 GOSUB 3620
ZQ 2140 POKE 82,0:POKE 83,40
GS 2150 ? :? "
"
HA 2160 ? " P PRINT"
HP 2170 ? " R RERUN"
SB 2180 ? " M MAIN MENU"
XN 2190 ? " SEL
ECT ";
IB 2200 POKE 82,2:POKE 83,39
CS 2210 INPUT A$
NY 2220 IF A$="P" THEN GOSUB 2780
IE 2230 IF A$="R" THEN A$="E":GOTO 690
JK 2240 IF A$="M" THEN 540
EV 2250 GOSUB 3670
TO 2260 GOTO 1850
KU 2270 REM SOLAR SYSTEM TABLE GENERATOR
XO 2280 ? "K":? :? " SOLAR SYSTEM BAS
ED RATIO"
TB 2290 ? :? :? :? :? "
"
UJ 2300 ? :? "WHAT IS SCALE SOLAR SYSTEM
DIAMETER?"
WQ 2310 ? :? " (Enter a number)"
PQ 2320 ? :INPUT 5
OW 2330 VAR=1:W=0
MM 2340 ? :? "WHAT IS UNIT OF MEASURE (UO
M)?"
RQ 2350 ? :? " (For UOM enter MILES, FE
ET,"
OB 2360 ? " INCHES, MILLIMETERS or any
UOM)"
QK 2370 ? :INPUT Z$
LV 2380 ? "K"
YQ 2390 SETCOLOR 4,8,2:? "SOLAR SYSTEM DI
A. = ";S;" ";Z$
CZ 2395 IF W<>0 THEN 2410
QF 2400 ? "(Based on Pluto's Mean Orbit)"
EN 2410 ? "
"
DX 2420 IF W<>0 THEN 2440
PQ 2430 ? " DISTANCE IN ";Z$;" FROM 50
L":GOTO 2450
JB 2440 ? " DISTANCE IN ";E$;" FROM 50
L"
EU 2450 R=J1/0.5
JG 2460 X=5*B1/R:X1=INT(X*1000+0.5)/1000:
? :? " MERCURY .....";X1*VAR
VN 2470 X=5*C1/R:X1=INT(X*1000+0.5)/1000:
? :? " VENUS .....";X1*VAR
XH 2480 X=5*D1/R:X1=INT(X*1000+0.5)/1000:
? :? " EARTH .....";X1*VAR
NN 2490 X=5*E1/R:X1=INT(X*1000+0.5)/1000:
? :? " MARS .....";X1*VAR
VP 2500 X=5*F1/R:X1=INT(X*1000+0.5)/1000:
? :? " JUPITER .....";X1*VAR
BN 2510 X=5*G1/R:X1=INT(X*1000+0.5)/1000:
? :? " SATURN .....";X1*VAR
DN 2520 X=5*H1/R:X1=INT(X*1000+0.5)/1000:
? :? " URANUS .....";X1*VAR
PW 2530 X=5*I1/R:X1=INT(X*1000+0.5)/1000:
? :? " NEPTUNE .....";X1*VAR
PP 2540 X=5*J1/R:X1=INT(X*1000+0.5)/1000:
? :? " PLUTO (Mean Dist) ..";X1*VAR
SH 2550 X=5*K1/R:X1=INT(X*100+0.5)/100:?
:?" ALPHA CENTAURI...";X1*VAR
MC 2560 POKE 838,175:POKE 839,242
KD 2570 IF A$="P" THEN GRAPHICS 0:SETCOLO
R 2,0,12:SETCOLOR 1,0,4:SETCOLOR 4,8,2
:A$="M":? :? :GOTO 2390
CS 2580 GOSUB 3620
AN 2590 POKE 82,0:POKE 83,40
GN 2600 ? :? "
"
ER 2610 IF W<>0 THEN 2630
OQ 2620 ? " P PRINT C CONVERT UO
M":GOTO 2640

```


Game Design Workshop



by Craig Patchett

An Introduction

Hi, and welcome to the column! This is just a brief introduction to let you know what we're going to be doing together in the pages to come.

Chances are that one of the biggest reasons you bought your computer (whether you admit it or not) was to play games.

Then somewhere along the way you discovered programming, and then came the realization that, "Hey, even I could write a game if I only knew how." Well, maybe it didn't happen quite like that, but I'm sure you know what I mean. In any case, it's true that writing a game isn't really that difficult, at least not a simple game. But unfortunately there aren't too many people willing to explain the techniques

involved. You know what I mean, right? After all, chances are you've read all about graphics already, but there's quite a difference between making a spaceship fly around the screen and putting it into a game. That's why this column is here: to explain how to tie everything—including graphics, animation, and sound—together into a complete graphics game. I'll be assuming that you already know the

basic techniques involved in player/missile graphics, redefining the character set, display lists and so forth. Although the upcoming columns do include review sections on the topics they cover, it will help you tremendously if you have some previous knowledge.

So much for the graphics. What about the language? We are going to be using BASIC. I'm sure you've read plenty of articles claiming that a good arcade-style game can't be written in BASIC, and it's true. So to help you overcome this obstacle, I've included a variety of machine-language routines that can be used from BASIC quite easily. Rest assured that you don't have to know any machine language in order to use these routines, and they've been designed so that they'll work in a number of different situations, not just the game we'll be designing in this column.

Even with the machine-language help, it's still true that BASIC will not allow you to do games as complicated as *Donkey Kong* or *Defender*, at least not with the speed of the arcade versions. So the question comes up as to what would be a good game to use as an example. We need something that utilizes as many of the graphics tricks as possible, while at the same time isn't too complicated so as to slow things down to a ridiculous rate. Also, it should ideally be something familiar, so that it is immediately clear how things should be working. What, if anything, fits these conditions? We decided on *Invaders* as our model.

Graphics fundamentals

A big part of computer graphics, and one that not many people think about, is the television set. After all, we're not drawing on a piece of paper, and the way that a TV set works has a lot to do with what the computer puts on the screen. More importantly, it has a lot to do with *how* the computer puts things on the screen. So, as one final step before we actually start going wild, let's take a good look at that screen.

The Screen

Let's begin with the basics, since I can't think of a better place. Do you know what the screen is made of? Well, I know that glass is probably the most obvious answer, but it goes much further than that. Look at the screen of a TV set that's not on. See how the back of the glass is coated with something? That something is called "phosphor." The neat thing about phosphor is that it glows when you hit it with electrons. The more electrons that hit it,

the brighter it glows (don't worry if you don't know what electrons are, just think of them as real tiny pellets. . . real tiny).

Behind the screen (near the back of the TV set), there is an electron gun that shoots a beam of electrons at the screen to make it glow. Pretty simple so far, right? This beam is pretty small and can only hit a small dot of phosphor at any one time. In order for it to make all of the phosphor on the screen glow, then, it scans the screen, much in the same way as we scan a book while reading. The beam starts at the upper left-hand corner of the screen and goes all the way over to the right-hand side, drawing a line as it goes. This line is called, appropriately, a "scan line" (if you look closely at your screen, you can see these lines; don't look for too long, though). When it gets to the right-hand side of the screen, the beam turns off and moves over to the beginning of the next line, where it preceeds to draw the next line. This happens over and over again until the beam has drawn the entire screen. (See Figure 1.)

There are over 200 lines that have to be drawn, but the beam gets it all done in less than 1/60 of a second! Unfortunately, the phosphor won't glow forever, and

by the time the beam reaches the bottom of the screen, the top is starting to fade. So the beam turns off, goes back to the top again and starts over. This means that the screen actually gets drawn 60 times a second. Now this may sound like a pain in the you-know-what, but it actually turns out to be somewhat of a blessing. You've probably seen, in one form or another, those little cartoon books that you flip through to make the pictures look like they're moving, right? Movies work the same way. They show you a picture, then blank the screen and show you another one that's just a little bit different. This is what creates the illusion of motion. Thus, if a different picture is drawn on the screen each time the beam goes over it, we can create motion. If the beam didn't redraw the screen so rapidly, we'd just have an expensive slide show.

Alright, now you know the basics of television, but what does this have to do with computers? Quite a bit, actually, as you'll soon see. If you're programming from BASIC, then all you need to worry about is knowing the various terms. We'll go over those in a second. Machine-language programmers, on the other hand, are programming in a language that works almost as fast as the electron beam, so they can have greater control over the screen. That's why so many incredible things can be done from machine language that can't be done from BASIC. Machine-language programmers need to know exactly how the beam draws the screen, because they can take advantage of it. So if you're a BASIC programmer, this short instruction to the world of the TV set has been a learning experience (surprise). If you program in machine language, there's a lot more you can learn if you really want to be able to make the computer say "Uncle." Not many people do.

Terms, terms, terms. Actually, in this column there are only a few that we'll be using, but the above explanation of the TV set was necessary for you to be able to completely understand what they mean. So, without any further ado, here they are:

Scan Line: This one you already know. A scan line is simply a horizontal line on the screen, as drawn by the electron beam. The screen is made up of several hundred scan lines, one on top of the other.

Vertical Blank: Also known as VBLANK and Vertical Refresh. I prefer VBLANK. In any case, VBLANK is the time during which the electron beam is off and on its way from the bottom of the screen back to the top. It happens 60



times a second, since that's how often the beam draws the screen.

Horizontal Blank: Also known, strangely enough as HBLANK and Horizontal Refresh. Guess which I prefer! HBLANK is the time during which the electron beam is off and on its way from the end

of one scan line to the beginning of the next. HBLANK happens a lot more often than VBLANK (over 200 times more often).

Well, that's it! That's right, all of this for only three simple terms. In the columns ahead, however, you'll be seeing these

terms a lot, especially "scan line" and "VBLANK," so please make sure they become a part of your vocabulary.

A little bit more

This section is for those of you with a

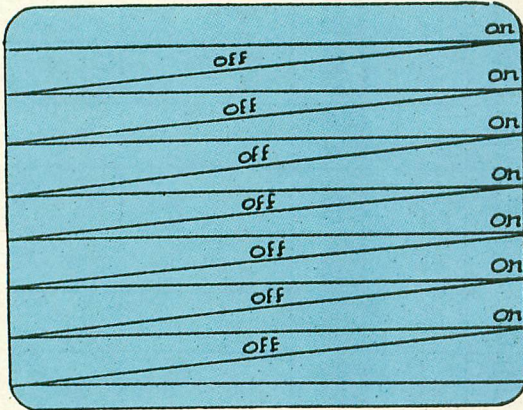


Figure 1.

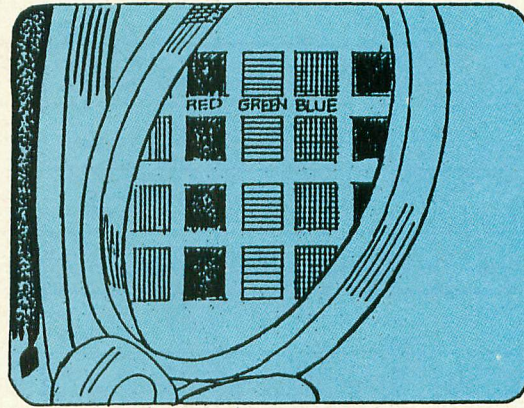


Figure 2.

TEXT MODES SUMMARY

Display Type	Graphics Mode Number	Screen Size (column x rows)	Memory Used (bytes)	Colors Available	Foreground Color Register	Background Color Register	Border Color Register
Normal text	0	40 X 24	992	1 color 2 luminances	1	2	4
Double-width text	1	20 X 20, split 20 X 24, full	674, split 672, full	5	0,1,2,3	4	4
Double-width, double-height text	2	10 X 20, split 12 X 20, full	424, split 420, full	5	0,1,2,3	4	4
Multicolor text	12	40 X 20, split 40 X 24, full	992	5	0,1,2,3	4	4
Large multi-color text	13	40 X 10, split 40 X 12, full	512	5	0,1,2,3	4	4

Table 1

Table 2

Column 1		Column 2		Column 3		Column 4	
#	CHR	#	CHR	#	CHR	#	CHR
0	Space	16	0	32	@	48	P
1	!	17	1	33	A	49	Q
2	”	18	2	34	B	50	R
3	#	19	3	35	C	51	S
				64	⬇	80	⬆
				65	⬆	81	⬇
				66	⬆	82	⬆
				67	⬆	83	⬆
				96	⬆	112	p
				97	a	113	q
				98	b	114	r
				99	c	115	s

Illustration © Copyright Atari, Corp.

4	\$	20	4	36	D	52	T	68		84		100	d	116	t
5	%	21	5	37	E	53	U	69		85		101	e	117	u
6	&	22	6	38	F	54	V	70		86		102	f	118	v
7	'	23	7	39	G	55	W	71		87		103	g	119	w
8	(24	8	40	H	56	X	72		88		104	h	120	x
9)	25	9	41	I	57	Y	73		89		105	i	121	y
10	*	26	:	42	J	58	Z	74		90		106	j	122	z
11	+	27	;	43	K	59	[75		91		107	k	123	
12	,	28	<	44	L	60	\	76		92		108	l	124	
13	-	29	=	45	M	61]	77		93		109	m	125	
14	-	30	>	46	N	62	^	78		94		110	n	126	
15	/	31	?	47	O	63	_	79		95		111	o	127	

Illustration © Copyright Atari, Corp.

1. In mode 0 these characters must be preceded with an escape, CHR\$(27), to be printed.

Table 3

ATASCII CHARACTER SET			DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER									
0	0		13	D		26	1A		39	27	'	55	37	7	71	47	G	87	57	W	103	67	g	119	77	w
1	1		14	E		27	1B		40	28	(56	38	8	72	48	H	88	58	X	104	68	h	120	78	x
2	2		15	F		28	1C		41	29)	57	39	9	73	49	I	89	59	Y	105	69	i	121	79	y
3	3		16	10		29	1D		42	2A	*	58	3A	:	74	4A	J	90	5A	Z	106	6A	j	122	7A	z
4	4		17	11		30	1E		43	2B	+	59	3B	;	75	4B	K	91	5B	[107	6B	k	123	7B	
5	5		18	12		31	1F		44	2C	,	60	3C	<	76	4C	L	92	5C	\	108	6C	l	124	7C	
6	6		19	13		32	20	Space	45	2D	-	61	3D	=	77	4D	M	93	5D]	109	6D	m	125	7D	
7	7		20	14		33	21	!	46	2E	.	62	3E	>	78	4E	N	94	5E	^	110	6E	n	126	7E	
8	8		21	15		34	22	"	47	2F	/	63	3F	?	79	4F	O	95	5F	_	111	6F	o	127	7F	
9	9		22	16		35	23	#	48	30	0	64	40	@	80	50	P	96	60		112	70	p			
10	A		23	17		36	24	\$	49	31	1	65	41	A	81	51	Q	97	61	a	113	71	q			
11	B		24	18		37	25	%	50	32	2	66	42	B	82	52	R	98	62	b	114	72	r			
12	C		25	19		38	26	&	51	33	3	67	43	C	83	53	S	99	63	c	115	73	s			
						39	27	'	52	34	4	68	44	D	84	54	T	100	64	d	116	74	t			
						40	28	(53	35	5	69	45	E	85	55	U	101	65	e	117	75	u			
						41	29)	54	36	6	70	46	F	86	56	V	102	66	f	118	76	v			

Illustration © Copyright Atari Corp.

little bit of curiosity. You don't need to read it, but you may find it interesting nonetheless. Way back near the beginning of this column, I mentioned that the greater the number of electrons that hit the phosphor on the screen, the brighter the phosphor glows. What I didn't say was that this is very important to the creation of the picture. After all, a black and white picture isn't all black and white, is it? There are all kinds of grays as well.

As the electron beam scans the screen, its intensity varies, thus varying the number of electrons that hit the screen. This, in turn, causes the brightness of the screen to vary between black (no electrons) and white (lots of electrons), allowing for all the shades of gray. This is how a black and white picture is created.

But what about color? Well, color pictures are nothing more than black and white ones with color added. Now I know that sounds kind of obvious, but it's an important thing to keep in mind. A color picture is drawn in exactly the same way as a black and white one, except in color. The question remains, of course, as to how that color is created.

If you take a close look at the screen of most color TVs, you should be able to see little red, green and blue stripes. These stripes are so small that their colors blend together when you look at the screen from a normal distance. Some of you may know the significance of these three colors already. We call them the electronic primary colors because all other known colors can be created by mixing one or more of the three together. For example, mixing yellow with blue gives green, and yellow with red gives orange. Can you see now what's going on? A color TV actually draws three pictures on the screen; a red and white one, a green and white one and a blue and white one. The three blend together to give you a genuine full-color picture. (See Figure 2.)

And you thought it was something complicated! Anyway, that's about all I wanted to tell you about your television set. If you have a monitor, don't worry—the basic principals are the same.

Screen memory

So far we know that a TV picture is drawn with an electron beam that scans across the screen. But how does the beam know what needs to be drawn? In the case of regular TV, it receives a signal from your antenna or cable system. In the case of the computer, the computer itself sends out the information. Inside your Atari is a chip called GTIA which is responsible

for controlling the electron beam. Actually, it just controls the intensity of the beam. As you might recall, the more intense the beam, the brighter the phosphor; so a picture is drawn by rapidly varying the intensity of the beam. Anyway, that's GTIA's responsibility. GTIA, however, still needs to know what it's supposed to be telling the electron beam, right? Somewhere inside the computer then, there must be a description of what the screen is to look like. There is, and it's in a special section of memory called, strangely enough, screen memory. In this section, we're going to take a look at how screen memory works.

Before we get into the heavy details, let's look at the basic idea. Screen memory is made up, like all other memory, of a whole bunch of bytes. In one way or another (we'll get into it later), GTIA looks at these bytes, interprets them, and then sends the information to the electron gun. The first byte in screen memory will determine what the upper left-hand corner of the screen will look like, the last will determine what the lower right-hand corner looks like. That should make sense, of course, since that's the order that the electron gun would expect. Depending on the graphics mode being used, the screen can be anywhere from ten to 40 bytes across, and from 12 to 192 down. We'll see why in a little bit. For now, let's take a look at just how GTIA interprets the bytes in screen memory.

Character-graphics modes 0, 1 & 2

We'll begin by taking a look at the character-graphics modes (GRAPHICS 0/1/2). In these modes, there is one byte in screen memory for each possible character on the screen. This is summarized in Table 1.

Do you know what ATASCII values are? Each Atari character has a number assigned to it, between 0 and 255, and these numbers are called ATASCII (ATARI Standard Code for Information Interchange) values. You've also probably heard of something called the "Internal Values." The internal values are what we are interested in, since these values are the ones used by screen memory and GTIA. (See Tables 2 and 3.) As a matter of fact, all screen memory in the character modes is a list of these internal values. GTIA looks at this list, sees what is currently on, then looks into another list called the "character set" (we'll go over the character set in a future column) to see what this particular character looks like. GTIA then sends this information to the electron gun which in turn causes the beam to draw the

character (it's not really quite that simple, but this is all you need to understand). Why don't we see an example of this in action. First we'll find out where screen memory is. The following line will do the trick for us:

```
100 SCRMEM=PEEK (88)+PEEK (89)*
256
```

Memory locations 88 and 89 hold the address of the beginning of screen memory.

Now let's try POKEing the internal value for "HI" into screen memory. The letter "H" has an internal value of 40, and "I" has an internal value of 41. So we do the following:

```
110 POKE SCRMEM,40:POKE SCRM
EM+1,41
```

Viola! We've now changed screen memory. If you don't have the word "HI" in the upper left-hand corner of your screen right now, it's probably because your screen is full and it got scrolled off. Add the following line to the program and try again:

```
99 GRAPHICS 0
```

What if we wanted to have "HI" written vertically instead of horizontally? Try making the following change:

```
110 POKE SCRMEM,40:POKE SCRM
EM+40,41
```

Do you understand why we added 40 to SCRMEM to get the "I" to print below the "H"? Remember that graphics mode 0 has 40 characters on each line, and that each character takes up one byte in screen memory. That means that the first character on the second line is the 41st byte in screen memory. SCRMEM is the address of the first byte, so SCRMEM + 40 is the address of the 41st. Now try changing Line 99:

```
99 GRAPHICS 1
```

What happened? Now the "I" is two lines below the "H." You should be able to figure out why by yourself this time, but I'll give you a hint just in case: There are only 20 characters on a graphics mode 1 line.

Before we leave the wonderful world of character graphics, we need to bring up a few more things. First of all, if you've looked at the ATASCII table yet, you've

probably noticed that there are only 128 characters in the character set. But each byte in screen memory can hold a number between 0 and 255, or a total of 256 numbers. What about the other 128 characters? Well, if GTIA sees a character value greater than 127 (in other words, if bit seven is set) then it takes the value, subtracts 128 from it to get a new value, and then puts the character with this new value up on the screen in inverse video. Inverse video is what you get when you type a character after pressing the inverse key (the key with either the Atari logo or a half-full square on it, depending on which Atari computer you have). So, for example, an inverse video "H" would have a value of 168 (40 + 128).

As if that's not enough, we still have something else to clear up; only this time it concerns graphics modes 1 and 2. In these modes, you can only use half the character set at a time (either uppercase and numbers or lowercase and graphics), and there is no inverse video. Why? Because in these modes, bits six and seven of the character values are used to specify the character color, which means that characters in these modes can be one of four colors. Here's how it works: GTIA gets the character value and looks at bits six and seven. Then, depending on how the bits are set, it goes to one of the Atari color registers to see what color is desired and puts the character up on the screen in that color. A color register is just a memory location that affects the colors of objects on the screen. When you use

the SETCOLOR command, you are actually just setting the value in one of the registers. Anyway, here's a list of the different bit combinations and the color registers they refer to:

BITS	REGISTER	LOCATION
00	0	708
01	1	709
10	2	710
11	3	711

Here's a simple program that will put the letter "H" on the screen in all four colors:

```

100 GRAPHICS 2
110 SCRMEM=PEEK(88)+PEEK(89)
*256
120 POKE SCRMEM+1,40
130 POKE SCRMEM+2,40+64
140 POKE SCRMEM+3,40+128
150 POKE SCRMEM+4,40+192

```

After running this program, you might like to play with the SETCOLOR command to see the effect of changing the various color registers. By the way, to switch to lowercase, try POKE 756,226. Use POKE 756,224 to get back to uppercase.

Now that we (hopefully) have a grasp of how GTIA and screen memory work with characters, let's take a look at graphics. We'll start with graphics mode 8, since it's a little easier to understand than the others. In graphics mode 8, the screen is made up of some 61,440 dots that can be turned on or off (320 across by 192 down). Does that phrase "on or off" ring a bell?

We ran across it before when we were talking about bits which, I'm sure you'll recall, can be turned on or off just like the dots on the screen. The way graphics mode 8 works then, is that each bit in screen memory represents a dot on the screen. Turn a bit on and the corresponding dot will also turn on. Turn it off and, well, you guessed it. Because there are 8 bits in a byte, and there are 320 dots across a graphics mode 8 screen, it takes 40 bytes (320/8) of screen memory for each graphics mode 8 line. That's a total of 7,680 (40*192 or 61,440/8) bytes of screen memory altogether, which is obviously a lot better than 61,440! Now this system does have its disadvantages, since one byte holds eight dots instead of just one. Turning individual bits on and off can be very frustrating, especially from BASIC. In other words, there are very few times when you'll want to go in and change this kind of screen memory directly. It's a lot easier to PLOT and DRAWTO. In one of the upcoming columns, however, we will see a time when the ability to change it directly will come in handy. In the meantime, let's take a look at the other graphics modes.

The other graphics modes

Graphics modes 4 and 6 are similar to graphics mode 8 in that each dot can either be on or off. Thus screen memory works in exactly the same way as we just discussed. The difference is that the dots are larger, which means there are less of them on the screen, which in turn means

GRAPHICS MODES 4 & 6 SUMMARY

Display Type	Screen Size (columns x rows)	Memory Used (bytes)	Default Colors	SETCOLOR (n)	POKE address	COLOR (n)
Graphics 4	80 X 40, split	694, split	Orange	0	708	1
	80 X 48, full	696, full				
Graphics 6	160 X 80, split	2174, split	Black	4	712	0
	160 X 96, full	2184, full				

TABLE 4

Applies to both modes.

GRAPHICS MODES 3, 5, & 7 SUMMARY

Display Type	Screen Size (columns x rows)	Memory Used (bytes)	Default Colors	SETCOLOR (n)	POKE address	COLOR (n)
Graphics 3	40 X 20, split	434, split	Orange	0	708	1
	40 X 24, full	432, full				
Graphics 5	80 X 40, split	1174, split	Light-green	1	709	2
	80 X 48, full	1176, full				
Graphics 7	160 X 80, split	4190, split	Blue	2	710	3
	160 X 96, full	4200, full	Black	4	712	0

TABLE 5

Applies to all three modes.

that screen memory is smaller. Table 4 summarizes these differences.

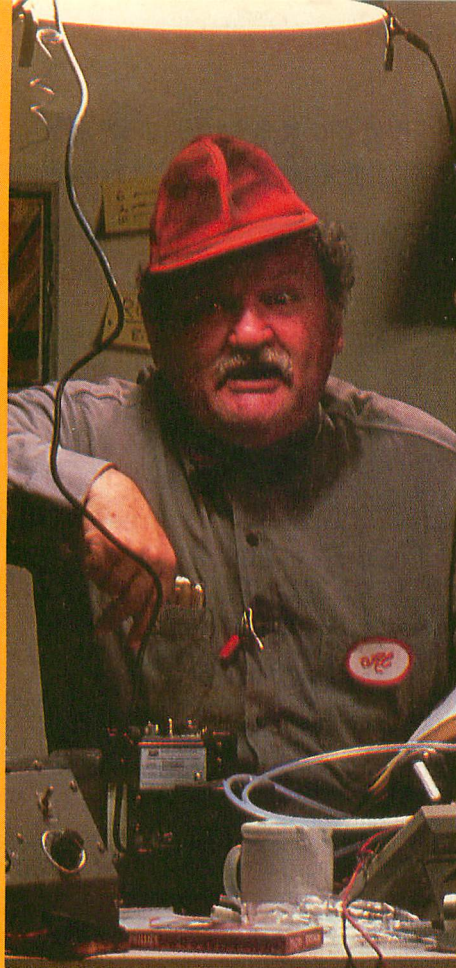
Graphics modes 3, 5 and 7 introduce a new twist. In these three modes, not only can the dots be on but when they are on, they can have one of three colors. This means that there are four possibilities for each dot instead of just two. How do we deal with this? Actually, the solution is quite simple; we use two bits for each dot instead of just one. Two bits can be arranged in four combinations (00,01,10,11), which is exactly what we need. So, in the four color modes, each byte of screen memory holds four dots instead of eight. Table 5 has all the information.

Before we continue, there's something I forgot to explain. GTIA knows from the two bits which of the four colors to give a particular dot, but how does it know what the four colors are? After all, the Atari has a total of 256 to choose from. It turns out we already know the answer from our discussion on graphics modes 1 and 2 above, so there's no point in going over it again here. There is a small difference, however, in that the bit combinations don't refer to the same color registers as they did for the characters. The following table shows how it works for graphics:

BITS	REGISTER	LOCATION
00	BACKGROUND	712
01	0	708
10	1	709
11	2	710

We've now covered all the modes except three, and those three aren't too tough to cover since they work in much the same way as the four color modes we just discussed. Before we take a look at them, however, I should point out that not all Atari computers have them. You see, the first Ataris to roll off the assembly lines didn't have the GTIA chip. Instead, they had a stripped-down version called CTIA. The only real difference between the two chips is that CTIA doesn't have the three modes we are going to talk about, namely graphics modes 9, 10 and 11.

We'll look at modes 9 and 11 first, since they are perhaps a little more straightforward than mode 10. In these two modes, you can have 16 (count 'em) colors on the screen at once, but with a sacrifice. The dots are as high as the ones in graphics mode 8, but they're four times as wide. This makes the screen 80 dots wide by 192 high. Why are the dots so wide? Well, it takes four bits, or half a byte, to store a number between 0 and 15, which is what is needed to get 16 colors. That translates



to 32K of screen memory if the dots were to be the same size as those in graphics mode 8. Even if they were only twice as wide as a graphics mode 8 dot, they would still need 16K of screen memory. Atari evidently figured that this was just too much memory (there were other more complicated factors as well). Anyway, that's the way things go.

The screen memory for modes 9 and 11 works, as you may have suspected, in exactly the same way as for the other color modes except that now there are four bits per dot instead of two. Well, not quite. Remember how I said before that there are five color registers? If that's true then, how do we pick the 16 colors we want to use for these two modes? As you probably know, the Atari computers give you a total of 256 colors to choose from altogether, made up of 16 hues (types of color) and 16 luminances (brightnesses). Sixteen times 16 gives us 256. Sixteen is also an awfully familiar number. By no small coincidence, graphics mode 9 lets us use all 16 luminances (but only one hue), and graphics mode 11 lets us use all 16 hues (but only one luminance). That way, in graphics mode 9 we only have to store the value of the hue we want in one of the color registers (color register four, of the background register) and then use screen memory to specify the luminance

of each dot. Similarly, in graphics mode 11 we store the luminance value in color register four and use screen memory to specify the hue of each dot.

Now we have a way to get 16 colors on the screen at the same time, but we're restricted to them all having either the same hue or the same luminance. Is there any way to get more than five colors on the screen without these restrictions? The answer is yes, and the solution is graphics mode 10. Before we look at this mode, however, let's take another look at the color registers. We saw earlier that there are five playfield color registers, with "playfield" being a fancy name for anything that's put on the screen using screen memory. So far we haven't seen how to put things on the screen in any other way, but there is something called "Player/Missile Graphics," or "PMG" for short, that provides an alternative. I won't go into detail on PMG here since there is a whole column devoted to the subject later, but for now you should be aware that there are four player color registers in addition to the playfield color registers, for a grand total of nine color registers in all. Graphics mode 10 lets you use all of these registers to get any nine colors on the screen at once.

You'd think that graphics mode 10 would take less screen memory than modes 9 and 11, right? After all, it only has nine colors per dot instead of 16, so it should take less bits per dot. Sounds good, but unfortunately things don't quite work that way. Four bits will hold a value between 0 and 15; we already know that. But three bits will only hold a value between 0 and 7. In order to get nine colors, we need to store a value between 0 and 8. Alas, that means we have to use four bits and just not use some of the possible values (9 through 15). And this in turn means that the dots in graphics mode 10 are going to be exactly the same size as those in modes 9 and 11. This time, however, screen memory will work in the same way as it did with the four color modes, except now all nine color registers can be accessed.

The color registers that don't have a SETCOLOR value beside them are the player color registers. You'll see how to set the color values for these registers in a future column on bit-mapped graphics.

We've now covered all the modes (Yipee!), and you should hopefully have a good grasp of what screen memory is and how it works for each of the 12 graphics modes. This knowledge will come in handy, since the concept of screen memory is one that will come up again and again.

When you want to talk Atari

XL/XE HARDWARE

INTERFACES

ICD	
P:R Connection	59.99
Printer Connection	41.99
Supra	
1150	38.99
1151 (1200 XL)	41.99
Xetec	
Graphix Interface	38.99
Atari	
850 Interface	109.00

COMPUTERS



Atari 130XE \$135

Atari	
65 XE	99.99

XL/XE ENHANCEMENTS

Axlon 32K Mem. Board (400/800) ..	19.99
Atari 80 Column Card	79.99

MODEMS

Atari	
SX212 300/1200 (ST)	89.99
835	19.99
XMM301	42.99
Anchor	
VM520 300/1200 ST Dir. Con.	119.00
Avatex	
1200 HC	89.99
2400	179.00
Supra	
2400 Baud XL, XE	169.00
2400 Baud ST	169.00
2400 Baud (no software)	149.00

MONITORS

Magnavox	
CM8505 14" Composite	199.00

ST HARDWARE



ATARI SM1224 RGB COLOR MONITOR \$329

Includes: 520 ST FM with 3 1/2" drive built-in, mouse, power supply and 1224 color monitor.
 1040 RGB/Color System 999.00
 1040 Monochrome System 819.00
 1040 Computer (no monitor) 679.00
 520ST FM RGB/Color System ... 789.00
 520ST FM Monochrome System ... Call (Monochrome monitor)
 SM124 Monochrome Monitor 169.00
 SM1224 Color Monitor 329.00

DRIVES

Atari	
XF551 Drive (XL/XE)	179.00
AA314 DS/DD Disk (ST)	209.00



Atari SHD204 20 Meg for ST \$579

I.B.	
5 1/4" ST Drive	209.00
Indus	
GTS 100 3 1/2" DS/DD (ST)	189.00
GT Drive (XL/XE)	179.00
GTS1000 5 1/4" DS/DD ST	199.00
Supra	
20 Meg Hard Drive (XL/XE)	679.00
20 Meg Hard Drive (ST)	559.00
30 Meg Hard Drive (ST)	689.00

PRINTERS

Atari	
1027 LQ XL/XE	129.00



Atari XM-M801 XL/XE Dot Matrix \$189

XM-M804 ST Dot Matrix	199.00
XDM 121 Letter Qlty. XL/XE ...	209.00

Brother

M-1109 100 cps Dot Matrix	189.00
M-1409 180 cps Dot Matrix	299.00
HR-20 22 cps Daisywheel	339.00

Citizen

120D 120 cps Dot Matrix	149.00
180D 180 cps Dot Matrix	179.00
Premier-35 35 cps Daisywheel ..	479.00

Epson

LX-800 150 cps, 80 col	179.00
Hi-80 4 pen plotter	269.00
FX-86E 240 cps, 80 col	Call
FX-286E 240 cps, 132 col	Call
LQ-500 180 cps, 24-wire	Call
LQ-850 330 cps, 80 col	Call
EX-800 300 cps, 80 col	Call

NEC

P2200 pinwriter 24-wire	369.00
P660 pinwriter 24-wire	459.00
P760 pinwriter 132 col	659.00

Okidata

Okimate 20 color printer	129.00
ML-182 + 120 cps, 80 column ..	229.00
ML-192 + 200 cps, 80 column ..	349.00
ML-193 + 200 cps, 132 col.	459.00

Panasonic

KX-P1080i 144 cps, 80 col	179.00
KX-P1091i 194 cps, 80 col	199.00

Star Micronics

NX-1000 140 cps, 80 column ...	179.00
NX-15 120 cps, 132 column ...	319.00

Toshiba

P321-SL 216 cps, 24-wire	499.00
--------------------------------	--------

**SELECT FROM
OVER 3000
PRODUCTS**

**WE SHIP 90%
OF ALL ORDERS
WITHIN 24 HOURS**

COMPUTER MAIL ORDER

.....you want to talk to us.

XL/XE SOFTWARE

Access	
Executive Disk	19.99
Accolade	
Hardball	19.99
Atari	
Atariwriter Plus	35.99
Filemanager	11.99
Music Painter	11.99



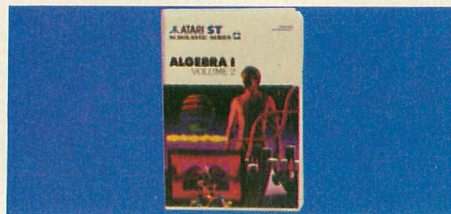
ACCESS Leaderboard Golf \$13⁹⁹

Atari Program Exchange	
Misc. Programs (cassettes)	at 1.99
Broderbund	
Graphics Library I, II, III	13.99
Printshop	25.99
Datasoft	
Alternate Reality (dungeon)	25.99
221 Baker St.	20.99
Electronic Arts	
Touchdown Football	12.99
Firebird	
Guild of Thieves	16.99
Silicon Dreams	16.99
Jewels of Darkness	16.99
Microprose	
Top Gunner	14.99
F-15 Eagle Strike	21.99
Silent Service	21.99
Origin Systems	
Ultima 4	36.99
Roklyn SPECIAL	
Anti-Sub/Journey to Planet ... (ea.)	3.99
Strategic Simulations	
Battalion Commander	23.99
Gemstone Warrior	10.99
Sublogic	
Flight Simulator II	31.99
Scenery CALIF	14.99
X-Lent	
Typesetter	22.99
Printshop Interface	21.99

ACCESSORIES

MD1-M SS/DD 5 1/4"	8.49
MD2-DM DS/DD 5 1/4"	8.99
MF-1DDM SS/DD 3 1/2"	11.99
MF2-DDM DS/DD 3 1/2"	18.49
Sony	
MD1D SS/DD 5 1/4"	6.99
MD2D DS/DD 5 1/4"	7.99
MFD-1DD SS/DD 3 1/2"	11.99
MFD-2DD DS/DD 3 1/2"	17.99
Allsop Disk Holders	
Disk File 60-5 1/4"	9.99
Disk File 30-3 1/2"	9.99
Curtis	
Emerald	39.99
Safe Strip	19.99
Universal Printer Stand	14.99
Tool Kit	22.99
ICD	
BBS Express (ST)	54.99
Sparta DOS Construction Set	28.99
US Doubler/Sparta DOS	47.99
Real Time Clock	48.99
Rambo XL	29.99
US Doubler	28.99

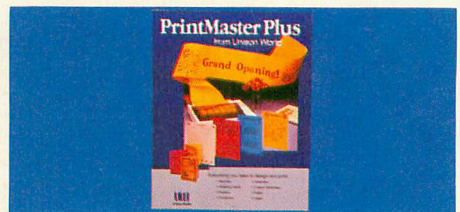
ST SOFTWARE



ATARI	
Geometry	
Vol. II	\$16⁹⁹
Abacus	
PC Board Designer	149.00
Access	
Leaderboard Golf	22.99
Activision	
Hacker II Doomsday	27.99
Antic	
CAD 3-D	31.99
Avant Garde	
PC Ditto	64.99
Batteries Included	
Degas Elite	38.99

ST SOFTWARE

Comnet	
ST Term	19.99
Dac	
Easy Accounting	59.99
Electronic Arts	
Gridiron Football/Auto Duel. (ea.)	26.99
Isgur Portfolio	119.00
Firebird	
Silicon Dreams	19.99
The Sentry	19.99
Infocom	
Beyond Zork	34.99
Michtron	
Major motion	25.99
Microprose	
Gunship	28.99
F-15 Strike Eagle	24.99
Miles Software	
ST Wars	24.99
Mark Williams	
C	119.00
Paradox	
Wanderer (3D)	24.99
Progressive Computer	
Graphic Artist 1.5	129.00
Psygnosis	
Barbarian/Deep Space	(ea.) 25.99
Soft Logik Corp.	
Publishing Partner	54.99
Strategic Simulations	
Rings of Zilfin/Phantasie III (ea.)	23.99
Sublogic	
Flight Simulator II	32.99
Timeworks	
Swiftcalc/Wordwriter	(ea.) 46.99
Partner ST	37.99



UNISON WORLD	
Printmaster Plus	\$25⁹⁹
Word Perfect Corp	
Word Perfect 4.1	189.00

In the U.S.A. and in Canada

Call toll-free: 1-800-233-8950

Outside the U.S.A. call 717-327-9575, Fax 717-327-1217

Educational, Governmental and Corporate Organizations call toll-free 1-800-221-4283
CMO, 101 Reighard Ave., Dept. B7, Williamsport, PA 17701

OVER 350,000 SATISFIED CUSTOMERS • ALL MAJOR CREDIT CARDS ACCEPTED • CREDIT CARDS ARE NOT CHARGED UNTIL WE SHIP

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

CIRCLE #108 ON READER SERVICE CARD.

B708

SHILOH

```
UNION RAITH-B INF 917 MEN 812  
DS N AMMO:4 HEL:4 FT:0 EF:70/70 712  
NORMAL OP:9 DIR:5 ADV:1 654  
RAVINE (1) X,Y:13,15 NO PLOT
```

Panak Strikes

by Steve Panak

As promised, this month we're going to take a look at some of the many simulations out on the market. And while the term simulation, at least in my mind, immediately places me on the battlefield, poised to relive some historic military confrontation, the true simulation genre is much broader, encompassing both warfare and peacetime activities, such as flight simulation and chess. Today, though, we will concentrate on the war game. But before we dive into battle, I have a little bad news.

Infocom, like some other manufacturers, has dropped Atari 8-bit support, citing memory constraints. While this is understandable in Infocom's case (as their text intensive works of interactive fiction eat up gobs of both RAM and disk space), the drop of support from other companies is less easily explained. The XE/XL lines have 64K, the same as the similar Commodore lines. Why then is support increasingly gravitating toward Commodore? Flippies (diskettes containing both Commodore and Atari versions, one on each side) have decreased, with games being produced for only one machine (usually the Commodore). This has to be a market phenomenon—there's just not enough demand out there to warrant the creation (or conversion) of an Atari version of many games. Otherwise, the manufacturers, being in this game for the money, would support our machines. While I don't know the underlying cause of this owner apathy, the solution is simple and straightforward.

Buy Atari hardware.
Buy Atari software.

And when someone you know is looking at computers, list the benefits of Atari. The Commodore and Atari lines are virtually identical, except, of course, for the ugly specter of limited software support. It's the old vicious circle, a computerized Catch-22. No one will buy the machines because there's no software available. And manufacturers avoid supporting the machines, fearful that they'll be stuck with excess inventory. But as I'm reporting this month, there really are a lot of games for the Atari. A lot of great games. And to make sure we continue to get more great games, we have to support the designers, programmers and software publishing companies that support us.

When you see a game you like, buy it, rather than simply borrow it from a fellow user. While sharing might provide you short-term gain, in the long run we'll all lose and possibly be stuck with unsupported machines. And when you see a game you think you'd like, but it's only available in a non-Atari format, write to the software publisher asking "why?" By casting these consumer votes, we are heard, and perhaps we'll get in on all the new games coming out. But if we're silent...well...I'd rather not even think about it. Enough of this pleading; let's get on to this month's games.

War simulations. These very specialized games are the easiest to buy and the easiest to review. This is because they are so similar. While each may feature a different battle in a different era, their basic programming structure is typically the same. Based on the old Avalon Hill board games which were just begging to be computerized, these games replace multi-sided dice and complex tables with the incredible speed of the computer. The resulting games are much easier to learn, much easier to play and much more enjoyable and accessible to the general public.

But whether the battle rages on a monitor, or on a table-top board, play progresses through a number of phases and stages in which you typically issue commands to your troops and then observe the results. Most games take into account troop strength and speed of movement, while others add additional depth, accounting for such variables as leader charisma and troop morale. Since most games operate just about the same, you'll want to choose your simulation based on the era you wish to examine. Will it be our country's Civil War, or a conflict on some futuristic planet? The choice is up to you.

If the number of titles a company is-

While I don't know the
underlying cause of this
owner apathy, the
solution is simple and
straightforward: Buy
Atari hardware. Buy
Atari software.

issues in a given genre were the sole indicator of its superiority, SSI would be the simulation king hands-down. Of course, by choosing Strategic Simulations as their name, they've really stuck their necks out. For if SSI's simulations aren't any good, then chances are pretty good that few players would be eager to try any of the many other titles in their massive catalog. And that would be a shame, since SSI simulations are typically the best available. Throughout the years SSI has refined their formula, cramming more and more action and realism into the scant 48K of code that their games occupy.

The mayhem started many titles ago, and the SSI catalog now stands over two dozen titles strong. And while the company has rendered a lot of their earlier titles obsolete, one of their earliest still stands out as one of their best. *NAM* is and was my favorite, perhaps because it came out so long ago, before it was acceptable to talk about the Vietnam War. And it was also a welcome change to the Civil War scenarios that make up the bulk of the simulation market. Best of all, *NAM* uses a joystick, a must for people like me whose backs rebel at the prospect of leaning over a keyboard for the one to four hours it takes to finish the game. Menus made the game easy to learn and sharp arcade-quality graphics kept play exciting. Simply put, *NAM* blew the competition away.

Since then, SSI has pumped out a string of Civil War simulations, starting with *Battle of Antietam*, followed by *Gettysburg: The Turning Point*. These revolution-

ary games crammed an incredible amount of realism into the sparse 48K of the 800, and allowed the game's complexity to grow along with the ability of the player. As you progress through the intermediate and advanced games, you'll find the play becoming more complex, more real and more demanding. Each successive game utilizing this system has reached new levels of realism and historical accuracy. Their latest, *Shiloh*, which we will examine in depth later, continues the evolution of SSI simulation.

But SSI is no longer alone in the simulation market, since Game Designer's Workshop arrived on the scene and became a force to be reckoned with. *The Battle of Chickamauga* is every bit as good as SSI's comparable *Rebel Charge at Chickamauga*, and better in that it allows the use of the joystick (my delicate back breathes a silent sigh of relief). Game Designer's Workshop makes games every bit as good as SSI, and their entry into the marketplace can only mean more and better simulations from both companies, as each tries to capture the limited simulation market. Thus, any of the recent simulations from either of these companies are worthy additions to the war gamer's library.

For those whose preferences lean toward conflicts of the future, I regret that there are few programs to appease your blood lust. About the only one that springs immediately to mind is *Ogre*, from Origin Systems. This futuristic game begins with the premise that a nearly invincible robot/tank creature exists, and that it is desirable to destroy it. Unfortunately, this is one tough *Ogre*, and you'll find it a worthy opponent on even the easiest difficulty settings. This challenging game features SF-like drop-down menus activated with the joystick and graphics which push the XE to its limit. I've heard a lot of complaints that it was too hard, but no one has dismissed the *Ogre* as a wimp.

Another space simulation is *Star Fleet*. I am at odds with a number of my computer brethren over this one because, although *Star Fleet* has its followers, I do not count myself among them. But, since it seems unlikely that so many people would be drawn to a really bad game, it can't be as bad as I think it is. In *Star Fleet* you search the galaxy for enemy encounters, using a number of offensive and defensive systems. Your goal is to progress through the *Star Fleet* ranks to Admiral Emeritus. Unfortunately, I never got past Ensign. It wasn't that I didn't know how to play, as great game design and superb documentation made play nearly effort-

less. I just never felt drawn into its world. Since *Star Fleet's* fervent followers feel otherwise, I am compelled by consensus to advise you to at least take a look at it.

Unfortunately, that's about all the futuristic simulations there are. Hopefully in the coming year, we'll see a few more. Personally, I'd like to see SSI port over the Roadwar series. These great games drop you in the middle of a world similar to that depicted in the *Mad Max* movie series. This premise, which lets you travel the globe in vehicular gangs, is not entirely original, but is nonetheless very engaging. And I notice that there is an Apple version running in 48K. How 'bout it, guys, give us an Atari 8-bit version? But until they respond to our pleas, I guess we'll just have to settle for this newest one.

SHILOH: Grant's Trial in the West by David Landrey and Chuck Kroegel

SSI
1046 N. Rengstorff Avenue
Mountain View, CA 94043
48K Disk \$39.95

In the newest simulation from SSI, the masters again use their patented formula to turn out a game that is not only historically accurate and complex, but also relatively easy to play. *Shiloh: Grant's Trial in the West* time warps us back to the 1800s, plunging us into the middle of our country's Civil War to experience the bloodiest battle our country had seen to date. Before the battle of Shiloh, General (later President) Grant had had little experience in the field. But that was to change on April 6, 1862, when his troops were surprised by the Confederate Army from Mississippi, under the command of General Johnston. A relatively inexperienced Grant, along with large numbers of untried troops, made for two of the most violent days in the war's history.

The program utilizes a refined version of the *Gettysburg* game system, which allows for three levels of play, the first letting the novice learn to play quickly using options, such as hiding enemy troops and elaborate handicapping of each side, keep play interesting for days to come. And it *will* take days, as each game is rated at 10-15 hours in length. Refinements of the gaming system include the ability to switch between strategic and tactical displays at any time, and special consideration for troops new to battle.

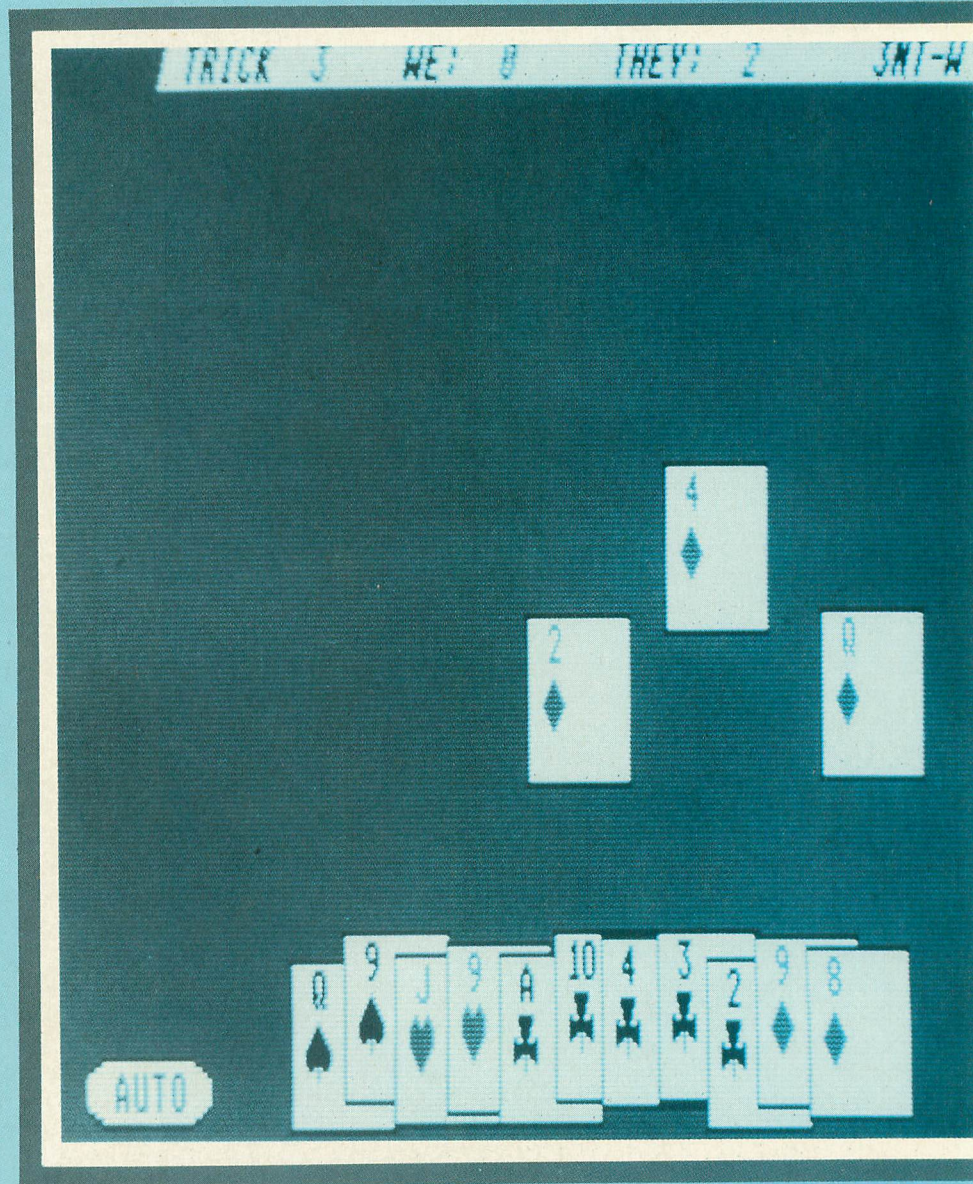
Play is simple and progresses in two phases, operation and combat. In the operation phase you give troops orders and check their condition, while the com-

bat phase lets you pause as the computer resolves the conflicts. Commands are easily given. You move an on-screen cursor about, reviewing the troops, giving orders to those that require them, pressing the space-bar to access the command menu. Thus you are always advised of a unit's status as well as the available commands.

A complete manual chock full of informative tables and charts helps you to quickly become familiar with the game and your troops, and a map provides a nice overview. Of particular interest is the

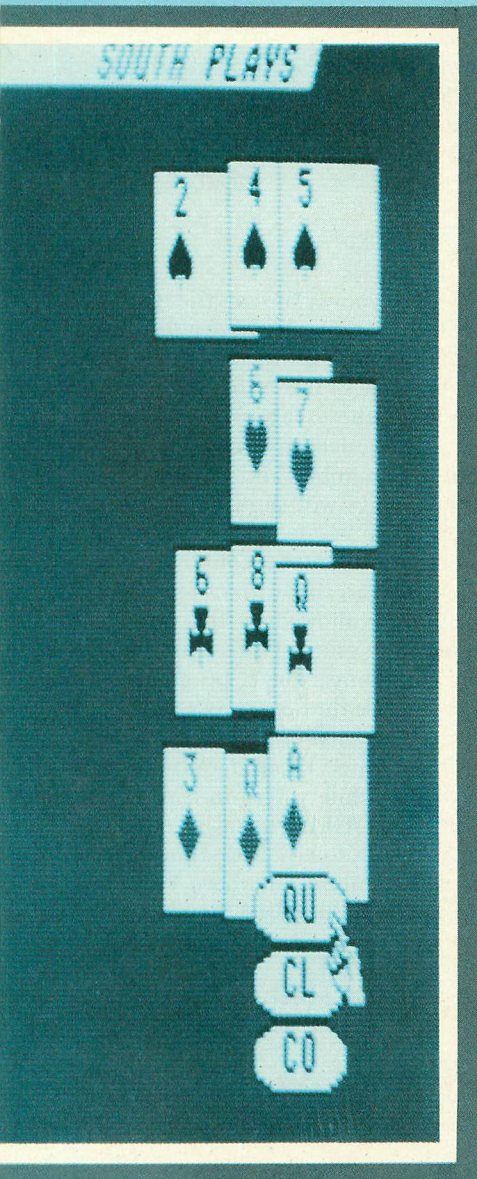
section which details the differences between *Shiloh* and other SSI games, allowing veteran gamers to quickly familiarize themselves with this complex program. All things considered, SSI has achieved another victory with *Shiloh*. While the program breaks no staggering new ground, it also offers no disappointing surprises.

BRIDGE 5.0
by Arthur M. Walsh
Artworx Software Company, Inc.
1844 Penfield Road



Penfield, NY 14526
48K Disk \$29.95

As I mentioned before, simulations don't always plunk you down in the middle of a raging battlefield. Indeed, some of the best simulations beat us at mankind's greatest strategy games. There are a number of chess games out there (*Chessmaster 2000* being the best), and a backgammon derivative (*Pegammon*). But while chess is one of mankind's most complex games, bridge claims enough devotees to make it a modern-day classic, and it contains enough complexity to



For those whose preferences lean toward conflicts of the future, I regret that there are few programs to appease your blood lust.

keep them interested. Since I am by no means an expert (or even a novice bridge player), I turned the program over to my mother, who has been addicted to bridge for years.

The program boots up nicely enough, although the graphic limitations of the 8-bit make the cards a little hard to decipher. Bidding, the first phase of play in bridge, is accomplished by typing your bid on the keyboard. The three other hands are bid by the computer. However, my expert found the computer's bids to be faulty at times. For instance, *Bridge 5.0* would not arrive at game bid when it was possible, and often would not open with a two bid when the card count merited it. It also did not always respond to a two opening bid or to a jump switch. Finally, it would occasionally rebid a four-card suit, which leads to an incorrect final bid. In addition, my young son displayed to me that pressing keys unexpected by the program caused it to terminate execution and confront us with a rude error code. While the START key got us going again, it did not reset our hand. Most annoying.

On the plus side, it does supply bridge junkies with 8-bits, an opponent who (presumably) doesn't cheat and never tires of play. Even my mother, who found some of its play faulty, nonetheless continues

to play. Especially appealing was the feature that allows the current hand to be replayed an unlimited number of times, letting you try different strategies, making the program a great learning tool. Unfortunately, the sparse manual tells little about the program and even less about bridge. You'll have to hit the library to learn how to play. Surprisingly, the program does not bid according to Goren or Culbertson, two popular systems, but instead utilizes the standard American bidding method. In addition it follows the Blackwood and Staymen conventions.

Other features allow you to save up to 66 hands, but unlike the ST version of the game, it does not let you set up your own hands. All in all, though, *Bridge 5.0* is an acceptable and affordable bridge program. While it might not provide the expert with expert play, it does take the place of human opponents when none are available, and it makes a good teaching tool.

That wraps it up for this month, so I'll just power down. But before I disconnect, I'll remind you that next time we'll move on to fantasy games, my favorite genre. I'll name the best D&D derivative, and warn you which ones aren't worth the media they're stored on. Until then, good gaming. **A**

When you see a game you think you'd like, but it's only available in a non-Atari format, write to the software publisher and ask "Why?"

an ak Strikes

DATABASE DELPHI

by Michael A. Banks



Things have been busier than usual in ANALOG's Atari SIG. The Forum has turned over almost 38,000 messages, making it the busiest Forum on Delphi. Message thread topics include everything from rumors to news and gossip. Stop in and put in your two cents' worth! Or, if you have a technical question, post it in the Forum; you're almost guaranteed to get an answer—the ANALOG Atari SIG membership represents one of the largest Atari resources anywhere.

New ST SIG

To unify the ST offerings and better organize the database, the ANALOG Atari SIG has been expanded to include a new ST SIG. The SIG is accessed by typing ST at the ANALOG> prompt (although, by the time you read this, the SIG may have gained its own entry on the main GROUPS menu). If you have an ST computer or if you are just interested in what's going on with the STs, be sure to drop by. Just as in the ANALOG SIG, the databases of the ST SIG are packed with useful and unique programs, reviews and informa-

tion files, and many more are added every week.

Weekly Conferences

The Atari ST SIG is now hosting a real-time conference each Tuesday to 10 p.m. EST. You'll find the conferences an excellent venue for sharing information about Atari computers, getting answers to questions and meeting new friends. Even though the conference is being held in the ST SIG, 8-bit computer owners are encouraged to attend.

If you haven't yet used Conference on Delphi, here's a quick guide. To get to

Conference, type CONFERENCE (or just CO at the Atari SIG main menu, as shown:

ANALOG's ATARI SIG Menu:

Announcements	Request Free Upload
Conference	Set Preferences
Databases	Topic Descriptions
Entry Log	Who's Here
Forum (Messages)	Workspace
MAIL (Electronic)	Help
Member Directory	Exit
Poll	

ANALOG> What do you want to do? CO

You'll be moved to the Conference Menu, which lists the major conference commands:

Welcome to the ANALOG's ATARI SIG Conference System

Conference Menu:

WHO (list groups)	PAGE a user
JOIN a group	NAME nickname
EXIT	
CONFERENCE>	

If you type WHO at the CONFERENCE> prompt, you'll see a list of who's currently online in the Atari SIG. Members who are in the conference area will have parentheses around their names. Any conference groups will also be listed—by number and name—and the group listings will show who is in each group.

You have to enter a conference group before you can chat with others. To do so, type JOIN, followed by the number or name of the group. Once you're in, type your comments; they'll be displayed to everyone else in the group, and you'll be able to read what everyone else types. (A number of special commands are available to you while you're in a conference group. Type /HELP for a list.)

If you wish to talk with another member who's online anywhere on Delphi, type PAGE, followed by the name of the

Delphi has two new offerings on its Library menu: Terra Nova and city/country information files.

member. If the member comes to the conference area and accepts your page, a conference group will be created for you.

To adopt a "handle" for use while you're in conference, type NAME followed by the handle you wish to use.

(All of these commands can be used while you are in a conference group, as well, but they must be preceded by a slash (/). For more information, type HELP at the CONFERENCE> prompt, or /HELP while you are in a conference group.)

The Atari SIG Conference area is a great place to meet and chat with other Atari users any time, by the way. (And, as you know if you've been in the Atari SIG, you'll find someone in the SIG most any time of day or night.)

International Additions


Delphi has two new offerings on its Library menu: Terra Nova and city/country information files.

Terra Nova is an international/foreign language area accommodating Spanish, Japanese, German, French, Esperanto and English (which is a foreign language to many Delphi members). Terra Nova databases and Forums contain information and HELP files in appropriate languages for each area. The area should be a welcome addition for foreign users—especially the large number of users who log in from Argentina's Delphi (called SISCOTEL), and the extremely active community of callers from Japan (who were formerly accommodated by the NIPPON group—a sub-SIG in the Micro Artists' Network). Anyone who is studying or otherwise has an interest in another language will also find this area useful.

The city/country information files are accessed through selections labeled "Metroline" and "Worldline." Each area prompts you for the name of a city or country, then provides a menu offering hotel, travel, cultural and other information on the section.

Collectibles Topic Added to Hobby Shop

"The Hobby Shop"—Delphi's SIG for model builders and other hobbyists—has added a "Collectibles" topic to their databases. This topic accommodates collectibles of all kinds—from stamps and coins to antiques.

In addition to numerous other books, Michael A. Banks is the author of DELPHI: The Official Guide and The Modem Book, both from Brady Books. You can write to him via E-mail on Delphi to membername KZIN. 

The Atari ST SIG is
now hosting a real-time
conference each
Tuesday to 10 p.m. EST.



Make the Delphi Connection!

As a reader of ANALOG Computing, you are entitled to take advantage of a special Delphi membership offer. For only \$19.95 (\$30 off the standard membership price!), you will receive a lifetime subscription 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 phone call. Make the Delphi connection by signing up today!

To join Delphi:

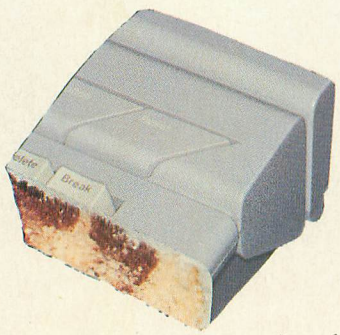
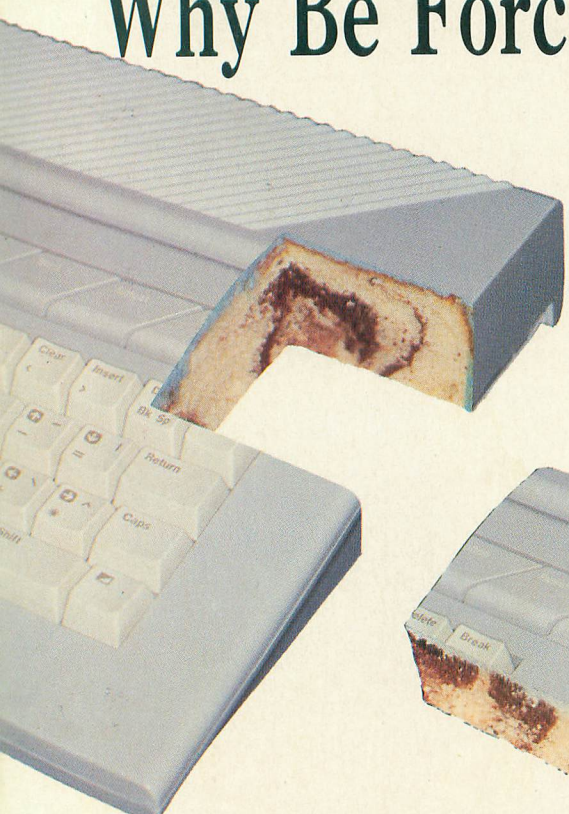
1. Dial 617-576-0862 with any terminal or PC and modem (at 2400 bps, dial 576-2981).
2. At the Username prompt, type JOIN-DELPHI.
3. At the Password prompt enter ANALOG.

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.

You Own an Atari

Why Be Forced to Read a Magazine that

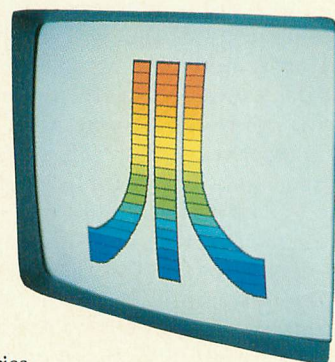


YOUR ATARI RESOURCE CENTER

ANALOG Computing continues to offer exciting products for you and your Atari Computer. And we're the only magazine for the Atari 8-bit computer line that hasn't allowed its content to be virtually taken over by coverage of the Atari ST. We include only a minimal amount of ST material so that you can stay informed of what's happening with the 8-bit computer's brother.

Whether you own a reliable ol' 400 or 800, a shiny XL, new XE or even an XE Game Machine... we offer usable utilities, entertaining educational software, dynamite disk programs and great graphics and games. In fact, our readers still use ANALOG programs that were published over five years ago!

So when software companies turn their heads to other computers, you can turn yours to the one that supports your 8-bit Atari. And that's ANALOG Computing.



ANALOG's Best! Over 88 of ANALOG Computing's best and most requested programs are now available on this series

of ten diskettes. The programs are all ready to run and come with complete documentation on the flip side of each floppy diskette. Select from Graphics, Educational, Utilities 1, Utilities 2, Disk Utilities and Games Disks 1, 2, 3, 4 and 5. Only \$9.95 each (plus \$1.50 shipping per order). Specify disk title when ordering.

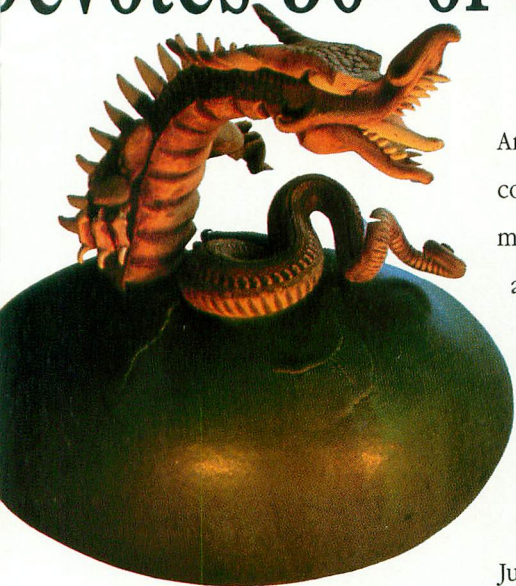
Unlock the secrets of your Atari Computer! This handy 16-page pocket reference card covers information you need when programming your 8-bit. Error codes, internal codes, PEEK & POKE locations, machine-language aids, graphic mode specs and BASIC commands with abbreviations are only some of the helpful items at your fingertips.

The ANALOG Computing Pocket Reference Card, only \$7.95 each!
(Plus \$1.50 shipping and handling.)



6502 Computer.

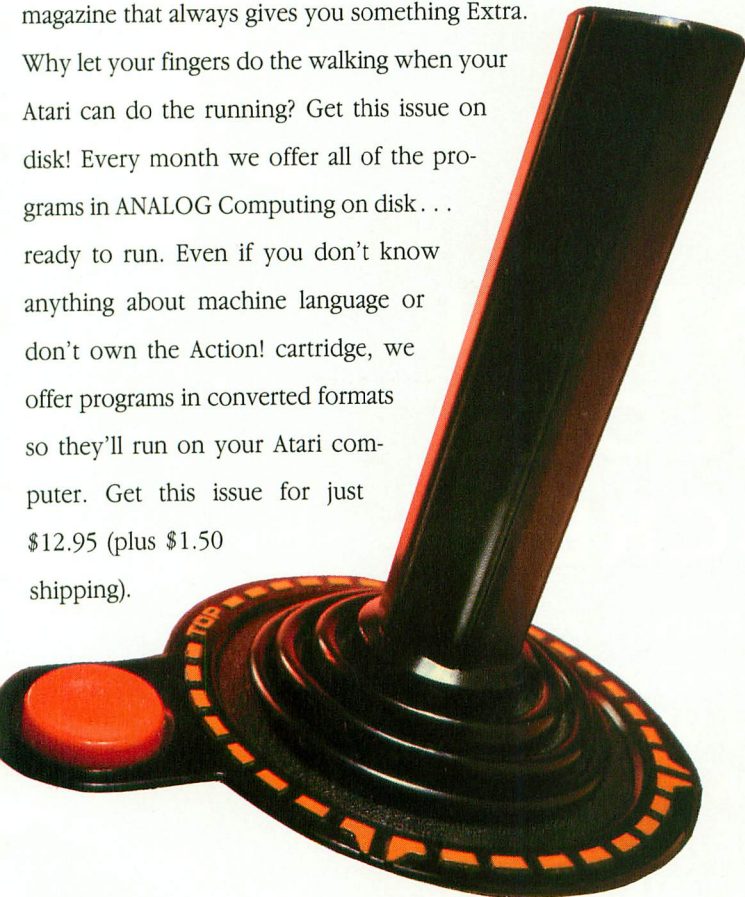
Devotes 50% of Its Pages to the Atari ST?



An Atari 8-bit Extra. While other "Atari" 8-bit magazines just make claims on how they cover your machine, we come through! Over 130 pages of new, never before published material. Programs like Easy Type, Dragon Chase, Pastels, Display List Mod, Tactics, Trivia and Create-a-base are all documented and ready to type in and run . . . all for just \$8.95! (Add \$1.50 for shipping.)

Get the Extra on disk! This special offer for Extra owners gets you all of the programs in an Atari 8-bit Extra on disk. Avoid typing errors, hours of tedious typing and frustration. Just plug in the disk and you are ready to roll! Two, ready-to-run double-sided floppies, \$24.95. (Disks only. Atari 8-bit Extra sold separately. Please add \$1.50 for shipping.) From the

magazine that always gives you something Extra. Why let your fingers do the walking when your Atari can do the running? Get this issue on disk! Every month we offer all of the programs in ANALOG Computing on disk. . . ready to run. Even if you don't know anything about machine language or don't own the Action! cartridge, we offer programs in converted formats so they'll run on your Atari computer. Get this issue for just \$12.95 (plus \$1.50 shipping).



ANALOG COMPUTING

ANALOG COMPUTING OFFICIAL ORDER FORM

Use this coupon to order the most complete up-to-date products specifically designed for your ATARI PC!

ANALOG'S BEST—Graphics Disk	\$ 9.95	\$ _____
ANALOG'S BEST—Educational Disk	\$ 9.95	\$ _____
ANALOG'S BEST—Utilities #1	\$ 9.95	\$ _____
ANALOG'S BEST—Utilities #2	\$ 9.95	\$ _____
ANALOG'S BEST—Disk Utilities	\$ 9.95	\$ _____
ANALOG'S BEST—Games #1	\$ 9.95	\$ _____
ANALOG'S BEST—Games #2	\$ 9.95	\$ _____
ANALOG'S BEST—Games #3	\$ 9.95	\$ _____
ANALOG'S BEST—Games #4	\$ 9.95	\$ _____
ANALOG'S BEST—Games #5	\$ 9.95	\$ _____
ANALOG COMPUTING—POCKET REFERENCE CARD	\$ 7.95	\$ _____
ANALOG COMPUTING—8-bit EXTRA	\$ 8.95	\$ _____
ANALOG COMPUTING—8-bit EXTRA (on disk)	\$24.95	\$ _____
ANALOG MAGAZINE ON DISK (please specify issue)	\$12.95	\$ _____
SHIPPING AND HANDLING—add \$1.50 for each product ordered	\$ _____	\$ _____
TOTAL ORDER		\$ _____

Payment Enclosed Charge My VISA Master Card

Card # _____ Exp. _____

Signature _____

Name _____

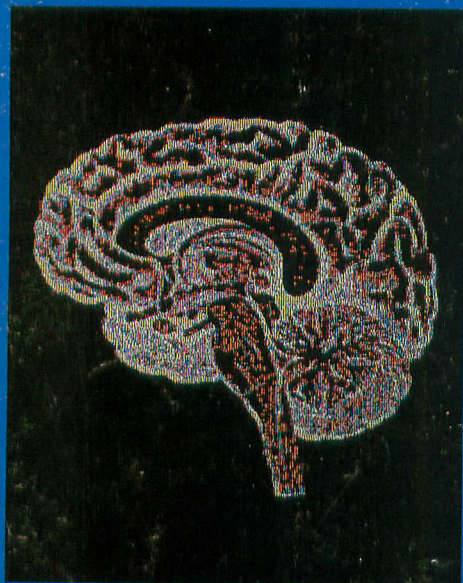
Address _____

City _____ State _____ Zip _____

Make checks payable to: LFP, Inc. P.O. Box 67068, Los Angeles, CA 90067.
Your order will arrive in 4 to 6 weeks — WATCH FOR IT! ZIHYY

California residents add 6.5% sales tax on all orders except back issues.

INSIDE THIS ISSUE:



ANIMATION WORDLOCK PRINTSCREEN

NEW FEATURES

MASTER MEMORY MAP
GAME DESIGN WORKSHOP

PLUS

DATABASE DELPHI
PANAK STRIKES!