NO. 40 $2.50 SEPTEMBER 1981

MICRO™

THE 6502/6809 JOURNAL

Double Apple Bonus featuring Graphics

Jumps and the 6502

Screen Print Utility for Atari 400/800

6809 Addressing Modes

Turn your Apple into the world's most versatile personal computer.

The SoftCard™ Solution. SoftCard turns your Apple into two computers. A Z-80 and a 6502. By adding a Z-80 microprocessor and CP/M to your Apple, SoftCard turns your Apple into a CP/M based machine. That means you can access the single largest body of microcomputer software in existence. Two computers in one. And, the advantages of both.

Plug and go. The SoftCard system starts with a Z-80 based circuit card. Just plug it into any slot (except 0) of your Apple. No modifications required. SoftCard supports most of your Apple peripherals, and, in 6502-mode, your Apple is still your Apple.

CP/M for your Apple. You get CP/M on disk with the SoftCard package. It's a powerful and simple-to-use operating system. It supports more software than any other microcomputer operating system. And that's the key to the versatility of the SoftCard/Apple.

BASIC included. A powerful tool, BASIC-80 is included in the SoftCard package. Running under CP/M, ANSI Standard BASIC-80 is the most powerful microcomputer BASIC available. It includes extensive disk I/O statements, error trapping, integer variables, 16-digit precision, extensive EDIT commands and string functions, high and low-res Apple graphics, PRINT USING, CHAIN and COMMON, plus many additional commands. And, it's a BASIC you can compile with Microsoft's BASIC Compiler.

More languages. With SoftCard and CP/M, you can add Microsoft's ANSI Standard COBOL, and FORTRAN, or Basic Compiler and Assembly Language Development System. All, more powerful tools for your Apple.

Seeing is believing. See the SoftCard in operation at your Microsoft or Apple dealer. We think you'll agree that the SoftCard turns your Apple into the world's most versatile personal computer.

Complete information? It's at your dealer's now. Or, we'll send it to you and include a dealer list. Write us. Call us. Or, circle the reader service card number below.

SoftCard is a trademark of Microsoft. Apple II and Apple II Plus are registered trademarks of Apple Computer. Z-80 is a registered trademark of Zilog, Inc. CP/M is a registered trademark of Digital Research, Inc.

MICROSOFT
CONSUMER PRODUCTS

Microsoft Consumer Products, 400 108th Ave. N.E.,
Bellevue, WA 98004. (206) 454-1315

THE NATIONAL COMPUTER SHOWS

HAVE WE GOT A PROGRAM FOR YOU IN '81 & '82

Attend the biggest public computer shows in the country. Each show has 100,000 square feet of display space featuring over 50 Million Dollars worth of software and hardware for business, industry, government, education, home and personal use.

You'll see computers costing $150 to $250,000 including mini and micro computers, software, graphics, data and word processing equipment, telecommunications, office machines, electronic typewriters, peripheral equipment, supplies and computer services.

All the major names are there including; IBM, Wang, DEC, Xerox, Burroughs, Data General, Qantel, Nixdorf, NEC, Radio Shack, Heathkit, Apple, RCA, Vector Graphic, and Commodore Pet. Plus, computerized video games, robots, computer art, electronic gadgetry, and computer music to entertain, enthrall and educate kids, spouses and people who don't know a program from a memory disk.

Don't miss the Coming Of The New Computers – Show Up For The Show that mixes business with pleasure. Admission is $5 for adults and $2 for children under 12 when accompanied by an adult.

Ticket Information

Send $5 per person with the name of the show you will attend to National Computer Shows, 824 Boylston Street, Chestnut Hill, Mass. 02167. Tel. 617 739 2000. Tickets can also be purchased at the show.

THE MID-WEST COMPUTER SHOW

CHICAGO
McCormick Place
SCHOESSLING HALL
23rd & THE LAKE
THURS-SUN
SEPT 10-13, 1981
11AM TO 7PM WEEKDAYS
11AM TO 6PM WEEKENDS

THE MID-ATLANTIC COMPUTER SHOW

WASHINGTON, DC
DC Armory/Starplex
2001 E. CAPITAL ST. SE
(E CAP. ST. EXIT OFF I 295 -KENILWORTH FRWY)
ACROSS FROM RFK STADIUM
THURS-SUN
SEPT 24-27, 1981
11AM TO 7PM WEEKDAYS
11AM TO 6PM WEEKENDS

THE NORTHEAST COMPUTER SHOW

BOSTON
Hynes Auditorium
PRUDENTIAL CENTER
THURS-SUN
OCT 15-18, 1981
11AM TO 7PM WEEKDAYS
11AM TO 6PM WEEKENDS

THE SOUTHEAST COMPUTER SHOW

ATLANTA
Atlanta Civic Center
395 PIEDMONT AVE NE AT
RALPH McGILL BLVD
THURS-SUN
OCT 29-NOV 1, 1981
11AM TO 7PM WEEKDAYS
11AM TO 6PM WEEKENDS

THE SOUTHERN CALIFORNIA COMPUTER SHOW

LOS ANGELES
LA Convention Center
1201 SOUTH FIGUEROA
THURS-SUN
MAY 6-9, 1982
11AM TO 7PM WEEKDAYS
11AM TO 6PM WEEKENDS

Losing Control?

⟨ESC⟩ape

With Products From Sirius Software

Gorgon: The earth has entered a time warp . . . and the battle has just begun.

Orbitron: Action in orbit and attacks from all sides.

Gamma Goblins: Avoid the hyperdermics while transporting blood to Hemophilia.

Both Barrels: A two-game pack including "Duck Hunt" and "High Noon."

Star Cruiser: Save yourself from the swooping aliens.

Pulsar II: Pulsar and Wormwall in a unique combination.

Phantoms Five: Real-time 3-D fighter-bomber challenge.

E-Z Draw: An easy, powerful, and versatile Apple graphics editor, includes the Higher Text generator by Ron and Darrel Aldrich.

Cyber Strike: Challenging galactic excitement for the real arcade gamer.

Autobahn: Hair raising excitement at 200 kilometers per hour.

Epoch: Explosive fast action with new 3-D effects. Complete with high speed docking and a time warp that is sure to give you vertigo.

Pascal Graphics Editor: Easy animation and powerful graphics editing has finally arrived.

Sneakers: Unending variety and fun as you meet the Sneakers, Cyclops, Saucers, Fangs, H-Wings, Meteors, Scrambles, and Scrubs.

Space Eggs: Hatch some fun with the spiders, lips, wolves, and Fuzzballs.

Apple II and Apple II + are registered trademarks of Apple Computer, Inc. All games mentioned in this advertisement are copyrighted products of Sirius Software, Inc. All rights reserved.

Contact your local computer dealer for more information. Dealer inquiries invited.

Sirius Software, Inc.

2011 Arden Way #2 Sacramento, CA 95825 (916) 920-1939

MICRO™
THE 6502/6809 JOURNAL

STAFF

Editor/Publisher
ROBERT M. TRIPP

Associate Publisher
RICHARD RETTIG

Associate Editors
MARY ANN CURTIS
FORD CAVALLARI

Special Projects Editor
MARJORIE MORSE

Production Coordinator
PAULA M. KRAMER

Typesetting
EMMALYN H. BENTLEY

Advertising Manager
CATHI BLAND

Circulation Manager
CAROL A. STARK

Dealer Orders
LINDA HENSDILL

MICRO Specialists
APPLE: FORD CAVALLARI
PET: LOREN WRIGHT
OSI: PAUL GEFFEN

Comptroller
DONNA M. TRIPP

Bookkeeper
KAY COLLINS

Sales Representative
KEVIN B. RUSHALKO
603/547-2970

DEPARTMENTS

5 Editorial
33 PET Vet
45 New Publications
93 Microbes
97 Hardware Catalog
101 Software Catalog
106 6502 Bibliography
111 Advertisers' Index
112 Next Month in MICRO

ARTICLES

8 **Jumps and the 6502** ..... *Mark Bernstein*
Methods of jumping and branching are discussed

15 **The Disk Switch** ..... *Nelson E. Ingersoll*
Add life to diskettes and disk drives, and save energy

20 **It's Time to Stop Dreaming, Part 4** ..... *Robert M. Tripp*
Addressing modes of the 6502 and 6809 are compared, new ones discussed

27 **Journal It: Screen Print Utility for Atari** ..... *John Elliott*
This utility will capture screen text and then journal it on a printer

37 **ASMLST: Full-sized Assembler Listing for AIM** ..... *Joel Swank*
Twenty-column listings are reformatted into full size

83 **MICROCRUNCH: An Ultra-fast Arithmetic Computing System, Part 2** ..... *John E. Hart*
A discussion of software support for the fast mathematics hardware outline in Part 1

DOUBLE APPLE BONUS

49 **Apple Graphics** ..... *Ford Cavallari*
A quick overview of what's offered in our Apple Graphics Section

50 **SHAPER: A Utility Program for Managing Shape Tables** ..... *Clement D. Osborne*
Build shape tables by vector input

59 **List Controller** ..... *Preston R. Black*
This program provides flexibility and ease of operation when using the LIST command

62 **Lo-Res Graphics and Pascal** ..... *C. Donald Heth*
Plot low resolution graphics using Pascal

66 **Dollars and Sense Revisited** ..... *David T. Delli Quadri*
Learn to print format in Applesoft BASIC using a MID$ statement

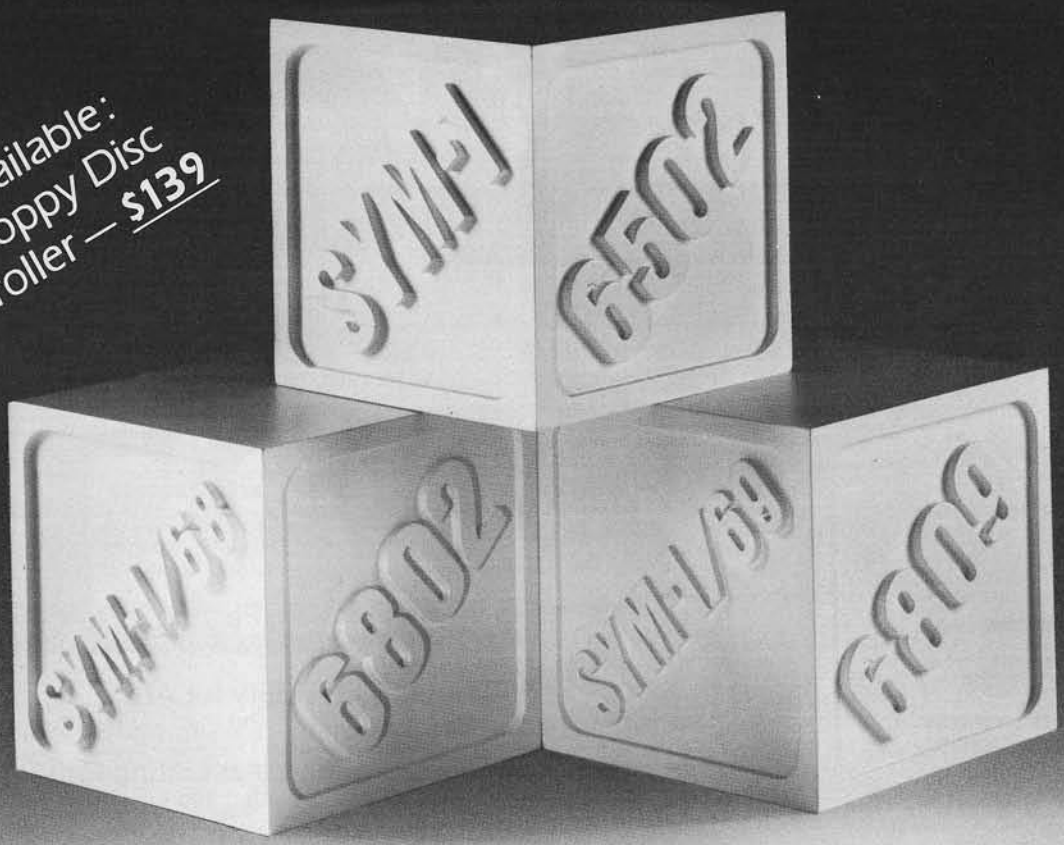
68 **Paddle Hi-Res Graphics** ..... *Kim G. Woodward*
Draw background scenes using paddles

71 **True 3-D Images on Apple II** ..... *Art Radcliffe*
Create stereo-pair images without accessory devices

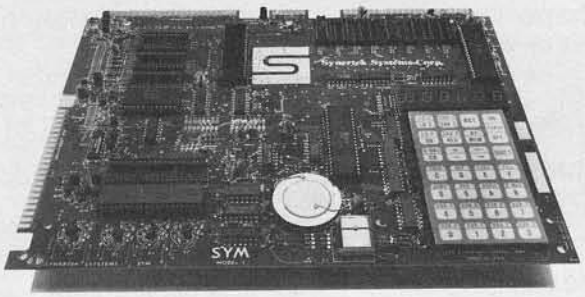
75 **Apple Bits, Part 1** ..... *Richard C. Vile, Jr.*
Aids for faster, more efficient low resolution graphics programming

79 **Tracemark, An Apple II Debugging Aid** ..... *Raymond Weisling*
Understand complex problems with this study-tool utility

Now available:
SYM Floppy Disc
Controller — $139



'Universatility.' It's as easy as 1. 2. 3.



'Universatility' can be found in three versions of Synertek Systems' SYM single board computer — the versatile, universal evaluation board.

Over 20,000 SYM-1 boards have been used for learning about and evaluating 6502 microprocessors for specific applications. OEM SYM boards are used in hundreds of products.

Now Synertek Systems presents the new SYM-1/68 for 6802 microprocessors, and the SYM-1/69 for 6809 microprocessors. These boards are designed to reliably perform the same functions as the SYM-1 board for these popular microprocessors. Each SYM board is complete and ready-to-use with its own version of the 4K byte ROM SUPERMON monitor firmware.

Modification kits are also available to quickly and easily convert existing SYM-1 boards to SYM-1/68 or SYM-1/69 microcomputers.

Build on your microprocessor knowledge with the 'universatility' of SYM microcomputer boards from Synertek Systems.

Every SYM-1, SYM-1/68, and SYM-1/69 single board computer features:

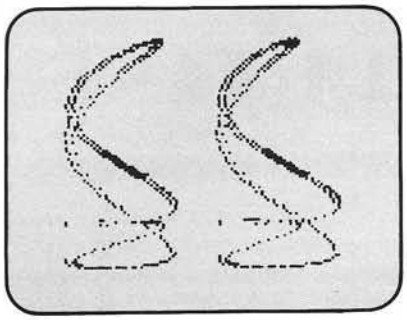
- 28 double-function keypad with audio response
- 4K byte ROM resident SUPERMON with over 30 standard monitor functions and expandable for individual requirements
- Up to 24K bytes of available program memory
- Expansion to 4K bytes of on-board static RAM
- Expansion to 71 Input/Output lines
- Single +5V power supply required
- Standard interfaces for audio cassette with remote control, 185 bytes/second cassette format, TTY and RS-232-C, system expansion bus, four I/O buffers, and oscilloscope single-line display

Synertek Systems

P.O. Box 552 Santa Clara, CA 95052
Telephone: (408) 988-5689
TWX: 910-338-0135

Dealer inquiries invited.

About the Cover



Your Own Roller Coaster!

What is more appropriate this time of year than the midway at a state fair or a trip to the local amusement park? The graphic on this month's cover is probably not immediately recognizable. But then, it is our first three dimensional live action graphic! As the photo suggests, the graphic is of a roller coaster. One of the bonus sections in this month's issue is dedicated to Apple High Resolution Graphics. One article covers the unusual topic of three dimensional moving images. As an example, it shows a roller coaster which you can see moving around its 3D track. So, "Wheeeee...."

Graphic from "True 3-D Images on Apple II" by Art Radcliffe

Photo by Robert Tripp

Thanks to Computer Mart of New Hampshire for producing the graphic on their equipment.

MICRO is published monthly by:
MICRO INK, Inc., Chelmsford, MA 01824
Second Class postage paid at:
Chelmsford, MA 01824 and Avon, MA 02322
USPS Publication Number: 483470
ISSN: 0271-9002

Send subscriptions, change of address, USPS Form 3579, requests for back issues and all other fulfillment questions to

MICRO
P.O. Box 6502
Chelmsford, MA 01824

or call
617/256-5515

Subscription rates	*Per Year*
U.S.	$18.00
Foreign surface mail	$21.00
Air mail:	
Europe	$36.00
Mexico, Central America	$39.00
Middle East, North Africa	$42.00
South America, Central Africa	$51.00
South Africa, Far East, Australasia	$60.00

Copyright© 1981 by MICRO INK, Inc.
All Rights Reserved

MICRO Editorial

Reader Feedback

Many thanks to the readers who took the time to fill out the questionnaire we included in our July issue. We have completed a preliminary analysis and some of the results were very surprising. For instance, we knew that the Apple was the most popular system among our readers, but we didn't suspect that the OSI portion was nearly as high! The proportion of readers using OSI systems at home is approximately equal to that for Apple systems (35.4% *vs.* 35.0%), while the Apple systems are used more at work (18.6% *vs.* 5.8%). The AIM and KIM percentages also came out a little higher than we expected.

Another significant statistic is that more than a quarter of our readers have access to more than one system. Because of this, the percentages below add up to well over 100%.

AIM	13.3%
Apple	50.0%
Atari	6.6%
KIM	14.6%
OSI	39.0%
PET/CBM	12.4%
SYM	6.6%
6809	4.4%
Other	2.7%

Sixty-three percent of our readers have had some formal training in computers, whether it be a masters in computer science or a night course in beginning BASIC. The average hardware level is somewhere between novice and intermediate (43% novice, 44% intermediate, and 12% expert), while the average software level is distinctly intermediate (19% novice, 62% intermediate, and 19% expert).

Readers were asked to rate (with 1 the best and 5 the worst) the various departments that appear in MICRO. Specialized departments such as Micros in Medicine, PET Vet, and the OSI columns, have understandably lower ratings because of the limited number of readers they serve.

Software Catalog	2.3
Hardware Catalog	2.4
Editorial	2.5
Letters	2.5
New Publications	2.6
Bibliography	2.8
OSI columns	3.2
Clubs	3.8
Micros in Medicine	3.9
PET Vet	5.1

Articles overall received higher ratings than departments. The biggest surprise was the lack of interest in Pascal. Specific hardware and software articles received high ratings, but often notes were added: "1 if it's for my system, 5 otherwise."

Specific software	1.8
Specific hardware	2.0
Programming techniques	2.0
BASIC	2.1
Applications	2.1
Assembly language	2.2
Tutorial	2.3
General software	2.4
General hardware	2.7
Pascal	4.0

These results are based on only 225 questionnaires. We have twice as many more to process, and will be doing more analysis to help guide our advertising, promotion, and editorial efforts. We'll keep you updated on new results.

Many readers made specific suggestions and comments. While it is impossible to respond to all of these, you can be assured they all will be read and considered! Thanks again!

Canadian Subscribers:

We hope you've received your August issue by the time you read this. If not, don't despair — it is on the way. Here's the reason for the delay: the post offices here in the U.S. would not accept Canadian mail because of the recent strike and subsequent backlog. However, we will be sending the issues August 17, and they should be in your hands soon!

Letterbox

You'll notice that our letterbox section rated high among readers answering the questionnaire. However, if you check our table of contents this month, you won't find it listed. Unfortunately, we haven't been getting enough general letters to present Letterbox regularly. We ask that you send us your comments, gripes, or thoughts, so we can rejuvenate this popular department.

Jon W Wright

HUNTINGTON COMPUTING

PROGRAMS LISTED BELOW ARE ON DISK:

HI-RES Soccer	$29.95 now	**$25.39**
Informer II	$49.95 now	**$42.99**
Wurst of Huntington Computing		**$19.99**
BPI		**15% OFF LIST**
Apple-Oids	$29.95 now	**$25.39**
Epson M-80 w/card & cable		**$575.00**
Autobahn	$29.95 now	**$25.39**
Mt. Comp Multi-Function Card		**15% OFF LIST**
Gobbler	$24.95 now	**$21.19**
On-Line Compiler		**15% OFF LIST**
VU#3 (Revised)		**15% OFF LIST**
Battler Cruiser Action	$39.95 now	**$33.99**
Ultima	$39.95 now	**$33.99**
Gorgon	$39.95 now	**$33.99**
Super Stellar Trek	$39.95 now	**$33.99**
Word Star	$375.00 now	**$289.00**
Mail Merge	$125.00 now	**$106.19**
Super Sort	$200.00 now	**$169.99**
Sentence Diagramming	$19.95 now	**$16.99**
Black Beauty	$34.95 now	**$29.69**
Moby Dick	$34.95 now	**$29.69**
Dymarc Surge Suppressor		**$53.99**
Milliken Math		**10% OFF LIST**
Gamma Goblins	$29.95 now	**$25.39**
DB Master	$229.00 now	**$194.59**
The Data Factory	$150.00 now	**$124.00**
Mission Asteroid	$19.95 now	**$17.99**
Wizardry	$39.95 now	**$33.99**
Bright Pen	$49.95 now	**$42.49**
Star Mines	$29.95 now	**$25.39**
PFS (Personal Filing System)	$95.00 now	**$80.69**
PFS: Report	$95.00 now	**$80.69**
Warp Factor	$39.95 now	**$33.99**
Microsoft Adventure	$29.95 now	**$26.09**
Compu-Math: Arithmetic	$49.95 now	**$44.95**
Modifiable Database II	$150.00 now	**$127.49**
TG Game Paddles	$39.95 now	**$33.99**
TG Joy Stick	$59.95 now	**$50.99**
The Wizard & The Princess	$32.95 now	**$28.99**
Flight Simulator (disk)	$34.95 now	**$29.50**
Odyssey	$29.95 now	**$25.50**
Sargon II	$34.95 now	**$29.70**
Program Line Editor	$40.00 now	**$34.99**
Space Eggs	$29.95 now	**$25.39**
Videx 80-Col. Board	$350.00 now	**$299.00**
3D Super Graphics	$39.95 now	**$35.99**
Compu-Math I or II	$40.00 now	**$34** ea.
HI-RES Cribbage	$24.95 now	**$21.99**
Lords of Karma (cassette)	$20.00 now	**$16.99**
Apple PIE & Formatter (Reg. $129.95)	Special	**$99.99**
The Book of Software	$19.95 now	**$17.99**
Versa Writer Expansion Pac-I	$39.95 now	**$33.99**
Apple II Users Guide	$14.95 now	**$12.99**
Oh Shoot!		**$19.99**
Serendipity Statistics or Gradebook	$169.00 now	**$143.59**
ABM (Muse)	$24.95 now	**$22.49**
Computer Conflict	$39.95 now	**$35.99**
Computer Air Combat	$59.95 now	**$52.99**
The Temple of Apshai	$39.95 now	**$33.95**
Super-Text II	$150.00 now	**$128.99**
Request	$225.00 now	**$191.19**
Thinker	$495.00 now	**$420.69**
Super Kram	$175.00 now	**$148.69**

Zork	$39.95 now	**$33.99**
Howardsoft Real Estate Analyzer	$150.00 now	**$129.99**
Super Disk Copy-Sensible	$30.00 now	**$25.49**
The Landlord	$795 now	**$649.99**
Nibble Express	$12.95 now	**$11.99**
All Nibble Software		**15% OFF LIST**
All Creative Computing		**15% OFF LIST**
All Programma		**15% OFF LIST**
Robot Wars		**15% OFF LIST**
Cranston Mannor		**15% OFF LIST**
Hayden AS Compiler	$200.00 now	**$169.99**
Beneath Apple DOS Book	$20.00 now	**$17.99**
Sublogic Animation Pack		**15% off list**
Memorex Disks		**10 for $24.99**
Pascal Programmer	$125.00 now	**$106.19**
NEC 12" Green/Black	$260 now	**$219.00**
VERBATIM DISKS (Datalife w/hub rings - unmarked) Ten for		**$27.99**
Spanish Hangman	$29.95 now	**$26.99**
Dragon's Eye	$24.95 now	**$21.19**
Computer Acquire	$20.00 now	**$16.99**
Twala's Last Redoubt	$29.95 now	**$25.39**
Snoggle	$24.95 now	**$21.19**
D.C. Hayes Modem	$379.00 now	**$299.00**
Data Capture 4.0	$60.00 now	**$52.99**
"Clock" Time Teaching Program	$29.95 now	**$26.99**
Alien Rain	$24.95 now	**$21.19**
Alien Typhoon	$24.95 now	**$21.19**
Raster Blaster	$29.95 now	**$25.39**
Payroll	$395.00 now	**$335.69**
Home Money Minder	$34.95 now	**$29.69**
3-D Skiing	$24.95 now	**$21.19**
Dr. Daley's Software Library	$79.95 now	**$69.99**
Physics I (Educ. Courseware)	$24.00 now	**$21.49**
J&S Computer Chemistry	$150.00 now	**$139.99**
Cook's Touch Typing	$14.95 now	**$13.49**
EAI Literal Comprehension (grades 2-4)	$39.95 now	**$35.99**
COMPAK MATH Grade by Concept	$495.00 now	**$445.49**
Hellfire Warrior	$39.95 now	**$33.99**
Mimco Joy Stick	$60.00 now	**$54.99**
Paper Tiger 560G	$1795.00 now	**$1499**
Mt. Com Music System	$545.00 now	**$479.00**
Statistics (Edu-ware)	$29.95 now	**$25.39**
Algebra I (Edu-ware)	$39.95 now	**$33.99**
Apple Crate	$59.95 now	**$54.89**
Creature Venture	$24.95 now	**$21.19**
Galaxy Space War I	$39.95 now	**$33.99**
DOS BOSS	$24.00 now	**$20.39**
Apple Barrel	$29.95 now	**$25.39**
Apple Roots	$39.95 now	**$33.99**
Interlude	$19.95 now	**$16.99**
ALL INTERACTIVE MICROWARE		**15% off list**
Meteorites in Space	$19.95 now	**$16.99**
Letter Perfect	$149.95 now	**$127.39**
VisiCalc 3.3	$199.95 now	**$169.95**
Visiplot	$179.95 now	**$152.89**
Visitrend	$259.95 now	**$220.89**
Visidex	$199.95 now	**$169.95**
Visiterm	$149.95 now	**$127.39**
Microsoft Typing	$19.95 now	**$16.99**
ASC II Express	$99.95 now	**$84.99**
Easy Writer Professional	$250.00 now	**$219.00**
ALL AVANT GARDE		**15% off list**
DRAGON FIRE		**15% off list**
Ed. Courseware Basic Tutor 1-6		**$28.79 ea.**

Special Birthday Sale *For Two-Year-Old Computer Users*

To celebrate our daughter Melody's second birthday, we are offering one of the best games we've seen anywhere at a special price. **DYNACOMP,** maker of exceptionally fine computer software, makes a game called **HODGE PODGE** for children ranging from under two years old to primary grades. Children learn the letters of the alphabet, words, numbers, musical scales, songs and animals. The child presses any key on the keyboard and something happens. For example, when "D" is pressed, a dog appears who frowns when a bathtub comes into sight – all to the tune of "Oh Where Has My Little Dog Gone." Our Melody loves it and jumps up and down in excitement when she plays it. It's a chance for her to finally get at the computer after watching Mommy and Daddy use it all day. It's the only program we know of for children that young. It runs on the Apple II* plus (Applesoft*) and needs 48K plus a disk drive. It lists for $23.95 but if you say "Happy Birthday Melody" you can have it for $18.99. We know you'll like it. We think it's fantastic.

HUNTINGTON COMPUTING
Post Office Box 787
Corcoran, California 93212

Order by Phone 800-344-4111
In California (209) 992-5411

SUPER DISCOUNTS

apple® SOFTWARE

VISA

master charge

We take MasterCard or VISA (Include card # and expiration date). California residents add 6% tax. Include $2.00 for postage. Foreign and hardware extra. Send for free catalog. Prices subject to change.

The ultimate APPLE® copy program
COPY II PLUS
$39.95

VERSATILE — Copy II Plus copies multiple formats — DOS 3.2, 3.3, PASCAL, FORTRAN, and CPM.

FAST — Copy II Plus copies diskettes in less than 45 seconds. That's faster than most other copy programs. Written entirely in ultra fast assembly language.

Search no more for that truly versatile, fast copy program. Copy II Plus is the most advanced copy program available for the Apple II Computer. Compare capability, compare speed, compare price, then call or write to order Copy II Plus. Requires Apple II with 48K and at least one Disk Drive.

CENTRAL POINT Software, Inc.

P.O. Box 3563
Central Point, OR 97502
(503) 773-1970

VISA master charge or check

Deliveries from stock. No C.O.D.'s
Apple is a registered trademark of Apple Computer, Inc.

Terrapin Turtle

Be one of the first persons to own your own robot. It's fun, and unlike other pets, the Turtle obeys your commands. It moves, draws, blinks, beeps, has a sense of touch, and doesn't need to be housebroken. You and your Turtle can draw pictures, navigate mazes, push objects, map rooms, and much, much more. The Turtle's activities are limited only by your imagination, providing a challenge for users of all ages. Interfaces, including software for easy control of the Turtle, are available for the Apple, Atari, and S-100 bus computers.

Terrapin will give a free Turtle to the person or persons who develop the best program for the Turtle by March 31, 1982. In addition, Terrapin will pay royalties. For more information, write or call;

Terrapin, Inc.
678 Massachusetts Avenue
Cambridge, MA 02139
(617) 492-8816

Books available from Terrapin

Turtle Geometry by Abelson and diSessa
An innovative book using Turtle Graphics to explore geometry, motion, symmetry and topology. MIT Press $20.00

Mindstorms by Seymour Papert
An exciting book about children, computers, and learning. Explains the philosophy of the new LOGO language. Basic Books $12.95

Artificial Intelligence by Patrick Winston
Explores several issues including analysis of vision and language. An introduction to the LISP language is incorporated in the second section. Addison-Wesley $18.95

Katie and the Computer by Fred D'Ignazio
A children's picture book adventure about a young girl's imaginary trip inside a computer. Creative Computing $6.95

Small Computers by Fred D'Ignazio
A book about the future of small computers and robots, aimed at adolescents. Franklin Watts $9.95

The Perfect Pet™

Jumps and the 6502

The 6502 microprocessor provides a variety of methods for jumping and branching. Several of these methods are not obvious, and can be very useful to the assembly language programmer. A little-known bug in the JMP INDIRECT instruction is also discussed.

Mark Bernstein
12 Oxford Street
Cambridge, Massachusetts 02138

Personal computers are powerful tools for program development, and many programmers have used them to discover the pleasures of machine and assembly language programming. Since current microprocessors are relatively slow, and the available languages (especially BASIC) are even slower, many other computerists find themselves forced to use assembly language routines simply to get the job done. Fortunately, personal computers are powerful tools for developing assembly language routines.

Since the personal computer is completely dedicated to its user, developing machine language routines is simple and often pleasant. Load a monitor or assembler, type in a section of code, and run it! If the program is correct, the results will be as expected. If the program is wrong, testing will reveal incorrect or unexpected results. Looking at the program listing should uncover the defect.

However, certain instructions are fraught with special difficulties. Whenever an instruction transfers control from one segment of program code to another, the instructions responsible must be *perfect* the first time. If an LDA or SBC instruction is misplaced or operates on the wrong address, the consequence will probably be a wrong answer. But if a JMP instruction tells the computer to execute nonsense, the machine will probably require a complete reset, hours of typing may be lost, and (worst of all) no record of the run will be preserved to help the programmer.

Not only are control transfers tricky, they are often the dominant consideration in choosing a program's structure. A program that constantly jumps all over the place without apparent rhyme or reason will be difficult to understand. A program that uses lots of subroutines called by the JSR instruction is much easier to understand, but may prevent any practical use of the stack. On occasion, special and even peculiar methods of transferring control from one program segment to another may be rewarding.

In the following discussion, I'll mention numerous ways of JUMPing through a program. The related issue of conditional jumps or branches is ignored.

The Simple Jump: JMP

The basic 6502 jump command is simple, easily understood, and quite useful. The syntax is simply

```
FROM JMP THERE
THERE...
```

If the computer executes step FROM, it will jump to wherever THERE is, and execute that instruction next. Of course, it is vital that the address THERE contain valid 6502 instructions!

The Simple Subroutine: JSR

The Jump-Subroutine instruction JSR is almost as simple as the JMP instruction. When the computer executes the command

```
FROM JSR SUBROUTINE
```

it jumps to the address SUBROUTINE and continues execution. Before it jumps, though, it leaves a message on the stack containing the address of *the next instruction following the JSR command.*

When (and if) the subroutine wants to return control to the program that called it, it can simply terminate with the command

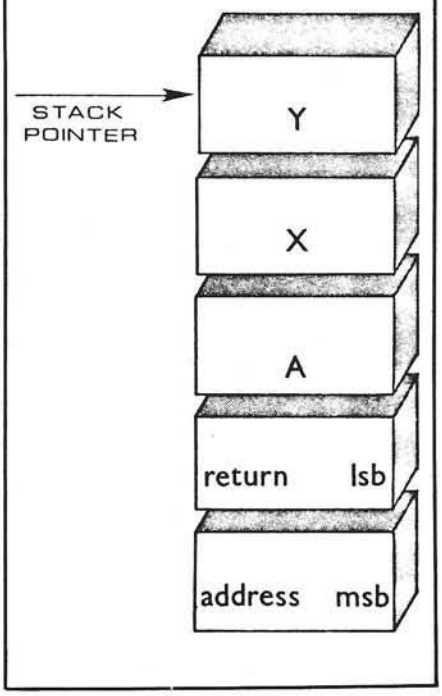
```
RTS
```

which jumps to the address on the top of the stack.

The usefulness of subroutines in writing simple, efficient programs is too familiar to discuss here. A common problem, though, is what to do with the machine registers (A, X, and Y) when calling a subroutine.

In the simplest case, we might allow the subroutine to alter any or all of the machine registers. This approach is fine for many cases, but clearly inadequate if the calling routine needs to preserve one or more registers containing data, loop counters, or other vital information.

Figure 1: **Arrangement of registers on stack after transfer of control.**



Alternatively, we may insist that the subroutine preserve all registers. To do this we may (and normally must) use the stack. All subroutines begin with a sequence:

```
PHA (save the A register)
TXA
PHA (save the X register)
TYA
PHA (save the Y register)
```

and end with the corresponding sequence:

```
PLA (retrieve the Y register)
TAY (put it back into Y)
PLA (retrieve the X register)
TAX (put it back into X)
PLA (retrieve the A register)
RTS
```

Note that the order in which registers are placed on the stack is important. The registers must be reloaded in exactly the reverse order in which they were pushed onto the stack. The sequence, "push A, then X, then Y" is standard and has the advantage of storing the registers in alphabetical sequence.

The programmer may take a page from the authors of the SYM monitor by ending all subroutines with a JMP to a special routine RESTORE:

```
        JMP RESTORE

RESTORE PLA
        TAY
        PLA
        TAX
        PLA
        RTS
```

Each subroutine must push the registers itself. This may be usefully written as a MACRO if the programmer's assembler is sufficiently powerful.

There are two drawbacks to this method of preserving registers. First, all control transfers are slowed rather drastically by all this pushing and pulling from the stack. Moreover, if subroutines are deeply nested, or especially if they are recursive, (that is, if a subroutine calls itself), all the extra registers on the stack may accumulate and exceed the 6502 stack capacity (256 bytes). A stack overflow is almost certain to cause a crash, and may be miserably difficult to detect if it is unexpected. Of course, each routine could check the stack pointer and abort the program if an overflow is imminent, but this checking adds still more overhead to the control transfer.

In addition, a program that uses the stack extensively for storing registers and subroutine return addresses must be very cautious in using the stack for anything else. In particular, a subroutine *must never leave more or less data on the stack* than were present on the stack when the subroutine was called. If the subroutine wants to leave a value on the stack, it cannot simply push it there:

```
LDA ANSWER
PHA
RTS
```

for the computer will treat ANSWER as part of its return address!

To avoid this disaster, it is possible to manipulate the stack more carefully. First, the return address is pulled from the stack and stored somewhere in memory. Next, the answer is pushed onto the stack. Finally, the return address is replaced on the stack. Notice that if the registers are saved on the stack, they too must be peeled off, stored, and then replaced. Even when the registers don't need to be preserved, the resulting code may seem a bit clumsy:

```
PLA (save the return address)
STA BUFFER1
PLA
STA BUFFER2
;
LDA ANSWER
PHA
;
LDA BUFFER2 (replace the
      return address)
PHA
LDA BUFFER1
PHA
RTS
```

Note also that the calling program or subroutine *must* remove the result ANSWER from the stack. If some program segment forgets to do this, disaster is likely to follow.

The Relocatable Jump

Occasionally, it is important to write a routine which will run, regardless of where it is stored in memory. For example, some utility programs are designed to be stored in the highest available memory locations of a user's computer, regardless of how much memory he actually has.

A normal absolute jump, though, needs to know the address to which the program should jump! If the program is moved in memory, the jump commands still point to the old locations. Of course, a computer program (called a relocator) could be written to search out all the absolute jumps and change them, but relocators are difficult to write and easy to fool.

The normal method of evading this difficulty on the 6502 is to use a *branch* instruction which *always* branches. Since the carry flag is changed infrequently, the most common sequence is:

```
SEC (force the branch)
BCS THERE (always branches)
```

Since branches can only jump (up or down) 127 bytes, this technique favors programs with only short-range jumps. When required, though, a sequence of jumps can move control over longer distances:

```
        SEC (start jumping)
        BCS ISLAND (go to ISLAND
          on the way to THERE)
        ;
        ;
ISLAND  SEC
        BCS THERE
        ;
        ;
THERE   ...
```

Self-Modifying Code

On occasion, it is necessary or convenient to compute the address to which the computer should jump. An obvious analogy is the BASIC statement:

```
ON I GO TO xxx,xxx,xxx
```

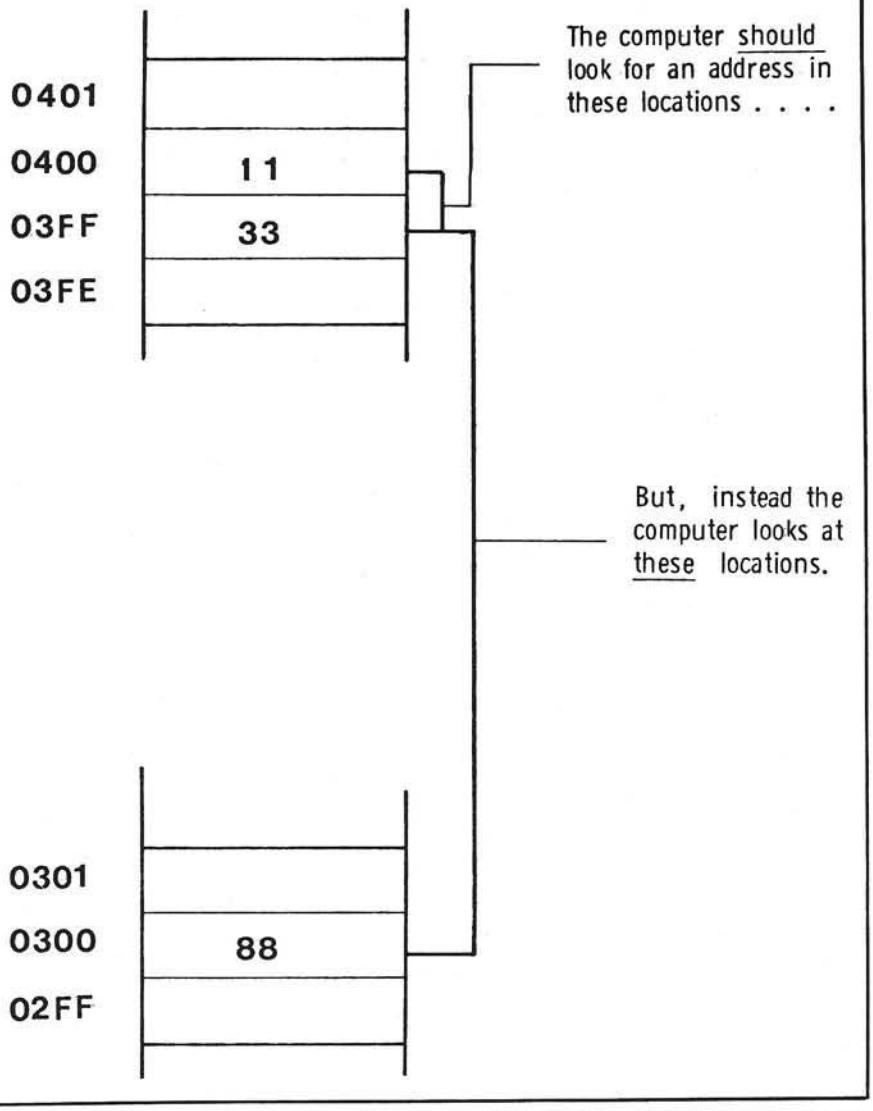
Provided that the program is (at least partially) stored in RAM and not in read-only memory, it is perfectly possible to change the address of a JMP or JSR instruction.

```
        LDA DESTINATION
          (least significant byte)
        STA JUMP + 1
          (store into the JMP instruction)
        LDA DESTINATION + 1
          (most significant byte)
        STA JUMP + 2
JUMP JMP...
```

If required, we could obtain DESTINATION from a table, compute it from a formula, or enter it from a terminal.

This approach is not without its hazards. First, it is dangerous. If the wrong address is stuffed into the JMP instruction, garbage is likely to be executed. If the computer is unlucky, the garbage will crash the computer, necessitating a system reset, which will erase all traces of the offending error.

Figure 2: **Illustration of bug in 6502 "JMP Indirect" instruction.**



Second, many programmers feel that data (things which are computed) and procedures (instructions which are performed) should be rigorously segregated. Preserving the separation of program and data improves clarity and helps ensure the integrity of both. Finally, many programs may need to be transferred to read-only memory at some point in their lives; clearly we cannot alter an address which is stored in read-only memory!

To use a self-modifying procedure in a read-only memory system, a copy of the self-modifying subroutine may be stored in ROM and transferred to read/write memory when the system is initialized. This approach is adopted by Microsoft in its 6502 BASIC. The subroutine NEXTCHAR, which obtains each character in sequence from the BASIC source program, is transferred from ROM to part of memory page zero whenever the system is turned on or reset. Sections of BASIC, and many extensions to Microsoft BASIC (like the "Programmer's Toolkit" produced by Palo Alto ICs Inc.) modify jumps in this routine.

While the above discussion used the JMP instruction as an example, it should be clear that the JSR instruction is equally suitable for use in a self-modifying sequence.

Indirect Jumps: JMP (addr)

A further method of selecting the address to which control will be transferred is the *indirect jump:*

```
JMP (WHERE?)
```

WHERE? is an address in memory which contains the first 8 bits of a 16-bit number. When the computer executes the indirect jump, it goes to the address *stored* in WHERE? The effect is identical to the self-modifying jump described above. The code is cleaner and easier to follow.

JMP (indirect) is also one of the most dangerous instructions of which the 6502 is capable.

Unfortunately, a confusing design decision (otherwise known as a bug) limits the usefulness of the indirect jump command. If WHERE? happens to be stored in an address ending in hexadecimal $FF (for example, at $03FF), the 6502 makes a bizzare and confusing jump. The programmer expects the computer to look at WHERE? for the least significant byte of the jump address, and at WHERE? + 1 for the most significant byte. Unfortunately, the 6502 does *not* carry when executing the JMP indirect instruction, and will look for the high-order byte *not at WHERE? + 1 but at WHERE? − $FF* if WHERE?'s address ends in $FF.

Fortunately, if the programmer knows and remembers this hazard, it is not too difficult to avoid. First, he can simply choose to locate all targets (like WHERE?) at addresses which do not end in $FF. Since indirect jumps are relatively uncommon, this is often perfectly reasonable.

Second, a self-modifying sequence will perform correctly. The disadvantages may be worthwhile if the time loss is not critical.

However, on rare occasions neither approach is fully satisfactory. An interesting example is the design of the 6502 FORTH language for the FORTH Interest Group (by W.F. Ragsdale). Here an indirect jump figures prominently in a crucial routine, which must run as quickly as possible since it is called during every single step in the FORTH program. Ragsdale resigns himself to check each and every possible target for the JMP indirect instruction, and relocates the whole program by a few bytes to move any offending addresses. Fortunately, there is only one chance in 256 that a given address will offend. The

programmer has some chance of getting away with ignoring this bug if he simply relies on luck.

Using RTS to Perform a Jump

RTS, of course, is normally used to terminate a subroutine. It performs two pulls from the stack, and jumps back to the address indicated by the two bytes it pulls. In normal usage, this returns control to the instruction that *follows* the JSR which originally invoked the subroutine.

Suppose a subroutine needs to return to an address different from that which called it. For example, we might define a subroutine called KEYBOARD that behaves as follows:

Subroutine Keyboard

1. Get the next key depression from the keyboard.
2. Convert the key code to an ASCII character.
3. Is the key the SYSTEM RESET key? *(a)* If not, then return normally (like a subroutine). *(b)* If so, return to the system monitor. Do not leave any junk on the stack!

End Keyboard

One way to implement step *(3b)* would be to fetch and discard the top two entries on the stack, and then use a simple JMP:

```
PLA (throw out the return
  address!)
PLA (leaving the stack where it
  should be)
JMP MONITOR
```

Alternatively, the address of the monitor could be put onto the stack *as if it were put there by a JSR instruction.*

```
PLA (throw out the old return
  address, as before)
PLA
LDA #L,MONITOR
PHA
LDA #H,MONITOR
PHA (put address of monitor onto
  the stack)
RTS (go to the monitor)
```

Of course, a subroutine can also change the return address of the subroutine that called it, by looking deeper into the stack. For example, we might have a subroutine WHAT-NEXT? that decides what the highest priority task for the computer is, and changes the return address.

Subroutine WHAT-NEXT?

1. Set pointer to DO-NOTHING.
2. Check all sensors.
3. If the room is on fire, set pointer to FIRE!
4. If the room is wet, set the pointer to FLOOD!
5. If the power is failing, set the pointer to FAMINE!
6. Change the return address of the routine that called WHAT-NEXT? to the routine indicated by the pointer.
7. Return to whatever routine called WHAT-NEXT?

A common jump usually performed using the RTS instruction is the *abort exit*. Often, it proves necessary to abort a program immediately upon detecting an error. If the program is called from BASIC or a machine language monitor, the usual method of exit is simply an RTS instruction.

If a subroutine must abort the program, it must be able to return to the master routine (e.g. BASIC), not simply to the routine that called it. If the program saves the stack pointer when it is first called

```
START TSX (stack contains exit
        address)
      STX ABORT
```

then any subroutine, regardless of what other data may be on the stack, can abort execution and exit by restoring the stack pointer:

```
QUIT  LDX ABORT (restore
        original pointer)
      RTS (back all the way to
        BASIC)
```

Software Interrupt:
The BRK Instruction

The BRK instruction allows an interrupt-request to be generated by the program itself. When the computer encounters a BRK, it takes the following steps:

1. Stores the status register and the current execution address + 2 on the stack.
2. Sets the BRK flag in the status register.
3. Sets the IRQ Inhibit flag in the status register.
4. Performs an indirect jump via hexadecimal addresses $FFFE-$FFFF.

The BRK instruction is most often used to jump to the system monitor. Since its opcode (00) is a common element in data tables, a program running amok has a fair chance to encounter a "BRK instruction" and so, return control to the monitor. Also, by writing BRK's on top of normal machine code, a simple software-controlled breakpoint monitor may be implemented and simple tracing performed.

Microcomputers, including the 6502, have simple interrupt structures, which cannot accept too much work. Moreover, the obstacles which make jumps difficult to debug make interrupt routines almost intolerably tedious to develop on most personal computers. Hence, use of the BRK routine should not be undertaken lightly.

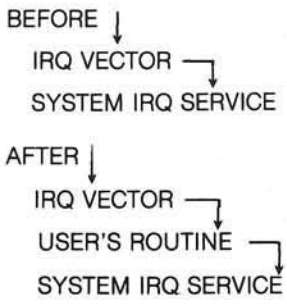
On occasion, though, BRK comes in handy, either when a special monitor-like routine is to be invoked, or for debugging normal interrupt service routines. Whenever you use it, be sure to remember that RTI will return to the calling program, skipping a byte after the BRK. BRK is really a two-byte operation. Also, remember that BRK inhibits interrupt requests. Some machines, notably the Commodore PET, use interrupt requests to check the keyboard or display. A mysteriously non-responsive system may result.

Changing the IRQ or NMI Vectors

The subject of interrupt handling is too complex to be considered here. However, changing the interrupt service routine vectors can be a valuable tool in several programs, and must be mentioned in the context of control transfers.

Most 6502 systems place the IRQ and NMI vectors in ROM, but use an indirect jump (explicit or via self-modifying code) through a RAM location to service interrupt requests (IRQs). The programmer may alter the RAM ad-

dresses to point to his own service routines. Frequently the user's IRQ service routines end by jumping to the normal (system) IRQ routines. For example:



Several computers generate periodic interrupts to service their keyboards, displays, or other peripherals. Other systems possess 6522 timer facilities which may be user-programmed to generate such interrupts. Setting the IRQ service vector to point to a user routine ensures that the new routine will be executed within a fixed period of time.

For example, I recently required a modified IRQ routine for the Commodore PET. The PET's screen is only 40 characters wide, and a program has to provide extended text messages and graphics simultaneously. The answer was to scroll the messages across one line of the screen in the manner of a stock-ticker display. The PET generates IRQs every 1/60th of a second. When the program had a message to send, it would add a new letter to the left end of the display line, three times a second. As long as more letters remained in the queue, the user IRQ routine was called. After the entire message was sent, IRQ handling reverted to normal until more text was placed in the queue.

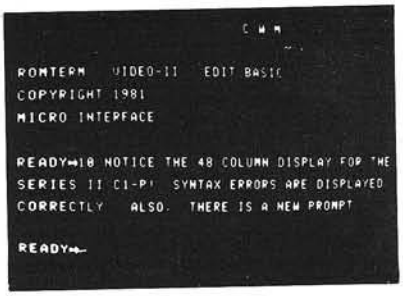
I should add that, whenever this program failed while it was being tested, it would leave a stream of amusing but useless text, cycling endlessly across the message line. The only way to stop the thing was to turn off the power!

The author gratefully acknowledges the support of the Merck Foundation, whose award of a Merck Foundation fellowship helped make this work possible.

Mark Bernstein is a graduate student in Chemistry at Harvard University. His research in picosecond laser spectroscopy, under the direction of Professor Kevin S. Peters, is assisted by several microcomputers. His most recent project has been implementing a FORTH-like language for laboratory data acquisition and analysis,based on an expanded Commodore PET.

WHICH EDITING "SMART-TERMINAL" MONITOR ROM IS BEST FOR YOUR C1P?*

ROM-TERM



ROM-TERM II

FEATURES AVAILABLE ON BOTH ROMS **AT THE TOUCH OF A KEY:**

- Enter "Smart Terminal" mode for **communications with** a time share Dial-up computer network.
- Select **half-duplex/full-duplex operation.**
- Select **auto or manual line feed** at carriage return.
- **Transmit a** pre-prepared **memory file** from C1P to remote computer: This memory file can be a message, letter or program and is sent **at full speed — saving time and telephone expense.** Programs can be listed to memory file while in basic.
- **Receive a message** or program (or all transactions) **into a memory file for later review** on video, recording on tape and printing. The file can be downloaded to basic after you exit the "smart terminal" mode.
- **Uploading/downloading of programs** can be done in this memory file manner or directly into basic by using a new serial output distributor and a new "Control-L" load command.
- Return to basic program operation at the same point of execution from which you entered the "smart-terminal" mode.
- "Smart-Terminal" mode can be utilized with the modem/telephone disconnected in order to prepare memory files, type directly to serial printer, send memory files to printer or tape, and to view tapes without interference from basic "Syntax Error."
- The serial output distributor can be turned on and off with a "control S" keystroke or with a poke which allows easy **control of a serial printer from basic.**
- **Basic program lines can be recalled, edited** and re-entered. The **editing includes backspacing, forward spacing, deleting, typing over, inserting new text, and changing line #** (duplicating a line). During editing, the cursor position and display are wrapped around, allowing operation on and displaying of an entire line up to 72 characters long. The preparation of line numbered messages can utilize these features — extremely handy for poor spelling, typists like me!
- **Keyboard** has been **completely corrected** to provide standard typing format. By the use of the control and repeat keys as modifiers, any character in the full **USASCII 128 character set can be entered from the keyboard. This will give you all the characters you need for running Pascal and other high level languages in a remote computer.**
- Video output may be halted at any time for easy viewing.
- Screen clear at keystroke.

ALL FEATURES ARE ROM RESIDENT AND ALWAYS AVAILABLE AT POWER ON.

ADDITIONAL FEATURES PROVIDED IN THE ROM-TERM:

- Disk bootstrap — In disk operation you can **alternate between ROM Basic and Disk Basic** with a keystroke. Can **Warm** Start **Disk Basic.**
- Easy transfer of programs between disk and ROM Basic. (Use the ROM Basic for editing disk basic programs.)
- **Memory files can be** stored or recalled in **Disk Basic, ROM Basic or the "Smart Terminal"** interchangeably.

ADDITIONAL FEATURES PROVIDED BY ROM-TERM II:

- A 48 column video display on series II C1P (Revision "D" Superboard). Selection of **48 or 24 column video with a** "Control-V" keystroke. Corrected "Syntax Error" messages.

***RECOMMEND THE ROM-TERM II** FOR NON-DISK OPERATION OF **SERIES II C1P** (OR REVISION "D" SUPERBOARD) AND THE **ROM-TERM FOR ALL OTHER SYSTEMS.**

ROM-TERM AND ROM-TERM II ARE A **LOT MORE THAN JUST "SMART TERMINAL" ROMS.** THE EDITING, **MEMORY FILE AND OUTPUT DISTRIBUTOR FEATURES ARE INVALUABLE.** EASY INSTALLATION. JUST PLUG INTO REGULAR C1P MONITOR ROM SOCKET. ONLY $59.95! ORDER NOW! Check with your Local Dealer. Phone orders/Visa/MC accepted. Sent Postpaid on prepaid Domestic orders.

MICRO-INTERFACE
3111 SO. VALLEY VIEW BLVD., SUITE I-101
LAS VEGAS, NEVADA 89102
Telephone: (702) 871-3263

KIM AIM SYM TIM

END FRUSTRATION!!

FROM CASSETTE FAILURES PERRY PERIPHERALS HAS THE HDE SOLUTION

OMNIDISK SYSTEMS (5" and 8")

ACCLAIMED HDE SOFTWARE

- **Assembler, Dynamic Debugging Tool, Text Output Processor, Comprehensive Memory Test**
- **HDE DISK BASIC NOW AVAILABLE**

PERRY PERIPHERALS S-100 PACKAGE

Adds Omnidisk (5") to Your KIM/S-100 System

- **Construction Manual—No Parts**
- **FODS & TED Diskette**
- **$20. +$2. postage & handling.** (NY residents add 7% tax) (specify for 1 or 2 drive system)

Place your order with:
PERRY PERIPHERALS
P.O. Box 924
Miller Place, N.Y. 11764
(516) 744-6462

Your Full-Line HDE Distributor/Exporter

OSI Disk Users

Double your disk storage capacity Without adding disk drives

Now you can more than double your usable floppy disk storage capacity—for a fraction of the cost of additional disk drives. Modular Systems' DiskDoubler® is a double-density adapter that doubles the storage capacity of each disk track. The DiskDoubler plugs directly into an OSI disk interface board. No changes to hardware or software are required.

The DiskDoubler increases free user disk space from 50K to 120K for mini-floppies, from 201K to 420K for 8-inch floppies. With the DiskDoubler, each drive does the work of two. You can have more and larger programs, related files, and disk utilities on the same disk—for easier operation without constant disk changes.

Your OSI system is an investment in computing power. Get the full value from the disk hardware and software that you already own. Just write to us, and we'll send you the full story on the DiskDoubler, along with the rest of our growing family of OSI-compatible products.

Modular Systems

P.O. Box 16A Oradell, NJ 07649
201-262-0093

Z-FORTH IN ROM by Tom Zimmer
5 to 10 times faster than Basic. Once you use it, you'll never go back to BASIC! $ 75.00
source listing add $ 20.00

OSI FIG-FORTH True fig FORTH model for 0S65D with fig editor named files, string package & much more $ 45.00

TINY PASCAL Operates in fig-FORTH, an exceptional value when purchased with forth.
TINY PASCAL & documentation $ 45.00
FORTH & TINY PASCAL $ 65.00

SPACE INVADERS 100% machine code for all systems with 64 chr. video. Full color & sound on C2, 4P & 8P systems. The fastest arcade program available. $ 14.95

PROGRAMMABLE CHARACTER GENERATOR $ 99.95
Use OSI's graphics or make a complete set of your own! Easy to use, comes assembled & tested.
2 Mhz. boards $109.95

PROGRAMMABLE SOUND BOARD $ 74.95
Complete sound system featuring the AY-3-8910 sound chip. Bare boards available. $29.95

32/64 CHARACTER VIDEO MODIFICATION $ 39.95
Oldest and most popular video mod. True 32 chr. C1P, or 32/64 chr. C4P video display.
Also adds many other options.

ROMS!!!
Augment Video Mod with our Roms. Full screen editing, print at selectable scroll, disk support and many more features. Basic 4 & Monitor $ 49.95
Basic 3 $ 18.95
All 3 for $ 65.00

65D DISASSEMBLY MANUAL. by Software Consultants. First Class throughout.
A must for any 65D user. $ 24.95

NUMEROUS BASIC PROGRAMS, UTILITY PROGRAMS AND GAMES ALONG WITH HARDWARE PROJECTS. ALL PRICES ARE U S FUNDS. Send for our $1.50 catalogue with free program (hardcopy) Memory Map and Auto Load Routine.

CHARGEX VISA

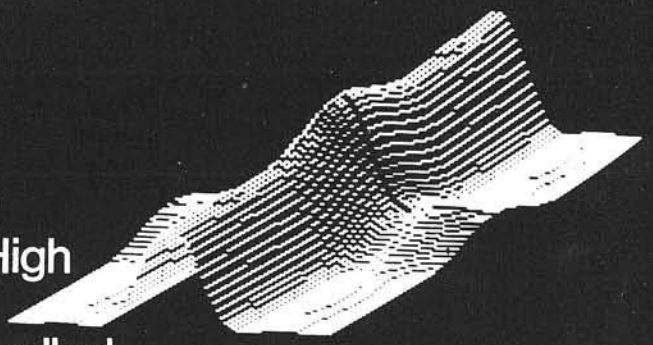
OSI Software & Hardware
3336 Avondale Court
Windsor, Ontario, Canada N9E 1X6
(519) 969-2500

3281 Countryside Circle
Pontiac Township, Michigan 48057
(313) 373-0468

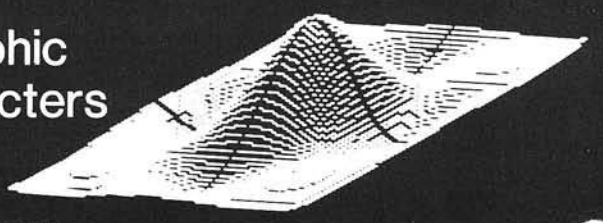
master charge

progressive computing

GRAPHICS
FOR OSI COMPUTERS

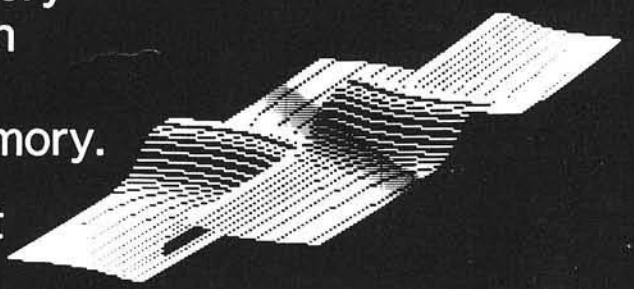


☆ You Can Produce The Images Shown Or Yours And Program Motion With Our 256 By 256 High Resolution Graphics Kit. Thats 65,536 Individually Controlled Points On Your TV Screen.



☆ Increase Column/Line Display. You Can Set Up Your Own Graphic Pixels Including Keyboard Characters And Unlimited Figures.

☆ This Kit Includes All Parts, Software And Assembly Instructions Required To Get Up And Running. The Included 8k Of 2114 Memory Is Automatically Available When Not Using The Graphics. Boot Up And See 8k More Memory.



☆ Adding The Kit Does Not Affect Your Existing OSI Graphics. Use Both At The Same Time Or Separately.

☆ Buy The Entire Kit, Including Memory, For $185.00 Or A Partial Kit For Less If You Have Parts. Board And Instructions $40.00. Instructions Include Software.

ME 01 10 ME

For This Kit Or A Catalog Of Other Kits, Software And Manuals Call Or Write:

MITTENDORF ENGINEERING
905 Villa Neuva Dr.
Litchfield Park, Az. 85340
(602)-935-9734

The Disk Switch

The Disk Switch will add many hours of useful life to your diskettes, extend the life of your disk drives, and even save energy!

Nelson E. Ingersoll
6910 S. Bannock #2
Littleton, Colorado 80120

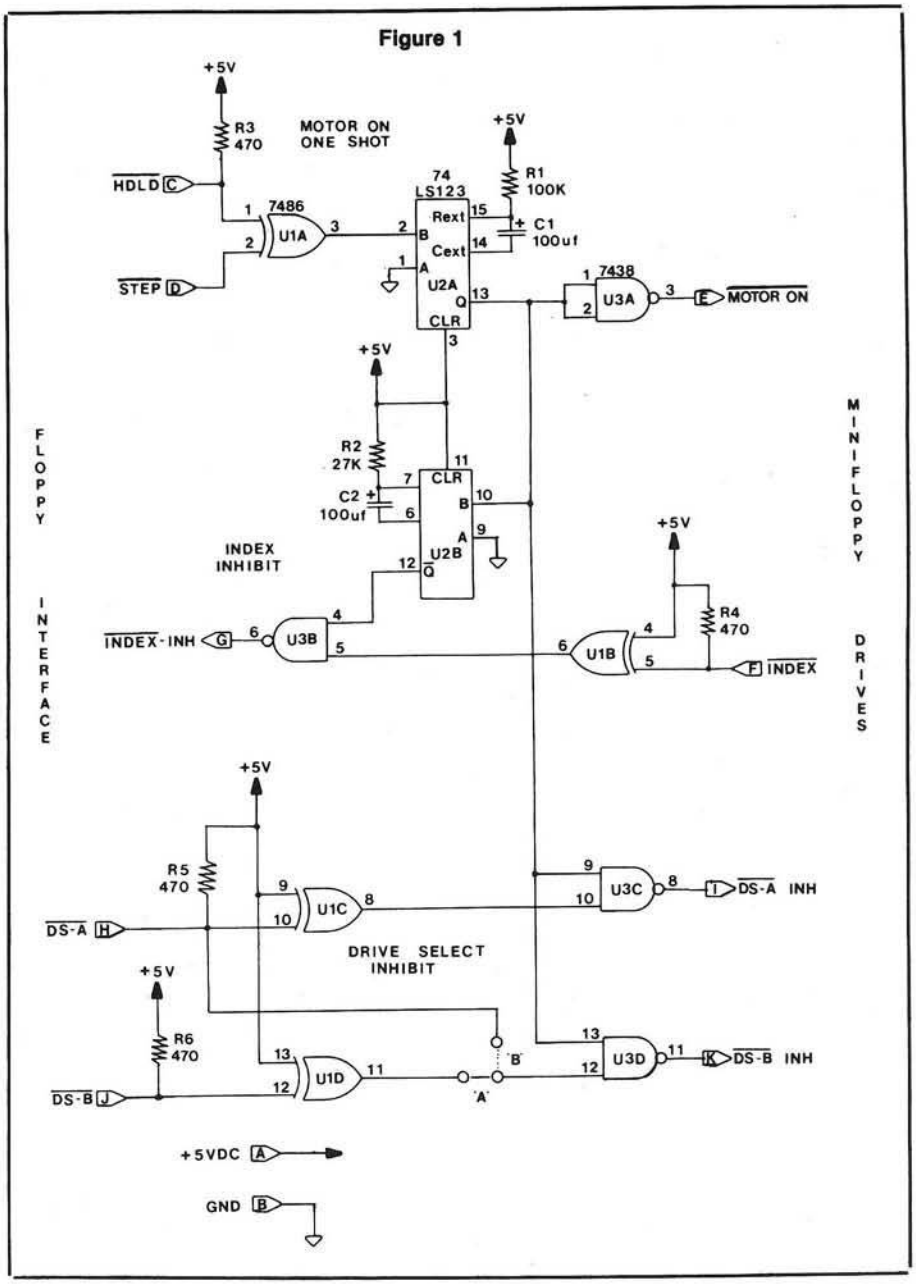
In the last year I have become very familiar with my OSI minifloppy system. I have learned its graces, and I have learned its faults. One of its faults is that OSI minifloppy drives have the nasty habit of eating diskettes.

Minifloppy drives have spindle motors which turn the diskettes inside their protective jackets. Depending on whether the head is loaded, data can be transferred from the rotating diskette to the system. When the disk head is loaded it actually comes into contact with the diskette.

This contact of the disk head with the rotating diskette is very hard on the diskette. It slowly wears away the oxide, which ultimately results in the destruction of the diskette. However, MPI, the company which makes OSI minifloppy drives, is not blind to the plight of diskettes. The MPI minifloppy drives have included in their circuitry the ability to turn off the spindle motor when the drive is not being accessed. MPI recommends that their drives be turned off two seconds (or ten revolutions of the diskette) after the last access.

Regretfully, OSI minifloppy systems do not do this. The MOTOR ON line is permanently wired in the ON state. OSI already had a working 8" floppy interface with the necessary software to drive it when they decided to mate a minifloppy with an OSI personal computer. Redesigning the interface and software would have cost a lot of money, and OSI customers would have had to pay that cost. So, OSI compromised. They kept the same floppy interface and software, added a few necessary modifications, and passed the savings on to us.

Figure 1



Nonetheless, OSI made it imperative that something like the Disk Switch be designed. It is the job of the Disk Switch to turn off the minifloppy spindle motors when the drives are not being

used. The Disk Switch works well for both single and dual minifloppy systems. It is reliable, and will extend the life of the media you use in your OSI system. Let's take a look at how the Disk Switch works.

Theory of Operation

Figure 1 is the Disk Switch circuit diagram. The Disk Switch monitors the HDLD (Head Load) and STEP outputs of the OSI floppy interface. Whenever either of these two signals is active, the Disk Switch activates the MOTOR ON line, thereby turning ON the minifloppy spindle motor.

The HDLD and STEP signals from the floppy interface are Exclusive OR'ed together by U1A, a 7486. The resulting signal is used to trigger U2A, the MOTOR ON 74LS123 one shot, into producing a two to three second output pulse. The output of the MOTOR ON one shot is inverted by U3A, a 7438 open collector NAND gate, to become the MOTOR ON signal to the minifloppy drives.

The MOTOR ON one shot and associated circuitry is the heart of the Disk Switch. All other functions of the Disk Switch are dependent upon the correct operation of this circuit. The 74LS123 is used for U2 because larger value resistors can be used in its timing circuit, thereby allowing the use of smaller capacitors. The 7438 open collector NAND gate is used because it can take the electrical load that the minifloppy imposes on the Disk Swtich.

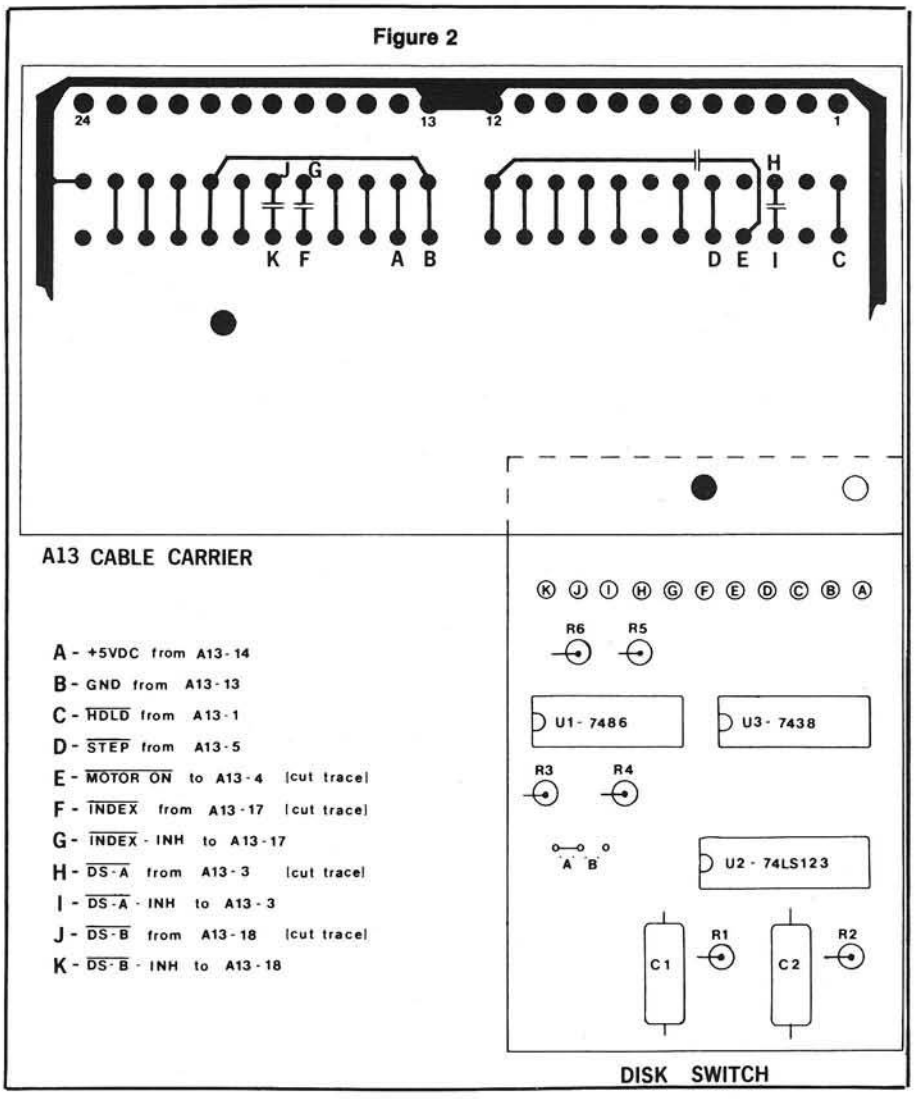
The Q output of the MOTOR ON one shot also drives two other circuits. The first is the INDEX Inhibit circuit; the second is the Drive Select Inhibit circuit.

INDEX is an output from the minifloppy drive which indicates the beginning of a track. The OSI floppy interface must have the Index signal. When Index is missing, the system will wait for it — forever if necessary. Thus, when there is no Index, the system is locked up.

I used this requirement of Index detection to give the spindle motor time to come up to speed before allowing the system to access the drives. The INDEX Inhibit one shot is triggered at the beginning of MOTOR ON. Its output will inhibit Index from getting to the floppy interface for about one second.

While inhibiting Index in this manner adds one second to the minifloppy access time, it is not unreasonable.

Figure 2



Once an access has begun, the INDEX Inhibit circuit will no longer affect the operation of the access. Whether the access is two seconds or twenty, only one extra second is added by the INDEX Inhibit circuit.

Note the wiring of the X-OR gate U1B at the input of the INDEX Inhibit circuit. In this configuration the X-OR gate has the function of an inverter. It is cheaper to get an inverter in this manner than to add another chip to the parts list.

At the bottom of the circuit diagram in figure 1 is the Drive Select Inhibit circuit. The Drive Select Inhibit de-selects the minifloppy drives when they are not being accessed. There are several reasons for this. The most important reason is to unload the disk head.

The disk head is loaded by the drive-select input when it is active. Unloading the head is useful in preserving the life of the diskette, and the disk head itself. An additional feature of the Drive Select Inhibit circuit is that it lowers the power consumption of the drive by placing it in the stand-by mode.

In operation, U1C and U1D invert DS-A (Drive Select A) and DS-B (Drive Select B) respectively. The inverted drive select signals are fed into U3C and U3D. U3 is a 7438 open collector NAND gate. The MOTOR ON signal on the second inputs of U3C and U3D prevents the drive select signals from being passed on unless MOTOR ON is a logic high.

Whenever the minifloppy drives have not been accessed for two seconds, the MOTOR ON signal from the MOTOR ON one shot is low. This low at the inputs of U3C and U3D forces their respective outputs high. The high output is the false or non-selected level, and neither of the drives is selected.

An additional feature of the Drive Select Inhibit circuit is that in a dual minifloppy system it can be jumpered to

cause the selection of drives A or B only. This prevents the selection of non-existent drives.

Note the jumper marked 'A' and 'B'. In the 'A' position $\overline{\text{DS-B}}$ is enabled such that its logical state selects drive B. When the jumper is in position 'B', $\overline{\text{DS-B}}$ is disabled and the uninverted state of $\overline{\text{DS-A}}$ selects drive B. This causes the selection of drive B whenever drive A is not selected. This feature will prevent the accidental selection of non-existent minifloppy drives C and D.

The Disk Switch can be simplifed by removing the Drive Select logic. It is not necessary to the operation of the Disk Switch, but helps to prevent diskette wear. In the event that the Drive Select Inhibit circuit is not used, the inputs of U1C, U1D, U3C, and U3D should be tied to +5VDC.

Two of the four 470 ohm pull-up resistors, R5 and R6, should also be removed if the Drive Select logic is not used. The pull-up resistors are necessary, since the interface between floppy and system is made with open collector gates. In the case of the four pull-up resistors, the associated signals are not terminated anywhere else.

By now you are probably wondering about the $\overline{\text{HDLD}}$ signal from the floppy interface. What is it for? The answer is simple — nothing! It is a relic of the floppy interface software used with the larger 8" disk drives. However, for our needs the $\overline{\text{HDLD}}$ signal is very useful. Assume that you need to access a track that has already been found. Since the head is already positioned over the correct track, the system will not generate a STEP signal. The only indication of the access will be the $\overline{\text{HDLD}}$ signal.

Without the use of the $\overline{\text{HDLD}}$ signal an access to the same track would not trigger the $\overline{\text{MOTOR ON}}$ signal, the spindle motor wouldn't spin, so the Index signal would not be generated. Therefore, the system would lock up waiting for the Index signal. It was kind of OSI to leave the $\overline{\text{HDLD}}$ signal lying around like that.

Construction

I have already constructed several versions of the Disk Switch, and have installed the final version in three operational systems. All versions of the Disk Switch that I have built easily fit on a 3 by 2 inch perforated circuit board.

I have used both wire wrap and point-to-point soldering to construct the Disk Switch. I do not recommend wire wrap because of the posts that protrude from the board. Wire wrapping the circuit is definitely easier, but the circuit board must be placed in a tight location. I suggest that the circuit be soldered point-to-point. Since there are no high frequency signals in the Disk Switch, the actual layout is not critical.

The Disk Switch should be mounted on the A13 floppy interface board to facilitate wiring. The A13 board is the cable carrier for the minifloppy cables, and plugs onto the 505 CPU board.

Mounting the Disk Switch on the A13 board is best since all of the signals used are available there. Servicing the Disk Switch is also much easier. The only precaution is to make sure the Disk Switch does not touch the 505 CPU, or whatever board may be located nearest it.

Attaching the input and output signals to the Disk Switch is a relatively simple matter. Just locate the appropriate traces on the A13 board, make the necessary cuts, and attach the wires. In all cases, the wires from the Disk Switch can be soldered into existing pads.

Figure 2 shows where to mount the Disk Switch on the A13 board. All trace cuts are shown, as well as the locations of the various connections to the Disk Switch.

In conclusion I will leave you with a word of caution. You have probably acquired the habit of opening the disk drive door when you don't need to access the disk. It is a reasonable habit that reduces diskette wear. With the Disk Switch this is no longer necessary. You should now leave the door closed when you are using the system. Otherwise, with the door open, the system will miss the Index signal when you try to access a disk. The Disk Switch will time out, and lock you out of your system.

There is a simple fix for this that you might wish to try. You can add a pull-up resistor and a grounding switch to pin 1 of U2, the MOTOR ON one shot. This would allow you to independently trigger the $\overline{\text{MOTOR ON}}$ signal and restart the minifloppy drive.

(If you don't feel up to the task of constructing the Disk Switch, you can obtain a kit from D&N Micro Products, 3684 N. Wells St., Fort Wayne, IN 46808. The kit contains all of the necessary parts and costs $29.00 plus $2.00 shipping and handling.)

CBM/PET? SEE SKYLES ... CBM/PET?

"They laughed when I sat down at my PET and immediately programmed in machine language... just as easily as writing BASIC."

With the new Mikro, brought to you from England by Skyles

Electric works, always searching the world for new products for PET/CBM owners. A 4K machine language assembler ROM that plugs into your main board. At just $80.00 for the Mikro chip, it does all the machine language work for you; all you have to do is start laying down the code.

The Mikro retains all the great screen editing features of the PET...even all the Toolkit commands. (If you own a Toolkit, of course.) Sit down and write your own machine language subroutine. The program you write is the source code you can save. And the machine language monitor saves the object code. The perfect machine language answer for most PET owners and for most applications. (Not as professional as the Skyles Macro-TeA...not as expensive, either.)

A great learning experience for those new to machine language programming but who want to master it easily. Twelve-page manual included but we also recommend the book, "6502 Assembler Language Programming," by Lance A. Leventhal at $17.00 direct from Skyles.

Skyles guarantees your satisfaction: if you are not absolutely happy with your new Mikro, return it to us within ten days for an immediate, full refund.

Skyles Mikro Machine language assembler **$80.00**

"6502 Assembler Language Programming" by Leventhal **17.00**

Shipping and Handling ..... *(USA/Canada)* **$2.50** *(Europe/Asia)* **$10.00**

California residents must add 6%/6½% sales tax, as required.

Skyles Electric Works
231E South Whisman Road
Mountain View, California 94041
(415) 965-1735

Visa/Mastercard orders: call tollfree **(800) 227-9998** (except California). California orders: please call **(415) 965-1735.**

SEE SKYLES ... CBM/PET? SEE SKYLES

... CBM/PET? SEE SKYLES ... CBM/

Color computer owners,

32K PLUS DISKS*

$298.00

Yes, that's right - for as little as $298.00 you can add 32K of dynamic RAM, and a disk interface, to your TRS-80 Color Computer! If you just want the extra memory it's only $199.00, and you can add the disk interface later for $99.00.

Just plug the *Color Computer Interface (CCI)*, from Exatron, into your expansion socket and "Hey Presto!" - an extra 32K of memory. No modifications are needed to your computer, so you don't void your Radio Shack warranty, and Exatron give both a 30 day money-back guarantee and full 1 year repair warranty on their interface.

The *CCI* also contains a 2K machine-language monitor, with which you can examine (and change) memory, set break-points, set memory to a constant and block-move memory.

So what about the *CCI Disk Card*? Well as we said it's only an extra $99.00, but you'll probably want Exatron's *CCDOS* which is only $29.95 – unless you want to write your own operating system. The *CCI Disk Card* uses normal TRS-80 Model I type disk drives, and *CCDOS* will even load Model I TRSDOS disks into your color computer – so you can adapt existing TRS-80 BASIC programs.

As a further plus, with the optional *ROM Backup* adaptor, you can dump game cartridges to cassette or disk. Once the ROM cartridge is on cassette, or disk, you can reload, examine and modify the software. The *ROM Backup* adaptor is only $19.95.

For more information, or to place an order, phone Exatron on their Hot Line *800-538 8559* (inside California 408-737 7111), or clip the coupon.

excellence in electronics

exatron

DEALER ENQUIRIES INVITED

Exatron,
181 Commercial Street,
Sunnyvale, CA 94086

☐ Please send a 32K Color Computer Interface for $199.00

☐ Please send a CCI Disk Card for $99.00

☐ Please include CCDOS and manual for $29.95

☐ Also include a ROM Backup adaptor for $19.95

Please add $5.00 for shipping to all orders, and 6 percent sales tax in California.

Name .....

Address .....

City .....

State ..... Zip .....

Charge my:

☐ MasterCard Interbank Code .....

☐ Visa Expiration Date .....

Card .....

☐ Check enclosed for .....

☐ Ship COD ($2.00 extra)

Signature .....

*32K RAM plus Disk Interface TRS-80 is a trademark of Tandy

STOP!

STOP BUYING THOSE "CANNED" SOFTWARE PROGRAMS THAT TURN OUT TO BE COSTLY, INEFFECTIVE AND INCOMPLETE. FINALLY THERE IS ONE SYSTEM, ONE SYSTEM THAT CONTAINS ALL THE POWER, ALL THE VERSATILITY YOU'LL NEED TO MAKE PROGRAMING FASTER, SIMPLER AND ERROR FREE. **C.O.R.P.**™ IS AN APPLICATION GENERATOR THAT ACTUALLY WRITES ALL THE SOFTWARE (COMPLETE APPLESOFT PROGRAMS THAT EXECUTE INDEPENDENTLY OF THE **C.O.R.P.**™ SYSTEM AND MAY BE MODIFIED BY THE USER) BASED ON INFORMATION SUPPLIED BY THE USER IN ENGLISH. COMPLETE PROGRAM GENERATION TAKES ONLY MINUTES.

C.O.R.P.™

THE FIRST ONE, NOT THE LAST ONE IN PROGRAM GENERATORS

C.O.R.P.™ CONSISTS OF 5 MAJOR SUBSYSTEMS: A DATA ENTRY PROGRAM GENERATOR A SORT SUBSYSTEM, A UPDATE SUBSYSTEM, A PRINT GENERATOR AND A COMPLETE DIAGNOSTIC PACKAGE. TOGETHER THE 5 SUBSYSTEMS MAKE **C.O.R.P.**™ THE MOST ADVANCED, COMPLETE AND USEFUL TOOL AVAILABLE FOR YOUR APPLE II*, ALL FOR THE INCREDIBLY LOW PRICE OF $189.95**. DOLLAR FOR DOLLAR **C.O.R.P.**™ IS THE BEST BUY IN TOWN! NO OTHER PROGRAM GENERATOR OFFERS YOU SO MUCH VERSATILITY FOR SUCH A SMALL PRICE. TO PURCHASE **C.O.R.P.**™ SYSTEM BRING THIS AD TO YOUR LOCAL DEALER OR WRITE US DIRECTLY.

MAROMATY & SCOTTO SOFTWARE CORP.

P. O. BOX 610 • FLORAL PARK, N. Y. 11001

© Copyright 1981 Maromaty & Scotto Software Corp.

*APPLE is a Registered Trademark of APPLE COMPUTER, INC.
**Plus Handling and Shipping. N.Y. State Residents please add Sales Tax.

It's Time to Stop Dreaming

Part 4

By Robert M. Tripp

Parts 1 and 2 (MICRO 37:07 and MICRO 38:27) of this series introduced the 6809 microprocessor. Part 3 (MICRO 39:16) described the 6809 instruction set from the 6502 user's point of view. That topic was easy to cover since there are so many similarities between the two micros. This month's article discusses the addressing modes of the 6809, and was more difficult to write. While the 6809 and 6502 do share a number of common modes (although often the nomenclature is different), the 6809 offers a number of new modes which will take some getting use to for the 6502 devotee. These additional modes are very significant and contribute a lot to the overall functioning of the 6809. So, the effort involved in understanding them is well spent.

Common Modes

The following modes are essentially identical between the two micros. The name for each mode is different, but the processing is identical.

IMPLIED or ACCUMULATOR (6502) and INHERENT (6809)

This mode is used when the effective address of an instruction is "implied" or "inherent" in the instruction itself. Examples on the 6502 are CLC (clear carry bit in status register), INY (increment Y register), RORA (rotate accumulator right, and so forth. "Accumulator" is another form of "implied." Examples on the 6809 are ABX (add B register to X register), CLRA (clear A register), and so forth. The basic addressing element of this mode is that any registers which will be affected are specified within the opcode and do not require additional information.

IMMEDIATE

The "immediate" mode is identical on the two processors. It takes as its effective address the location(s) immediately following the opcode. On the 6502 the immediate data is always 8 bits since that is all that the 6502's registers can handle. On the 6809 the immediate data may be 8- or 16-bit. For example, an LDA #$05 would be 8 bits into the A register, but an LDX #$1234 would load 16 bits into the X register.

ABSOLUTE (6502) and EXTENDED (6809)

These two modes are identical. The effective address is provided in the two bytes following the opcode. LDA $1234 will load A from the absolute/extended address $1234. This mode of addressing permits the processor to directly access any location in its 64K addressing space.

ABSOLUTE INDIRECT (6502) and EXTENDED INDIRECT (6809)

The only instruction on the 6502 which can use the absolute indirect addressing mode is the JMP. Any other instruction which requires an absolute indirect mode must either set the X register to zero and use the Indexed Indirect mode LDA (ADDR,X), or set the Y register to zero and use the Indirect Indexed mode LDA (ADR),Y. The 6809 does not require such chicanery, but provides the Extended Indirect mode for all major operations, such as LDA (JUNK). This makes indirect addressing much simpler.

ZERO PAGE (6502) and DIRECT PAGE (6809)

These two modes are conceptually similar, and, for the zero page of memory, are identical. On the 6502, any address in page zero may be absolutely referred to by providing only the low 8 bits of the address, with the mode defined by the opcode specifying that the high 8 bits are all zero. This makes accessing the 256 locations on page zero faster and requires only one byte of addressing instead of two.

The 6809 carries this concept of only providing the low byte of the address a bit further. Instead of forcing the high byte of the address to zeros, it has a separate register called the "Direct Page Register," which contains the high byte of the address. If the DP register is set to zero, which it is automatically on RESET, then a "direct page" mode reference will be identical to the 6502 "zero page" mode. But, the DP reigster may be changed to any other value, allowing the directly referenced page to be anywhere in memory!

This function can be very useful. First, it removes the limitation of only 256 bytes of directly addressable memory. Second, and probably more significant, it will support multiple tasks and multiple users. Code can be written which makes all task/user references in the direct page mode. By setting the DP register to a different value for each task/user, there will be no conflict and multiple tasks/users can easily share the same program code. One other addition to the 6809 mode is that JMP and JSR instructions are permitted via the Direct page.

RELATIVE (6502 and 6809)

This mode is used on both micros for the Branch operations. The data following the branch opcode is added to the Program Counter (PC) to determine the effective address for the branch. On the 6502, the signed data is limited to one byte, providing a branch range of 127 locations forward and 128 locations backward. The 6809 permits the signed data to be either one byte, in which case

it is identical to the 6502, or two bytes, in which case any address in the normal 64K addressing space is accessible.

The 8-bit mode is called a "short" branch and the 16-bit mode is called a "long" branch. (*Note:* Several years ago the savings associations of Red Bank, NJ and Long Branch, NJ were planning a merger, but it fell through because they did not want to be known as the Long Branch Branch of the Red Bank Bank! But, I digress.) Since the 6809 has a Branch to Subroutine and a Branch Always, the value of the long branch is even geater since JSRs and JMPs can be replaced by the relative BSRs and BRAs.

INDEXED Addressing on the 6502

Since the 6502 has 8-bit index registers, its indexing can only cover a range of 256 bytes and the 16-bit base address for all indexing operations must come from somewhere else. This may be an ABSOLUTE 16-bit address as in LDA JUNK,X or STA JUNK,Y. It may be a PAGE ZERO address which assumes the high 8 bits to be zero and directly specifies the low 8 bits as in LDA JUNK,ZX or CMP JUNK,ZY. Or, it may be one of the two special modes which combine indexing and indirect addressing.

INDEXED INDIRECT uses X as an index to a set of Page Zero byte-pairs which are 16-bit addresses to the final effective address anywhere in memory. An example is ADC (JUNK,X) which will take the byte-pair that is X locations above JUNK in Page Zero as the effective address for the operation. INDIRECT INDEXED uses a Page Zero byte-pair as the 16-bit address of the start of a 256-byte section of memory which is indexed by Y. An example is SBC (JUNK),Y which will take the 16-bit value of JUNK and JUNK + 1 in Page Zero and add to it the current contents of Y to form the effective address.

All of the indexed address modes are not available for all instructions, which while not fatal, can be a nuisance. Have you ever written a clever piece of code based on an INC (JUNK),Y only to find that there is no such animal?! Although the designers of the 6502 gave us some very useful indexed addressing modes, they were limited by the fact that the X and Y registers are only 8-bit.

INDEXED Addressing on the 6809

The best way to understand the 6809 indexed addressing modes is to first forget all about the 6502's. While there are some similarities, they are outweighed by the great differences. So, let's depart from the comparative evaluation of the 6809 and look at the index addressing from scratch.

One of the most important features of the 6809 is that its index registers are all 16-bit. A second significant feature is the multiple index registers: X, Y, S, U, and sometimes PC. That's right! The two stack registers, S and U, can perform all of the indexing operations, and the program counter, PC, can be used in several special index operations. This is quite different from the 6502 with its two 8-bit index registers. There are three major categories of indexed addressing modes which can use the X, Y, S and U registers.

Constant Offset from X, Y, S or U

In this mode, a constant value is added to the contents of the specified index register to form the effective address. This is similar to the Absolute Indexed mode of the 6502: LDA JUNK,X in which JUNK is a 16-bit constant value to which is added the current value of X to form the effective address. The 6809 supports this 16-bit mode of constant offset, but also provides three other modes: No offset, 5-bit offset, and 8-bit offset. Since the index register is 16 bits, it can contain the entire effective address. It makes sense, then, to have a mode in which the contents of the index are used directly without any additional constants. The instruction of the form LDA ,X on the 6809 will use the current contents of X as the effective address.

There are many instances in which the offsets required are within a limited range. For example, if an index points to the base address of a VIA device which has 16 internal registers, only four bits of offset would be required to access all registers. Why use 16 bits when fewer will suffice (especially if you can save time and space with a smaller value)?

The 5-bit constant offset of the 6809 provides for a signed offset which can cover the range of + F to − F locations from the 16-bit index. The format of the instruction is the same as for the 16-bit constant offset: CLR JUNK,Y but JUNK is now a 5-bit or less signed value. Similarly, the 6809 provides an 8-bit constant offset. It covers a range of +7F to −7F locations from the 16-bit index. In summary, the 6809 provides four forms of Constant Offset indexing: No Offset, 5-bit Offset, 8-bit Offset and 16-bit Offset. Each mode of increasing the number of bits requires either more instruction cycles, more instruction bytes, or both.

Accumulator Offset from Index Register

The 6809 can use one register to index another. Any one of the accumulators, A, B, or D, can be combined with the X, Y, S or U registers. The instruction form is INC B,U which will perform a two's complement addition of the B register and the U register to form the effective address. All combinations are valid and may be used for all major operations. The Accumulator Offset mode can be very powerful in performing table lookups in which the position within the table is calculated by some routine and then implemented through this indexing mode. The A and B registers provide 8-bit signed offsets; the D register provides a 16-bit signed offset.

Auto Increment/Auto Decrement

One very common use of an index is to step through a set of values. This is used in searching a table, moving a number of bytes from one location to another, and so forth. The 6809 provides four modes which either increment or decrement the index register as part of each operation. All four index registers may be used. They may be incremented or decremented by one when used for stepping through a table of single bytes, or may be incremented or decremented by two when working with 16-bit words, such as addresses. The form of the single increment is LDA ,X+ in which X has the 16-bit address which is used by the instruction as the effective address, and which is then incremented by one after the operation has taken place. The double increment form is LDA ,Y++ and the register is incremented twice after each operation.

The decrement modes work in an identical fashion except that the decrement is done *before* the operation. If X contained $2000 at the start of a decrement instruction such as LDA ,−X then the value loaded into A would come from location $1FFF. A double decrement instruction would be STA ,−−S. Note that the minus sign(s) in the decrement comes before the register and that the plus sign(s) in the increment comes after the register. This indicates the order of the operation: decrement before the operation and increment after the operation.

Program Counter Relative Addressing

The previously discussed indexing modes all use the X, Y, S or U registers. It is very important in writing position-independent code to be able to refer to data in a position-independent manner. One way this is supported on the 6809 is

Indexed Addressing Mode Summary for the 6809

Type	Forms	Non-Indirect	Indirect
Constant Offset from R (2's Complement Offsets)	No Offset	,R	[,R]
	5-bit Offset	n,R	defaults to 8-bit
	8-bit Offset	n,R	[n,R]
	16-bit Offset	n,R	[n,R]
Accumulator Offset from R (2's Complement Offsets)	A Register Offset	A,R	[A,R]
	B Register Offset	B,R	[B,R]
	D Register Offset	D,R	[D,R]
Auto Increment/Decrement	Increment by 1	,R+	not allowed
	Increment by 2	,R++	[,R++]
	Decrement by 1	,-R	not allowed
	Decrement by 2	,--R	[,--R]
Constant Offset from PC (2's Complement Offsets)	8-bit Offset	n,PCR	[n,PCR]
	16-bit Offset	n,PCR	[n,PCR]
Extended Indirect	16-bit Address	xxxxx	[n]

NOTES: R is X, Y, S or U register.
Extended Indirect is not an index mode but is included in the table for the sake of completeness on the Indirect Addressing forms.

through an indexing mode which uses the current value of the Program Counter Register (PCR) as the base for calculating the effective address. There are two forms of this mode: 8-bit offset and 16-bit offset. These two modes are identical except for the size of the offset. LDA JUNK,PCR will add the 8-bit or 16-bit offset of "JUNK" to the current value of PCR to calculate the current location of JUNK. This is identical in concept to the Relative Addressing used by the Branch instructions. Combined with the Load Effective Address instruction, this mode permits the easy loading of an index register with the relocated address of a table. LEAY TABLE,PCR will put the absolute addresss of TABLE into the Y register where it can be used by the other modes of indexing to access the entire table.

INDIRECT Addressing on the 6809

All of the indexing modes discussed above, with the exception of the Auto Increment One and Auto Decrement One, can also have a level of Indirect addressing. This means that all of the calculations which go into producing the Effective Address take place as usual, but then this Effective Address is used to access a location which contains the final Effective Address. This provides many modes for implementing the Indexed Indirect which the 6502 supports in a limited manner for the X register only.

The Indirect Indexed mode, which is very important on the 6502, is not needed as such on the 6809. Since the 6809 has a variety of ways of getting the correct 16-bit address into one of its four index registers, there is no real need for the "indirect" portion of the mode. Why specify an address "indirect" on page zero as in the 6502 when it can be specified "direct" so easily on the 6809? The "index "portion of the mode is, of course, supported by all of the normal 6809 indexing modes.

Summary

This four-part series has presented an introduction to the 6809 microprocessor as seen through the eyes of a 6502 user. If it has encouraged you to find out more about this potential successor to the 6502, then the series has served its purpose. As an admitted novice on this particular microprocessor, I have tried to make an accurate presentation of the major features of the 6809, but may have made some mistakes! If you are serious about using this device, I recommend that you obtain the MC6809 or MC6809E data sheet from your Motorola representative.

I am actively working with the 6809E on the Flexi Plus board that my sister company The COMPUTERIST, Inc. introduced in August 1981. I am sure that I will be finding out a lot more about this micro and will write additional articles as needed. Meanwhile, several very knowledgeable 6809 computerists have offered to write articles. It will obviously take time to get our coverage of the 6809 rolling, so, if you have 6809 information to contribute, please contact me.

MICRO™

6809 SYSTEMS **6809 SYSTEMS** **6809 SYSTEMS** **6809 SYSTEMS**

Featuring the **GIMIX** mainframe with 30 amp C.V. ferro-resonant power supply; fifteen 50 pin and eight 30 pin slot Mother Board; 2 Mhz CPU with time of day clock & battery back-up, 1K RAM, 6840 programmable timer, provisions for 9511A or 9512 Arithmetic processors, and 4 PROM/ROM/RAM sockets that can hold up to 32KB of monitor or user software.

VARIETY: you can have 32KB, 56KB, 128KB and up of static RAM. You can use 5" and/or 8" disk drives, single or double density, single or double sided, and single or double tracking with **GIMIX** disk controllers. You have a wide choice of serial or parallel I/O cards.

EXPANDABILITY: You can add memory, I/Os, Video or Graphics cards, Arithmetic processors, additional drive capacity, and other hardware now or in the future to this SS50 bus structured system from **GIMIX** or other SS50 bus compatible manufacturers.

SOFTWARE VERSATILITY: GIMIX systems can use TSC's FLEX or UNIFLEX and MICROWARE'S OS-9 operating systems. A wide variety of software and languages is available for these systems.

QUALITY: All boards are assembled, burned-in, and tested and feature **GOLD PLATED BUS CONNECTORS.** Only top quality components are used and all boards are fully buffered for maximum system expansion. All boards come complete with bus connectors and all necessary instruction and documentation.

GIMIX designs, manufactures and tests, in-house, their complete line of products. Complete systems are available to fit your needs. Please contact the factory if you have any special requirements.

For further information, pricing and brochures, contact:

GIMIX INC.
The Company that delivers Quality Electronic products since 1975.
1337 WEST 37th PLACE, CHICAGO, IL 60609
(312) 927-5510 • TWX 910-221-4055

GIMIX* and GHOST* are registered trademarks of GIMIX Inc.

Flex and Uniflex are trademarks of Technical Systems Consultants Inc. OS9 is a trademark of Microware Inc. See their ads for other **GIMIX** compatible software.

A TEAM OF 6809 SUPERSTARS:
Smoke Signal's Chieftain™ Computer, and Software by Microware

HERE'S THE *TOTAL* 6809-BASED SYSTEM FOR THOSE WHO DEMAND UNSURPASSED POWER, FLEXIBILITY AND RELIABILITY

After years of worldwide use in diverse and challenging applications, the outstanding performers in 6809 computer operations are SMOKE SIGNAL and MICROWARE. These leading companies are recognized as the undisputed choices **when there is no room for compromises**.

WHY SMOKE SIGNAL AND MICROWARE LEAD THE 6809 FIELD

Smoke Signal began pioneering research and development on 6800/6809-based computer systems back in 1977. Microware worked three years to perfect OS-9 and BASIC09.

Both companies have evolved outstanding 6809-based products from early engineering research, **and both pay almost fanatical attention to detail**. For example . . .

SMOKE SIGNAL'S 6809-based Chieftain™ computer series has **proven** its superiority in hundreds of demanding tasks. From gold-plated connectors to highest-quality materials throughout, each Chieftain™ is built to deliver absolute dependability from day one, and **stay** that way through years of service.

• ENDURANCE-CERTIFIED . . . an exclusive Smoke Signal quality-control measure that positively verifies a component is free of defects, and meets or exceeds all specifications.

Every Chieftain™ is meticulously ENDURANCE-CERTIFIED at 2.2 MHz. That's SMOKE SIGNAL's endorsement of product perfection.

MICROWARE's state-of-the-art OS-9 UNIX*-like operating system and the BASIC09 language have been developed in close coordination with computer manufacturers to maximize optimum system performance. The finest possible support and documentation further ensure satisfaction. Microware software performance is best summed up in this remark by a 25-year computer veteran:

"BASIC09 IS THE FINEST HIGH-LEVEL LANGUAGE I'VE EVER SEEN IN THE INDUSTRY!"

Thousands of engineers and programmers use MICROWARE software products as their standard time-saving tool . . . to execute process-control applications . . . and for other vital functions. COBOL and PASCAL are also available under the OS-9 operating system.

*UNIX is a trademark of Bell Telephone Laboratories.

HOW THIS REMARKABLE TEAM OF COMPUTER SUPERSTARS CAN SERVE *YOU*

SMOKE SIGNAL's Chieftain™ computer provides an array of configurations ranging from 5¼-inch drives for single-user applications to multi-user, multi-tasking capabilities. Winchester hard-disk drive systems are also available.

In other words, **breathtaking power** with as little as 48k memory; Microware's OS-9 Level Two can access up to one full megabyte that your Chieftain™ can address!

One more sampling of the awesome processing potential at your fingertips with the Smoke Signal Chieftain™ computer:

MICROWARE'S Stylograph screen-oriented word processing package instantly makes Chieftain™ an easy-to-use document preparation system with comprehensive editing commands.

THERE'S MUCH, MUCH MORE! Call or write SMOKE SIGNAL for details on Chieftain™ computers and MICROWARE software.

SMOKE SIGNAL Dealer opportunities are still available . . . please request information.

☐ Send information about Chieftain™ computers and Microware software.

☐ Provide information about Smoke Signal's Dealer program.

Name _____

Address _____

City _____ State _____ Zip _____

Telephone ( ) _____

SMOKE SIGNAL BROADCASTING | MICROWARE

31336 VIA COLINAS
WESTLAKE VILLAGE, CA 91362
TEL (213) 889-9340

The Newest In

Apple Fun

We've taken five of our most popular programs and combined them into one tremendous package full of fun and excitement. This disk-based package now offers you these great games:

Mimic—How good is your memory? Here's a chance to find out! Your Apple will display a sequence of figures on a 3 × 3 grid. You must respond with the exact same sequence, within the time limit.

There are five different, increasingly difficult versions of the game, including one that will keep going indefinitely. Mimic is exciting, fast paced and challenging—fun for all!

Air Flight Simulation—Your mission: Take off and land your aircraft without crashing. You're flying blind —on instruments only.

A full tank of fuel gives you a maximum range of about 50 miles. The computer will constantly display updates of your air speed, compass heading and altitude. Your most important instrument is the Angle of Ascent/Bank Indicator. It tells if the plane is climbing or descending, whether banking into a right or left turn.

After you've acquired a few hours of flying time, you can try flying a course against a map or doing aerobatic maneuvers. Get a little more flight time under your belt, the sky's the limit.

Colormaster—Test your powers of deduction as you try to guess the secret color code in this Mastermind-type game. There are two levels of difficulty, and three options of play to vary your games. Not only can you guess the computer's color code, but it will guess yours! It can also serve as referee in a game between two human opponents. Can you make and break the color code...?

Star Ship Attack—Your mission is to protect our orbiting food station satellites from destruction by an enemy star ship. You must capture, destroy or drive off the attacking ship. If you fail, our planet is doomed...

Trilogy—This contest has its origins in the simple game of tic-tac-toe. The object of the game is to place three of your colors, in a row, into the delta-like, multi-level display. The rows may be horizontal, vertical, diagonal and wrapped around, through the "third dimension". Your Apple will be trying to do the same. You can even have your Apple play against itself!

Minimum system requirements are an Apple II or Apple II Plus computer with 32K of memory and one minidisk drive. Mimic requires Applesoft in ROM, all others run in RAM or ROM Applesoft.

Order No. 0161AD $19.95

Solar Energy For The Home

With the price of fossil fuels rising astronomically, solar space-heating systems are starting to become very attractive. But is solar heat cost-effective for you? This program can answer that question.

Just input this data for your home: location, size, interior details and amount of window space. It will then calculate your current heat loss and the amount of gain from any south facing windows. Then, enter the data for the contemplated solar heating installation. The program will compute the NET heating gain, the cost of conventional fuels vs. solar heat, and the calculated payback period—showing if the investment will save you money.

Solar Energy for the Home: It's a natural for architects, designers, contractors, homeowners...anyone who wants to tap the limitless energy of our sun.

Minimum system requirements are an Apple II or Apple II Plus with one disk drive and 28K of RAM. Includes AppleDOS 3.2.

Order No. 0235AD (disk-based version) $34.95

Math Fun

The Math Fun package uses the techniques of immediate feedback and positive reinforcement so that students can improve their math skills while playing these games:

Hanging—A little man is walking up the steps to the hangman's noose. But YOU can save him by answering the decimal math problems posed by the computer. Correct answers will move the man down the steps and cheat the hangman.

Spellbinder—You are a magician battling a computerized wizard. In order to cast death clouds, fireballs and other magic spells on him, you must correctly answer problems involving fractions.

Whole Space—Pilot your space craft to attack the enemy planet. Each time you give a correct answer to the whole number problems, you can move your ship or fire. But for every wrong answer, the enemy gets a chance to fire at you.

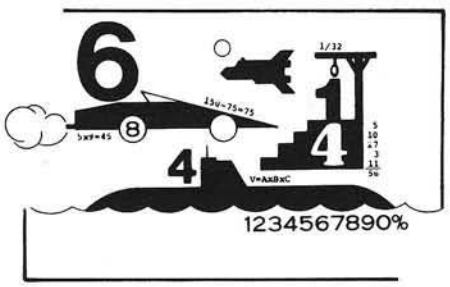
Car Jump—Make your stunt car jump the ramps. Each correct answer will increase the number of buses your car must jump over. These problems involve calculating the areas of different geometric figures.

Robot Duel—Fire your laser at the computer's robot. If you give the correct answer to problems on calculating volumes, your robot can shoot at his opponent. If you give the wrong answer, your shield power will be depleted and the computer's robot can shoot at yours.

Sub Attack—Practice using percentages as you maneuver your sub into the harbor. A correct answer lets you move your sub and fire at the enemy fleet.

All of these programs run in Applesoft BASIC, except Whole Space, which requires Integer BASIC.

Order No. 0160AD $19.95



Paddle Fun

This new Apple disk package requires a steady eye and a quick hand at the game paddles! It includes:

Invaders—You must destroy an invading fleet of 55 flying saucers while dodging the carpet of bombs they drop. Your bomb shelters will help you—for a while. Our version of a well known arcade game! Requires Applesoft in ROM.

Howitzer—This is a one or two person game in which you must fire upon another howitzer position. This program is written in HIGH-RESOLUTION graphics using different terrain and wind conditions each round to make this a demanding game. The difficulty level can be altered to suit the ability of the players. Requires Applesoft in ROM.

Space Wars—This program has three parts: (1) Two flying saucers meet in laser combat—for two players, (2) two saucers compete to see which can shoot out the most stars—for two players, and (3) one saucer shoots the stars in order to get a higher rank—for one player only. Requires Applesoft.

Golf—Whether you win or lose, you're bound to have fun on our 18 hole Apple golf course. Choose your club and your direction and hope to avoid the sandtraps. Losing too many strokes in the water hazards? You can always increase your handicap. Get off the tee and onto the green with Apple Golf. Requires Applesoft.

The minimum system requirement for this package is an Apple II or Apple II Plus computer with 32K of memory and one minidisk drive.

Order No. 0163AD $19.95

Skybombers

Two nations, seperated by The Big Green Mountain, are in mortal combat! Because of the terrain, their's is an aerial war—a war of SKYBOMBERS!

In this two-player game, you and your opponent command opposing fleets of fighter-bombers armed with bombs and missiles. Your orders? Fly over the mountain and bomb the enemy blockhouse into dust!

Flying a bombing mission over that innocent looking mountain is no milk run. The opposition's aircraft can fire missiles at you or you may even be destroyed by the bombs as they drop. Desperate pilots may even ram your plane or plunge into your blockhouse, suicidally.

Flight personnel are sometimes forced to parachute from badly damaged aircraft. As they float helplessly to earth, they become targets for enemy missiles.

The greater the damage you deal to your enemy, the higher your score, which is constantly updated at the bottom of the display screen.

The sounds of battle, from exploding bombs to the pathetic screams from wounded parachutists, remind each micro-commander of his bounden duty. Press On, SKYBOMBERS—Press On!

Minimum system requirements: An Apple II or Apple II Plus, with 32K RAM, one disk drive and game paddles.

Order No. 0271AD (disk-based version) $19.95

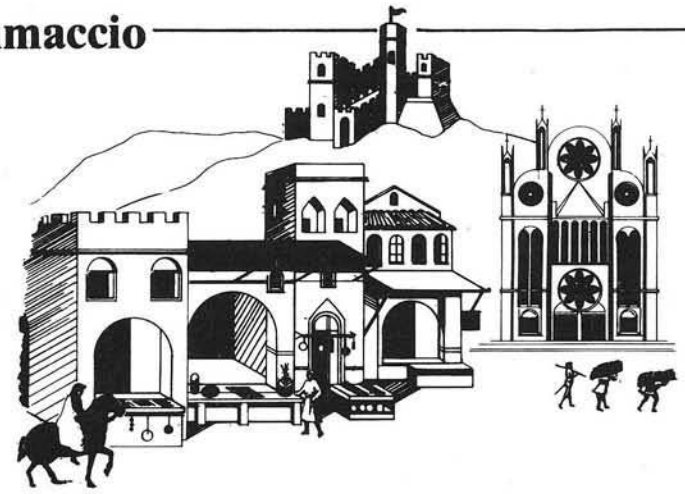
Instant Software™

*A trademark of Apple Computer Inc.

PETERBOROUGH, N.H. 03458
603-924-7296

Apple* Software
From Instant Software

Santa Paravia and Fiumaccio



Buon giorno, signore!

Welcome to the province of Santa Paravia. As your steward, I hope you will enjoy your reign here. I feel sure that you will find it, shall we say, profitable.

Perhaps I should acquaint you with our little domain. It is not a wealthy area, signore, but riches and glory are possible for one who is aware of political realities. These realities include your serfs. They constantly request more food from your grain reserves, grain that could be sold instead for gold florins. And should your justice become a trifle harsh, they will flee to other lands.

Yet another concern is the weather. If it is good, so is the harvest. But the rats may eat much of our surplus and we have had years of drought when famine threatened our population.

Certainly, the administration of a growing city-state will require tax revenues. And where better to gather such funds than the local marketplaces and mills? You may find it necessary to increase custom duties or tax the incomes of the merchants and nobles. Whatever you do, there will be far-reaching consequences...and, perhaps, an elevation of your noble title.

Your standing will surely be enhanced by building a new palace or a magnificent *cattedrale*. You will do well to increase your landholdings, if you also equip a few units of soldiers. There is, alas, no small need for soldiery here, for the unscrupulous Baron Peppone may invade you at any time.

To measure your progress, the official cartographer will draw you a *mappa*. From it, you can see how much land you hold. how much of it is under the plow and how adequate your defenses are. We are unique in that here, the map IS the territory.

I trust that I have been of help, signore. I look forward to the day when I may address you as His Royal Highness, King of Santa Paravia. *Buona fortuna* or, as you say, "Good luck". For the Apple 48K.

Order No. 0174A $9.95 (cassette version).
Order No. 0229AD $19.95 (disk version).

TO ORDER

SEE YOUR LOCAL INSTANT SOFTWARE DEALER OR USE THE ORDER FORM BELOW

For Fast Service *call now* Toll-Free 1-800-258-5473

Apple Cassettes

0018A Golf.....$7.95
0025A Mimic.....$7.95
0040A Bowling/Trilogy.....$7.95
0073A Math Tutor I.....$7.95
0079A Oil Tycoon.....$9.95
0080A Sahara Warriors.....$7.95
0088A Accounting Assistant.....$7.95
0094A Mortgage w/Prepayment Option/ Financier.....$7.95
0096A Space Wars.....$7.95
0098A Math Tutor II.....$7.95
0174A Santa Paravia and Fiumaccio.....$9.95
0148A Air Flight Simulation.....$9.95

We Guarantee It!

Instant Software Guarantee

OUR PROGRAMS ARE GUARANTEED TO BE QUALITY PRODUCTS. IF NOT COMPLETELY SATISFIED YOU MAY RETURN THE PROGRAM WITHIN 60 DAYS. A CREDIT OR REPLACEMENT WILL BE WILLINGLY GIVEN FOR ANY REASON.

109

Name _____

Address _____

City _____ State _____ Zip _____

☐ Check ☐ Money Order ☐ VISA ☐ AMEX ☐ Master Charge

Card No. _____ Exp. Date _____

Signed _____ Date _____

Order your Instant Software today!

Quantity	Order No.	Program name	Unit cost	Total cost
		Shipping and handling		$1.00
			Total order	

Instant Software™ Inc.
Peterborough, N.H. 03458

WHAT'S NEW?

Consumer Computers announces it's lowest prices ever.

Apple Add-Ons

Language System w/Pascal	379
Hayes Micromodem II	299
Novation Apple-Cat II	339
Videx Videoterm 80 w/graphics	269
Z-80 Microsoft Card	269
16K Ram Card Microsoft	159
ABT Numeric Keypad (old or new kybrd)	110
ALF 3 Voice Music Card	239
Heuristics Speechlink 2000	249
Alpha Syntauri Keyboard System	1399
Corvus 10 MB Hard Disk	CALL
Lazer Lower Case Plus	50
Micro-Sci Disk Drives (A40 & A70)	CALL
55M AIO Serial/Parallel Card A&T	189
ThunderClock Plus	CALL
Integer or Applesoft II Firmware Card	145
Graphics Tablet	619
Parallel Printer Interface Card	135
Hi-Speed Serial Interface Card	135
Smarterm 80 Column Video Card	299
MusicSystem (16 voices)	479
A/D + D/A Interface	289
Clock/Calendar Card	225
Supertalker SD-200	239
Romplus + Card	135
Clock/Calendar Module (CCS)	99
Asynchronous Serial Interface Card (CCS)	129
We carry all CCS hardware	Please Call

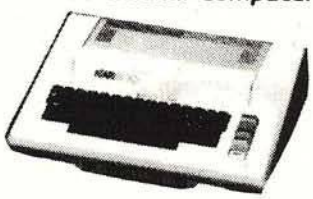
FREE CATALOG!

Please mail us your name and address.

Solution Software for the Apple II.

Visicalc 3.3	169
CCA Data Management	85
DB Master	169
WordStar (Apple 80 col. ver.)	299
Desktop Plan II	169
Applewriter	65
Easywriter	225
Appleplot	60
Peachtree Business	1200
VisiTerm	129
VisiTrend/VisiPlot	219
DOS Toolkit	65
Dow Jones Portfolio Evaluator	45
Dow Jones News & Quotes Reporter	85
Apple Fortran	165

Atari Personal Computer



ATARI 800 16K $749

Atari Accessories

410 Program Recorder	59
810 Disk Drive	449
850 Interface Module	159
16K Ram Memory Module	89

Please call us for an Atari Software List.

S-100 Cards

SSM	Kit	Assm & Tested
Z-80 CPU CB-2	219	279
104 2 P + 25 IO	189	249
VB-3 80 x 24 VIDEO	529	379
VB-3 80 x 48 VIDEO	569	429
BV-2 VIDEO	159	219
PB1 PROM PRGMMR & EPROM BOARD	139	199

CCS		
280 CPU 2810	n/a	229
64K Dynamic RAM card	n/a	499
16K Static RAM 2 MHz	n/a	249
16K Static RAM 4 MHz	n/a	269
Dbl. Density Floppy Disc Controller	n/a	299
12 Slot S-100 Mainframe	n/a	349

Apple II Plus Computer

16K $1025 48K $1129

Disk II Drive w/controller **$519**

Disk drive w/out controller 439

We carry the Apple III

CALL FREE!

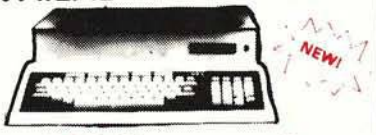
800-854-6654

In California and outside continental U.S.

(714) 698-8088

Warehouse (714) 698-0260
Service (714) 460-6502
TELEX 695000 BETA CCMO

AVAILABLE NOW . . .



The NEC Microcomputer

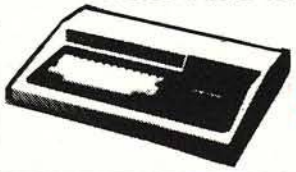
Please call for more details.

Exidy Sorcerer II

Please call for our price.

S-100 Unit	449
Word Processing Pac	179
Development Pac	89

PMC-80 Micro Computer



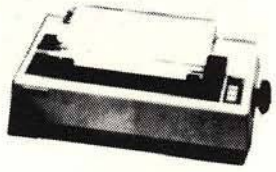
$599

Ohio Scientific Challenger Computer.

4P **$549**

1P MOD II 399

Printers, Etc.

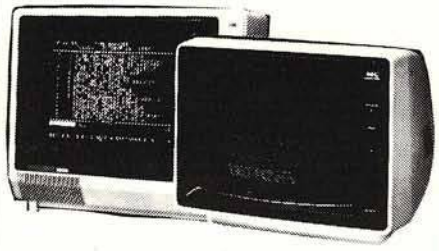


Epson MX-80

$499

Silentype w/Apple Interface	349
Epson MX-80 F/T	649
Epson MX-100	829
Epson MX-70	339
Paper Tiger IDS-445	729
Paper Tiger IDS-460	1099
Paper Tiger IDS-560	1450
Qume Sprint 5/45	2495
Anadex DP-9500/ w/2K Buffer	1349
C. Itoh Starwriter 25 CPS	1649
C. Itoh Starwriter 45 CPS	2249
Centronics 737	699
Watanabe DigiPlot	1149

Save on Video displays.



Amdek/Leedex Video 100 12" B&W	159
Amdek (Hitachi) 13" Color	359
NEC 12" P31 Green Phosphor	CALL
NEC 12" RGB Hi-Res Color	CALL
Panasonic 13" Color	449
Sanyo 9" B&W	159
Sanyo 12" B&W	239
Sanyo 12" P31 Green Phosphor	279
Sanyo 13" Color	419

We carry much more than listed. Please call our toll free order line to request our complete price list.

How to Order

Ordering information: Phone orders using VISA, MASTERCARD, AMERICAN EXPRESS, DINER'S CLUB, CARTE BLANCHE, bank wire transfer, cashiers or certified check, money order, or personal check (allow ten days to clear.) Unless prepaid with cash, please add 5% for shipping, handling and insurance. (minimum $5.00). California residents add 6% sales tax. We accept CODs, OEM's, Institutions and corporations please send for a written quotation. All equipment is subject to price change and availability without notice. All equipment is new and complete with manufacturer's warranty (usually 90 days). Showroom prices may differ from mail order prices.

Send orders to:

consumer computers

Mail Order

8314 Parkway Drive
La Mesa, California 92041

Journal It: Screen Print Utility for Atari

This program describes a versatile utility for the Atari. It will 'capture' all screen text I/O, including user-program generated data and system prompts or responses, and then 'journal' it on a printer. The journal will provide the user with a hard copy of a session's transactions, for logging or debugging purposes.

John Elliott
33 Gold Street, Apt. #708
New York, New York 10038

The program described in this article is a screen print utility for the Atari 400/800, and it will be of interest to both BASIC and assembly language programmers. Readers not familiar with assembly language will want to start with the section, "Using the Program," which describes how to use the program with BASIC.

My need for a program to print screen text arose when I began using the Atari Assembler Editor cartridge. This cartridge contains three separate programs:

1. The EDITOR program allows you to create and maintain assembler language source files.
2. The ASSEMBLER program converts the source code to machine language.
3. The DEBUG program provides many program testing aids, such as single-stepping through a program and/or tracing its execution path. DEBUG will also disassemble machine code, display the contents of memory, etc.

Developing programs in assembly language can be difficult and time-consuming. Therefore a good debug package is a welcome addition to the assembly language programmer's systems software library. I found the Atari DEBUG program to be very useful, but I was disappointed to find that output from the debugger could not be directed to a printer.

Fortunately, the Atari I/O subsystem is very flexible, and a straightforward 'patch' to the system I/O tables can be used to reroute data from one device to another. Specifically, it is not difficult to 'trap' data directed to the screen through the screen-editor, and then redirect it to a printer.

The program in listing 1 is a logical extension of this idea. However, instead of *rerouting* data directed to the screen,

Figure 1: **Normal Screen Editor Data Flow.**

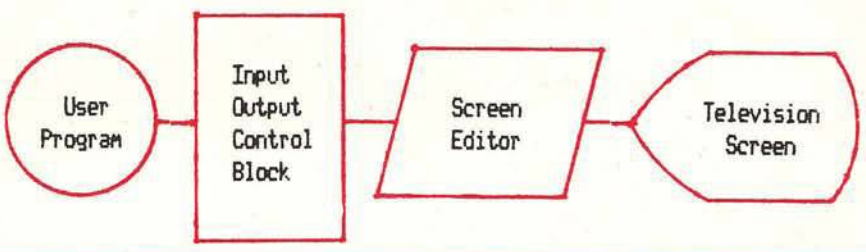
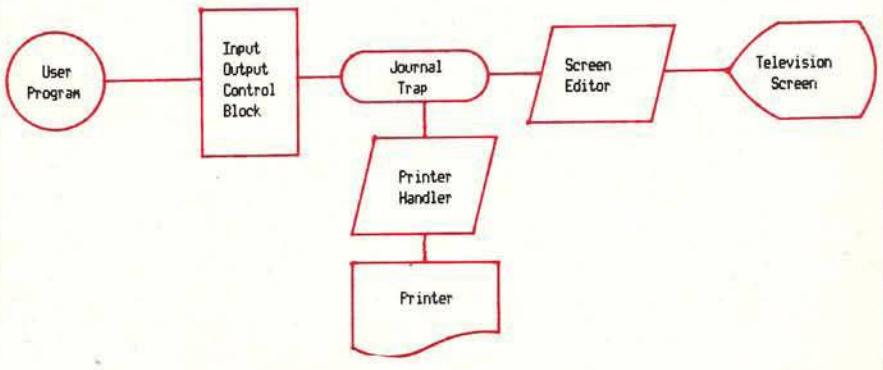


Figure 2: **Data Flow with Journal Trap Installed.**



Listing 1: **Screen Journal**

```
0000          0120          .PAGE
              0130 ;++++++++++++++++++++++++++++++++
              0140 ;+     ASSEMBLER ROUTINE TO     +
              0150 ;+  TRAP SCREEN INPUT/OUTPUT    +
              0160 ;+ &  JOURNAL IT ON A PRINTER   +
              0170 ;+    JOHN ELLIOTT    MAY '81   +
              0180 ;++++++++++++++++++++++++++++++++
0008          0190 WARMST =     $0008      ;WARM START FLAG
031A          0200 HATABS =     $031A      ;DEVICE HANDLERS
              0210 ;
```

(Continued)

the program will *duplicate* it on a printer. That is, the data will appear on both the screen and the printer. Additionally, the program will trap data coming from the screen editor, and copy that on a printer. The result is a 'journal' of all screen activity.

Program Description

To understand how the program works, we need to look at how the Atari talks to the outside world. The Atari OS (operating system) includes an I/O subsystem for communicating with the system I/O devices, such as disks, cassette drives, printers, etc. The subsystem is composed of resident 'handlers,' one per device. In the normal environment the user program will communicate with the handlers via an IOCB (input/output control block). Typical data flow is illustrated in figure 1.

In addition to providing a means of passing parameters, such as data addresses and request codes, the IOCB contains pointers to the handler routines. Since the IOCBs are in RAM these pointers can be changed to address custom routines, and that is exactly what the journal program does. The handler pointers in the screen editor IOCB (IOCB # zero) are changed to point to the I/O trap routines. The result is that all data coming from, or going to, the screen through the screen editor (not the display handler) will be intercepted. These routines then call the printer handler to duplicate all screen I/O. The data flow is illustrated in figure 2.

Besides intercepting screen I/O, the program performs a certain amount of control code translation. This is because the screen editor recognizes sixteen control codes, most of which are treated as unprintable characters by the printer, and will appear as spaces on a printout. The translation is as follows:

1. the "Clear Screen" code ($7D) is translated to an EOL (end of line) character ($9B);
2. the "Insert Line" code ($9D) is translated to an EOL character ($9B);
3. all codes and characters with a hexadecimal value of less than $20 are ignored.

This is by no means a comprehensive translation, but I have found it to be satisfactory for journaling most BASIC, Assembler Editor, and DOS transactions.

Listing 1 *(Continued)*

```
                    0220 ;INPUT/OUTPUT CONTROL BLOCKS
                    0230 ;
0000                0240 IOCB    *=    $0340      ;I/O CONTROL BLOCKS
0340                0250 ICHID   *=    *+1        ;HANDLER ID
0341                0260 ICDNO   *=    *+1        ;DEVICE NUMBER
0342                0270 ICCOM   *=    *+1        ;COMMAND CODE
0343                0280 ICSTA   *=    *+1        ;STATUS
0344                0290 ICBAL   *=    *+1        ;BUFFER ADDRESS LSB
0345                0300 ICBAH   *=    *+1        ;BUFFER ADDRESS MSB
0346                0310 ICPUT   *=    *+2        ;PUT ROUTINE ADDRESS
0348                0320 ICBLL   *=    *+1        ;BUFFER LENGTH LSB
0349                0330 ICBLH   *=    *+1        ;BUFFER LENGTH MSB
034A                0340 ICAX1   *=    *+1        ;AUX CONTROL BYTE 1
034B                0350 ICAX2   *=    *+1        ;AUX CONTROL BYTE 2
                    0360 ;
                    0370 ;ENVIRONMENT ADDRESSES/EQUATES
                    0380 ;
BFFA                0390 CSTART  =     $BFFA      ;CARTRIDGE START
007D                0400 CLEAR   =     $7D        ;CLEAR SCREEN CODE
0020                0410 SPACE   =     $20        ;FIRST PRINTABLE CHAR
009C                0420 DELETE  =     $9C        ;DELETE-LINE CODE
009B                0430 EOL     =     $9B        ;END OF LINE CODE
                    0440 ;
034C                0450         *=    $E400      ;START OF VECTOR TABLES
                    0460 ;
                    0470 ;EDITOR VECTOR TABLE
                    0480 ;
E400                0490 EDITRV  =     *          ;<E:> VECTOR TABLE
E400                0500 EDOPEN  *=    *+2        ;OPEN VECTOR
E402                0510 EDCLOS  *=    *+2        ;CLOSE VECTOR
E404                0520 EDGETB  *=    *+2        ;GETCHAR VECTOR
E406                0530 EDPUTB  *=    *+2        ;PUTCHAR VECTOR
E408                0540 EDSTAT  *=    *+2        ;STATUS VECTOR
E40A                0550 EDSPEC  *=    *+2        ;SPECIAL VECTOR
E40C                0560 EDINIT  *=    *+3        ;INITIALIZATION
E40F                0570         *=    *+1        ;SPARE
                    0580 ;
E410                0590 SCRENV  *=    *+16       ;S: VECTOR TABLE
E420                0600 KEYBDV  *=    *+16       ;K:VECTOR TABLE
                    0610 ;
                    0620 ;PRINTER VECTOR TABLE
                    0630 ;
E430                0640 PRINTV  =     *          ;<P:> VECTOR TABLE
E430                0650 PROPEN  *=    *+2        ;OPEN VECTOR
E432                0660 PRCLOS  *=    *+2        ;CLOSE VECTOR
E434                0670 PRGETB  *=    *+2        ;GETCHAR VECTOR
E436                0680 PRPUTB  *=    *+2        ;PUTCHAR VECTOR
E438                0690 PRSTAT  *=    *+2        ;STATUS VECTOR
E43A                0700 PRSPEC  *=    *+2        ;SPECIAL VECTOR
E43C                0710 PRINIT  *=    *+3        ;INITIALIZATION
E43F                0720         *=    *+1        ;SPARE
E440                0730         .PAGE
0600                0740 BEGIN   =     $0600      ;PROGRAM ORIGIN
E440                0750         *=    BEGIN
                    0760 ;
                    0770 ;COPY <E:> VECTOR TABLE
                    0780 ;
0600 A00F           0790         LDY   #15        ;LENGTH OF VECTOR TABLE
0602                0800 COPVEC  =     *          ;COPY <E:> VECTOR TABLE
0602 B900E4         0810         LDA   EDITRV,Y
0605 998306         0820         STA   HAVECS,Y
0608 88             0830         DEY
0609 10F7           0840         BPL   COPVEC     ;LOOP 'TIL ALL COPIED
                    0850 ;
                    0860 ;CHANGE <E:> HANDLER VECTOR
                    0870 ; TO POINT TO OUR TABLE
                    0880 ;
060B A983           0890         LDA   #HAVECS&$FF
060D 8D2103         0900         STA   HATABS+7   ;<E:> VECTOR LSB
0610 A906           0910         LDA   #HAVECS/256
0612 8D2203         0920         STA   HATABS+8   ;<E:> VECTOR MSB
                    0930 ;
                    0940 ;POINT TO OUR PUTCHAR ROUTINE
                    0950 ;
0615 A94A           0960         LDA   #OUTPUT-1&$FF
0617 8D8906         0970         STA   HAPUTB     ;HANDLER TABLE...
061A 8D4603         0980         STA   ICPUT      ;...AND IOCB
061D A906           0990         LDA   #OUTPUT-1/256
061F 8D8A06         1000         STA   HAPUTB+1
0622 8D4703         1010         STA   ICPUT+1
                    1020 ;
                    1030 ;POINT TO OUR GETCHAR ROUTINE
                    1040 ;
```

(Continued)

Listing 1 *(Continued)*

```
0625 A933   1050        LDA  #INPUT-1&$FF
0627 8D8706 1060        STA  HAGETB    ;HANDLER TABLE ONLY
062A A906   1070        LDA  #INPUT-1/256
062C 8D8806 1080        STA  HAGETB+1
            1090 ;
            1100 ;VECTORS ESTABLISHED - NOW EXIT
            1110 ;
062F 8508   1120        STA  WARMST    ;SET WARMSTART FLAG
0631 6CFABF 1130        JMP  (CSTART)  ;CARTRIDGE START
            1140 ;
            1150 ;
            1160 ;MIRROR SCREEN INPUT TO PRINTER
            1170 ;
0634        1180 INPUT  =    *
0634 8D9206 1190        STA  SAVEA     ;SAVE ACCUMULATOR
0637 203F06 1200        JSR  EGET      ;GO GET SCREEN INPUT
063A 204006 1210        JSR  PPUT      ;JOURNAL INPUT ON PRINTER
063D D017   1220        BNE  EXIT      ;BACK TO CALLER
            1230 ;
            1240 ;GO GET SCREEN INPUT
            1250 ;
063F        1260 EGET   =    *
063F AD05E4 1270        LDA  EDGETB+1  ;HI BYTE FIRST
0642 48     1280        PHA
0643 AD04E4 1290        LDA  EDGETB    ;LO BYTE LAST
0646 48     1300        PHA
0647 AD9206 1310        LDA  SAVEA
064A 60     1320        RTS            ;GO GET THE SCREEN INPUT
064B        1330        .PAGE
            1340 ;
            1350 ;MIRROR SCREEN OUTPUT TO PRINTER
            1360 ;
064B        1370 OUTPUT =    *
064B 204006 1380        JSR  PPUT      ;PUT CHARACTER TO <P:>
            1400 ;PUT CHAR TO <E:>
            1410 ;
064E        1420 EPUT   =    *
064E AD07E4 1430        LDA  EDPUTB+1  ;HI BYTE FIRST
0651 48     1440        PHA
0652 AD06E4 1450        LDA  EDPUTB    ;LO BYTE LAST
0655 48     1460        PHA
0656        1470 EXIT   =    *         ;COMMON EXIT POINT
0656 AD9206 1480        LDA  SAVEA     ;RESTORE ACCUMULATOR
0659 AE9306 1490        LDX  SAVEX     ;RESTORE REGISTER  X
065C AC9406 1500        LDY  SAVEY     ;RESTORE REGISTER  Y
065F 60     1510        RTS            ;PUT CHAR TO SCREEN/EXIT
            1520 ;
            1530 ;PUT CHAR TO <P:>
0660        1550 PPUT   =    *
0660 8D9206 1560        STA  SAVEA     ;SAVE ACCUMULATOR
0663 8E9306 1570        STX  SAVEX     ;SAVE REGISTER  X
0666 8C9406 1580        STY  SAVEY     ;SAVE REGISTER  Y
0669 C920   1590        CMP  #SPACE    ;PRINTABLE CHAR?
066B 9015   1600        BCC  PCALL     ;NO...IGNORE IT
066D AD37E4 1610        LDA  PRPUTB+1  ;HI BYTE FIRST
0670 48     1620        PHA
0671 AD36E4 1630        LDA  PRPUTB    ;LO BYTE SECOND
0674 48     1640        PHA
0675 AD9206 1650        LDA  SAVEA     ;PICKUP CHARACTER...
0678 C97D   1660        CMP  #CLEAR    ;CLEAR SCREEN?
067A F004   1670        BEQ  PSUB      ;YES...SUBSTITUTE EOL
067C C99C   1680        CMP  #DELETE   ;DELETE LINE?
067E D002   1690        BNE  PCALL     ;NO...PRINT IT
0680        1700 PSUB   =    *         ;SUBSTITUTE EOL
0680 A99B   1710        LDA  #EOL      ;SUBSTITUTE EOL
0682        1720 PCALL  =    *
0682 60     1730        RTS            ;...AND PRINT IT
            1750 ;MODIFIED <E:> VECTOR TABLE
            1760 ;
0683        1770 HAVECS =    *         ;HANDLER VECTORS
0683        1780 HAOPEN *=   *+2       ;OPEN VECTOR
0685        1790 HACLOS *=   *+2       ;CLOSE VECTOR
0687        1800 HAGETB *=   *+2       ;GETCHAR VECTOR
0689        1810 HAPUTB *=   *+2       ;PUTCHAR VECTOR
068B        1820 HASTAT *=   *+2       ;STATUS VECTOR
068D        1830 HASPEC *=   *+2       ;SPECIAL VECTOR
068F        1840 HAINIT *=   *+3       ;INITIALIZATION
            1860 ;REGISTER SAVE AREAS
            1870 ;
0692        1880 SAVEA  *=   *+1       ;ACCUM SAVE AREA
0693        1890 SAVEX  *=   *+1       ;REG X SAVE AREA
0694        1900 SAVEY  *=   *+1       ;REG Y SAVE AREA
0695        1910        .END
```

Program Environment

The program is designed to work equally well with BASIC or the Assembler Editor cartridge, with or without DOS. It does not conflict with any of the BASIC or Editor functions, nor does it impose any limitations on application programs. It does not use any of the IOCBs (other than the screen editor's), therefore no program changes will be required before it can be used.

Other than the system I/O tables, the program's use of RAM is limited exclusively to page 6; no page zero locations are used. Page 6 is not used by BASIC or any other Atari system software, and so is available for our use. Once loaded into RAM, the program will remain intact until removed by the user. None of the system programs will encroach upon it. Conversely, the program does not encroach upon the RAM allocated to the system programs. Therefore it does not have an impact on memory availability, and will run on the minimum 8K system configuration.

The program is compatible with all the standard Atari software. However, it uses the in-ROM system vectors to call the printer handler. If you have installed a non-resident printer handler, you will need to modify the journal program to conform to the interfacing conventions of your handler.

So far as hardware is concerned, the program will work with either the Atari 820 or 825 printer. It should also work with the Atari 822 Thermal Printer, although I have not been able to verify this.

Using the Program

The program is an assembly language routine, written to reside on page 6 of RAM. Those of you who have the Assembler should assemble the program shown in listing 1. For those of you who do not have the Assembler, I include a BASIC program (listing 2) which will POKE the machine code into RAM. (Listing 3 shows a sample RUN of this program.) If you use the BASIC program, be sure that you enter the DATA statements exactly as shown, otherwise the system might crash when you use the journal program.

When you have loaded the program into RAM, you should execute it whenever you wish to initiate journaling. Before doing so, be sure that the printer is connected and powered on. If you are using BASIC, execute the pro-

Listing 2: BASIC program to set up Screen Journal Program.

```
10 REM ++++++++++++++++++++++++++++++++
11 REM +                              +
12 REM +   BASIC PROGRAM TO SETUP     +
13 REM +  MACHINE LANGUAGE ROUTINE    +
14 REM +          THAT  WILL          +
15 REM + TRAP SCREEN INPUT / OUTPUT   +
16 REM + &  JOURNAL IT TO A PRINTER   +
17 REM + JOHN ELLIOTT        MAY '81  +
18 REM +                              +
19 REM ++++++++++++++++++++++++++++++++
20 REM
22 DIM HEX$(3)
24 MLPROG=1536:REM PAGE 6 ADDRESS
26 GRAPHICS 0
28 POKE 752,1:REM CURSOR OFF
30 ? :? "LOADING JOURNAL":?
35 REM
36 REM ++++++++++++++++++++++++++++++++
37 REM + SETUP MACHINE LANGUAGE PGM +
38 REM ++++++++++++++++++++++++++++++++
39 REM
40 FOR ADDR=MLPROG TO MLPROG+1024
45 READ HEX$:SOUND 0,ADDR-MLPROG,10,4
50 IF HEX$="END" THEN POP :GOTO 80
55 HI=ASC(HEX$(1,1))-48
60 IF HI>9 THEN HI=HI-7
65 LO=ASC(HEX$(2,2))-48
70 IF LO>9 THEN LO=LO-7
75 POKE ADDR,HI*16+LO:NEXT ADDR
80 REM
81 REM ++++++++++++++++++++++++++++++++
82 REM +    DISPLAY INSTRUCTIONS     +
83 REM ++++++++++++++++++++++++++++++++
84 REM
85 ? "JOURNAL LOADED":? :POKE 752,0
90 ? "TO INITIATE JOURNALLING TYPE :"
95 ? :? "X=USR(";MLPROG;")"
100 REM
101 REM ++++++++++++++++++++++++++++++++
102 REM +  HEXADECIMAL M/L PROGRAM  +
103 REM ++++++++++++++++++++++++++++++++
104 REM
105 DATA A0,0F,B9,00,E4,99,83,06
110 DATA 88,10,F7,A9,83,8D,21,03
115 DATA A9,06,8D,22,03,A9,4A
120 DATA 8D,89,06,8D,46,03,A9,06
125 DATA 8D,8A,06,8D,47,03,A9,33
130 DATA 8D,87,06,A9,06,8D,88,06
135 DATA 85,08,6C,FA,BF,8D,92,06
140 DATA 20,3F,06,20,60,06,D0,17
145 DATA AD,05,E4,48,AD,04,E4,48
150 DATA AD,92,06,60,20,60,06
155 DATA AD,07,E4,48,AD,06,E4,48
160 DATA AD,92,06,AE,93,06
165 DATA AC,94,06,60,8D,92,06
170 DATA 8E,93,06,8C,94,06,C9,20
175 DATA 90,15,AD,37,E4,48,AD,36,E4
180 DATA 48,AD,92,06,C9,7D,F0,04
185 DATA C9,9C,D0,02,A9,9B,60
190 DATA END
```

gram, using the command X=USR (1536), as shown in listing 3. If you are using the Assembler Editor cartridge, use the DEBUGGER's GO (execute program) command, as shown in listing 4. Those of you who have DOS may use the RUN AT ADDRESS option to execute the program, as shown in listing 5. Note that both the DEBUGGER and DOS expect the address to be given in hexadecimal, whereas BASIC expects the address in decimal.

To terminate journaling, press the SYSTEM RESET button on the console. The program will remain intact in RAM, and can be re-executed whenever you wish to re-initiate journaling. Turning the printer power off will, of course, terminate journaling. However, this is not recommended, since the trap program will continue to try to journal screen I/O, and will succeed only in wasting time.

These operating instructions are intended only as guidelines. There are many other ways the program can be invoked and you should use the procedure which best suits your own operating environment.

Conclusion

I believe you will find that the more you use this program, the more uses you will find for it. In addition to providing a journal of screen transactions, the program can be used to provide printed output from in-ROM programs which you cannot change, as in the case of the DEBUG program. Furthermore, you need never again go through your own programs, changing all the PRINT statements to LPRINT statements when you want the output to go to a printer. Another interesting feature is that you can communicate with your Atari using

Listing 3: Sample run of program in listing 2.

```
READY
RUN

LOADING JOURNAL

JOURNAL LOADED

TO INITIATE JOURNALLING
  TYPE :

X=USR(1536)

READY
```

the printer instead of the TV screen: turn off the TV and talk to the computer through the printer. Or, you can treat your Atari as an intelligent typewriter, using the screen editing features to correct errors.

In short, there are many possible applications for this program, and I'm sure you will find even more uses than are listed here. I will be pleased to hear from those of you who do.

Listing 4: **Example of using the debugger to initiate journalling.**

```
EDIT
BUG

DEBUG
G600

EDIT
```

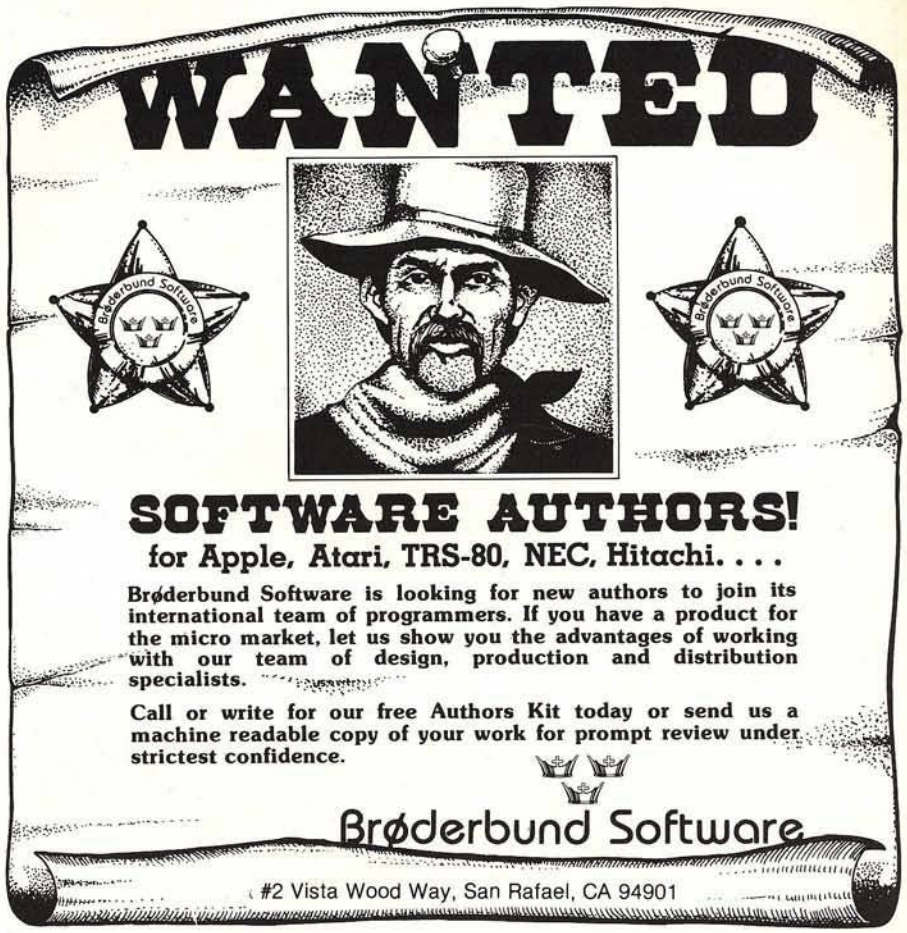
Listing 5: **Example of using DOS to initiate journalling.**

```
READY
DOS

DISK OPERATING SYSTEM    9/24/79
COPYRIGHT 1979 ATARI

A. DISK DIRECTORY
B. RUN CARTRIDGE
C. COPY FILE
D. DELETE FILE(S)
E. RENAME FILE
F. LOCK FILE
G. UNLOCK FILE
H. WRITE DOS FILE
I. FORMAT DISK
J. DUPLICATE DISK
K. BINARY SAVE
L. BINARY LOAD
M. RUN AT ADDRESS
N. DEFINE DEVICE
O. DUPLICATE FILE
SELECT ITEM
M
RUN FROM WHAT ADDRESS?
600
READY
```

John Elliott is currently a Systems Consultant for On-Line Software International. He has been in the data processing field for over twelve years, specializing in implementing telecommunications software on large computer systems. He has served as a consultant in various locations in the USA, Europe and the Middle East. He finds that tinkering with his Atari provides an interesting contrast to his work on mainframes.



80 × 25

On The Built-in Display

PET/CBM™ 2000/3000/4000 Series

$275.00 *

Select either . . . . . **80** x **25** or **40** x **25** . . . . . display format

From the keyboard or program

Features the same memory map and subroutines from the Basic ROMs [specify which version of Basic when ordering] * *, for standard programming. You also gain the use of **1023** extra memory locations in the 40 column mode, or **47** extra locations in the 80 column mode. [These locations are available only to machine language programs, or peek & poke from Basic].

Displays the full, original character set, including graphics characters in either mode.

All utility software, firmware, like Toolkit™, Dos Support [Wedge], Extra-mon, etc., is compatible in both modes of operation.

The complete enhancement consists of: 1 dual 24-pin socket [one socket for the 40 column screen editor, and one for the 80 column screen editor], and a circuit board that replaces the existing screen RAM. Each circuit board is registered to the original owner. There is also an 80 column reference ROM that plugs in one of the expansion sockets [specify the address when ordering]. An option board is available [$25.00] that allows the ROM to be used with any other 2K ROM, in any of the expansion sockets.

Available from your local dealer or:

EXECOM CORP.
1901 Polaris Ave.
Racine, WI 53404
Ph. 414-632-1004

* Plus appropriate installation charges. This requires some circuit modification. [available from the factory for $75.00 plus shipping]

* * If power-on message = ### COMMODORE BASIC ### you have 3.0 Basic. [Available only for Basic 3.0 & Basic 4.0 at the present].
PET™ & CBM are trademarks of Commodore Business Machines.
We will ship via Master Charge, VISA, C.O.D., or pre-paid.
Toolkit™ is a trademark of Palo Alto IC's, Inc.
Installation may void your Commodore 90 day warr.
The Execom™ board is guaranteed for 1 year.

MICRO

PET Vet

By Loren Wright

2114 RAM Adaptor

If you have one of the old PETs with 6550 RAM chips, you probably have had one or more of them fail by now. You probably had to pay a high price for the replacement, if, in fact, you were able to find one. Optimized Data Systems (P.O. Box 595, Placentia, CA 92670) offers an inexpensive board that plugs into a pair of empty 6550 sockets and allows substitution of up to half of the 6550s with the considerably more common and less expensive 2114 RAM chips. The board is available in a number of versions, from a bare board ($8.95) to an assembled version with one 2114 and two sockets ($24.95). The board is conveniently laid out and works well.

PET Power Problems

Owners of older PETs may have experienced various forms of flakiness, particularly when additional devices draw from the PET's power supplies. The universal advice has been simply to avoid placing additional loads on the PET supplies. The fault, however, is not in the capacity of the power supplies, but rather in the poor design of the power connection to the main logic board. The board has a male connector consisting of round pins, while the female connector has flat spring contacts — not a very good connection! Under normal operation, the whole thing heats up and may eventually cause loss of contact and a crash. With an additional draw on the supply, the critical point is reached more quickly. Jim Yost of Somerville, Massachusetts, has come up with a simple and cheap solution:

> Identify the two outside wires (usually brown) on the female power connector. These lead to two separate terminals of the power transformer. Between these two transformer terminals and the positive (+) terminal of the large electrolytic capacitor connect two rectifier diodes (3A, 100V PIV). The diode positive terminals should lead to the capacitor positive terminal. This bypasses the power surges to the capacitor. See figure 1 for details.

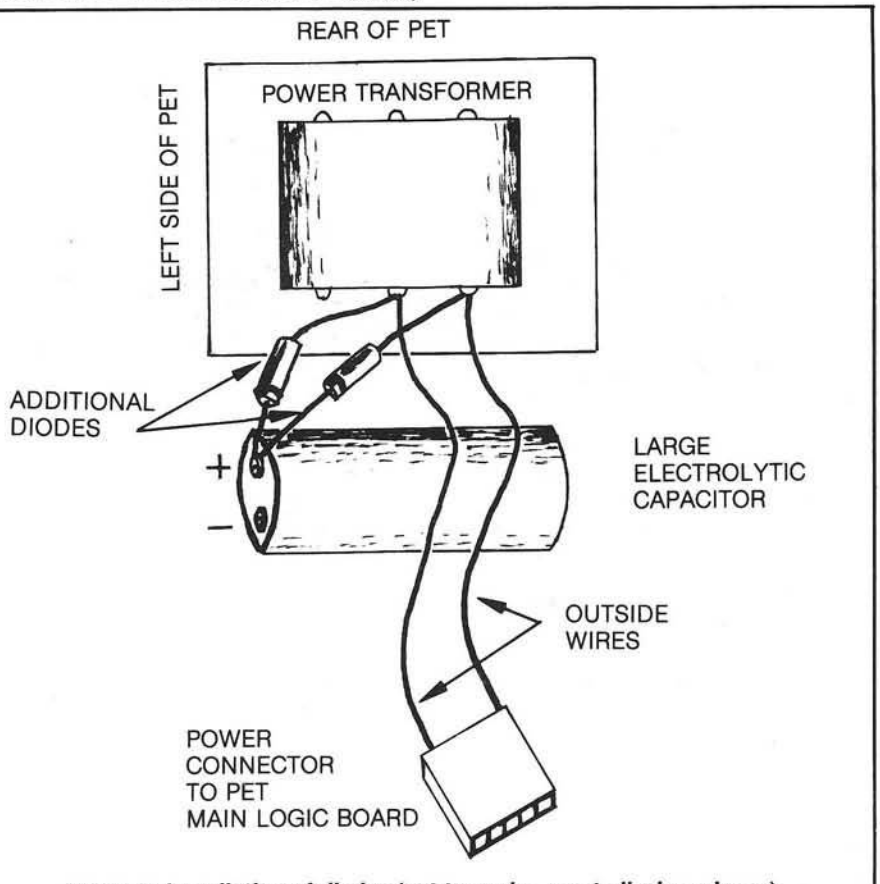


Figure 1: **Installation of diodes (not to scale — not all wires shown).**

Old PETs Live On!

Duncan Moyer of Auburn, California, points out that while a SYS 64824 on the old PET does a system reset, it fails to reset the Auxilliary Control Register to 0. To get the cassette to work properly, you must also POKE 59467,0. I have received a number of letters in support of the old PET, and I *will* treat it as a respected member of the Commodore family.

More on Command-O

During the course of writing an 8032 program to process the questionnaires included in the July MICRO, I made good use of Command-O's SCROLL and "softkey" features. The "softkey" saved me from typing lots of repetitive lines. It's nice to be able to enter with a single keystroke "X= :Y= :GOSUB5000" (followed by the appropriate number of back cursors to wind up right after the first '='.

In my review of BASIC upgrades (36:62), I casually mentioned that the FIND command of Command-O had been improved. It allows specification of a range of lines in which the search will occur. Each occurrence causes the line found to be listed with the beginning of the search string in reverse field. This means that if the same string appears three times in the same line, then that line will be listed three times.

Be careful when you make corrections to lines listed by the FIND command, though. If the found string is within quotes, then the first character will stay reversed. This is not a problem, unless you happen to hit one of the special 8032 control characters. It took me about 20 minutes one day to discover an embedded reverse 'o' as the cause of an unwanted WINDOW command!

Typesetting for the Common Man?

Type Share (8315 Firestone Blvd., Downey, CA 90241) has announced an inexpensive typesetting input system based on the Commodore VIC-20. The announced price is less than $500 including the VIC and software, but does not include the video display. Users record the input text on cassette for transfer to a larger computer for further editing/merging, or the cassette can be sent directly to one of the Type Share centers. The VIC-based system makes typesetting available to a number of small volume users, who could not otherwise afford to have typesetting done.

MICRO™

we beat the price...

800™ $779

WITH ONE YEAR EXTENDED WARRANTY

ATARI®
Computers for people.™

815 DUAL DISK DRIVE

$1199

ATARI SOFTWARE

CX401 General Accounting	$399.00
CX402 Accounts Receivable	$399.00
CX403 Inventory Control	$399.00
CX404 Word Processor	$119.00
CX405 PILOT	$68.00
CX413 MICROSOFT BASIC	$68.00
CX4101 Invitation to Programming 1	$17.00
CX4102 Kingdom	$13.00
CX4103 Statistics	$17.00
CX4104 Mailing List	$17.00
CX4105 Blackjack	$13.00
CX4106 Invitation to Programming 2	$20.00
CX4107 Biorhythm	$13.00
CX4108 Hangman	$13.00
CX4109 Graph It	$17.00
CX4110 Touch Typing	$20.00
CX4111 Space Invaders	$17.00
CX4112 States & Capitals	$13.00
CX4114 European Countries & Capitals	$13.00
CX4115 Mortgage & Loan Analysis	$13.00
CX4116 Personal Fitness Prog	$59.00
CX4117 Invitation to Programming 3	$20.00
CX4118 Conversational French	$45.00
CX4119 Conversational German	$45.00
CX4120 Conversational Spanish	$45.00
CX4121 Energy Czar	$13.00
CX4125 Conversational Italian	$45.00

CX6001 U.S. History	$23.00
CX6002 U.S. Government	$23.00
CX6003 Supervisory Skills	$23.00
CX6004 World History	$23.00
CX6005 Basic Sociology	$23.00
CX6606 Counseling Proced	$23.00
CX6607 Principal of Act	$23.00
CX6608 Physics	$23.00
CX6609 Great Classics	$23.00
CX6010 Business Comm	$23.00
CX6011 Basic Psychology	$23.00
CX6012 Effective Writing	$23.00
CX6014 Principals of Econ	$23.00
CX6015 Spelling	$23.00
CX6016 Basic Electricity	$23.00
CX6017 Basic Algebra	$23.00
CX8106 Bond Analysis	$20.00
CX8107 Stock Analysis	$20.00
CX8108 Stock Charting	$20.00
CXL4001 Education System Master	$21.00
CXL4002 Basic Computing Language	$46.00
CXL4003 Assembler Editor	$46.00
CXL4004 Basketball	$24.00
CXL4005 Video Easel	$24.00
CXL4006 Super Breakout	$30.00
CXL4007 Music Composer	$45.00
CXL4009 Chess	$30.00
CXL4010 3-D Tic-Tac-Tow	$24.00

CXL4011 Star Raiders	**$32.00**
CXL4012 Missle Command	**$32.00**
CXL4013 Asteroids	**$32.00**
CXL4015 TeleLink	$20.00
Compuhome	$74.95
Visicalc	$149.00
Letter Perfect (Word Processor)	$119.00
Source	$89.00

Atari® Peripherals:

400 16K	$329.00
410 Recorder	$59.00
810 Disk Drive	$449.00
815 Disk Drive	$1199.00
822 Printer	$359.00
825 Printer	$629.00
830 Modem	$159.00
850 Interface	$139.00

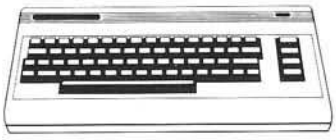
Atari® Accessories

New DOS 2 System	$21.00
CX70 Light Pen	$64.00
CX30 Paddle	$18.00
CX40 Joy Stick	$18.00
CX853 16K RAM	$89.00
Microtek 16K RAM	$75.00
Microtek 32K RAM	$169.00

commodore

CBM 8032 $1099

4016	$799.00
4032	$999.99
8096	$1795.00
CBM4022 Printer	$629.00
Tally 8024	$1699.00
CBM C2N Cassette Drive	$69.00
CBM4040 Dual Disk Drive	$999.00
CBM8050 Dual Disk Drive	$1349.00



VIC 20

$259

Vic-TV Modual	$19.00
Vic Cassette	$69.00
Vic Disk Drive	$ Call
Vic 6 Pack program	$44.00

CBM Software

WordPro3 Plus	$199.00
WordPro4 Plus	$299.00
Commodore Tax Package	$399.00
Visicalc	$149.00
EBS Accts. Rec./Inventory Interactive Syst.	$595.00
BPI General Ledger	$329.00
OZZ Information System	$329.00
Dow Jones Portfolio	$129.00
Pascal	$239.00
Legal Time Accounting	$449.00
World Craft 80	$289.00
Word Check	$180.00
Create-A-Base	$180.00
Power	$89.00
Socket-2-Me	$20.00
Jinsam	Call

Disks

CX8100 Blank Disk (5)	$22.00
Sycom Blank Disk (10)	$29.00
Maxell Blank Disk (10)	$36.00
Maxell Blank Disk (10)	$46.00

Printers

Epson MX-70 Epson MX-80 Epson MX-80 FT Diablo 630	Call for Prices
Nec 5530	$2495.00
TEC 1500 Starwriter 25cps	$1495.00
TEC 1500 Starwriter 45cps	$1995.00

No Risk, No Deposit On Phone Orders, COD or Credit Card, Shipped Same Day You Call *

* on all in stock units

IN PA. CALL: 717 - 323-7921

(800) 233-8950

COMPUTER MAIL ORDER
501 E. 3RD ST., WILLIAMSPORT, PA 17701

To Order:

Phone orders invited (800 number is for order desk only). Or send check or money order and receive free shipping. Pennsylvania residents add 6% sales tax. Add 3% for Visa or M.C. Equipment is subject to price change and availability without notice. Please call between 11 AM & 6 PM.

DTACK 10

Did you know that the Motorola 68000, **not** the 6809, is the logical successor to the 6502?

Did you know that you can attach a 68000 processor board to your Apple II or PET/CBM for about $600? That such a board is available from us?

Did you know that a four-function floating point package, with a logarithm routine, is available in microsoft Apple II/PET format? That the log routine runs over 10 times faster than in the 6502? That you can buy a copy of this package from us in **source** form for only $10 plus 50 cents postage?

Did you know that your Apple II or PET/CBM would make a very nice I/O processor for a 68000 CPU?

A newsletter covering **simple** 68000 systems and attached processors is available from us for $15/6 issues. Send $2.50 for just the first issue.

For more information, send an S.A.S.E. to:

DTACK GROUNDED
Dept. 101
1415 E. McFadden St. F
Santa Ana, California 92705

---

computer case company

comp case

This is an attache style case for carrying and protecting a complete ATARI computer system. It will hold the 800 or 400 computer, disk drive, program recorder and a small printer in a variety of combinations. Constructed of the highest quality luggage material. Will accommodate equipment in a fully operational configuration along with manuals, working papers and disks. Never a need to remove equipment from case. Simply remove lid, connect power and operate.

AT 301 Computer case	**$109.00**
AP 105 12" Monitor & accessory case	**99.00**
P 402 825 Printer case	**89.00**

Cases also available for Apple, Radio Shack and peripherals. Ask at your local computer store or order directly.

COMPUTER CASE COMPANY
5650 INDIAN MOUND CT. COLUMBUS, OHIO 43213 (614) 868-9464

EHS

SPECIAL—This month only!!

(Please mention this ad when ordering.)

EHS SOFTWARE

- PET MAE $169.95
  (Read the rave reviews about this Disk-based Macro Assembler Text Editor.)
  Graphics Drawing Compiler with purchase of MAE. . . $ 10.00
- APPLE and ATARI MAE $169.95
  Similar features as PET MAE. (Free — either 10 diskettes or Mini-Flex diskette file case.)
- PET Macro Assembler/Editor and Graphics Drawing Compiler. Both for . . . . . $ 55.00
- Cassette Rabbit for 3.0 Rom PET Computers. High-speed cassette I/O. Load and save an 8K program from your cassette deck in approximately 30 seconds versus almost 3 minutes without Rabbit. Specify memory. . . . . $ 19.95
  Rom version of Rabbit for 3.0, 4.0 or 8032 . . . . . $ 49.95
- TRAP 65 — Use this device to intercept unimplimented opcodes and even expand the 6502's instruction set. For practically any 6502 microcomputer. Super Special $112.50
- EPROM Board for ATARI Computers. Plugs into slot A or B and can accept 2716, 2516, 2532, 2732 type EPROMS. Half the price that ATARI gets for theirs . . . . . $ 19.95

Send mailing label and two 18¢ stamps for free EHS Gazette. Write for our catalog and spec sheets on our products.

EHS HARDWARE

- PEDISK (by CGRS) disk drive $550.00
  For COMMODORE computers. The most cost effective way to add a disk. MAE will be available for PEDISK Soon.
- VIC Color Computer $265.00
- ATARI 400 16K memory $345.00
- ATARI 800 16K memory $779.00
- EPROM programmer for PET Computers
  (The Branding Iron) . . . . . $ 75.00
  Includes Hardware and Software for programming 2716 and 2532 EPROMS.
- 2532 EPROM . . . . . $ 18.00
  4K byte EPROM. Use them in PET/APPLE/ATARI/SYM
- Starwriter 25 cps printer with tractors. CBM = $1690.00. APPLE = $1645.00 (parallel), $1700.00 (RS232)
- Universal Data System Modem direct connect 300 baud ORG/ANS $169.00
  with auto answer $199.00
  1200 baud $259.00

ACCESSORIES

- Syncom or Memorex
  5-1/4" diskettes . . . . . 10 for $30.00
- Mini-Flex diskette file case
  holds 50 — 5-1/4" diskettes $24.95

Call for prices on Zenith and Super Brain Computers and whatever else you need.

VISA* MasterCard

EASTERN HOUSE SOFTWARE
3239 Linda Drive
Winston-Salem, N.C. 27106 U.S.A.

PHONE ORDERS
(919) 924-2889
(919) 748-8446

(Please add sufficient Funds for Postage.)

HDE inc.

BOX 120
ALLAMUCHY, N.J. 07820
201-362-6574

HUDSON DIGITAL ELECTRONICS INC.

THE TASK* MASTERS

HDE supports the *TIM, AIM, SYM and KIM (TASK) with a growing line of computer programs and peripheral components. All HDE component boards are state-of-the-art 4½" x 6½", with on board regulation of all required voltages, fully compatible with the KIM-4 bus.

OMNIDISK 65/8 and 65/5
Single and dual drive 8" and 5¼" disk systems. Complete, ready to plug in, bootstrap and run. Include HDE's proprietary operating system, FODS (File Oriented Disk System).

DM816-M8A
An 8K static RAM board tested for a minimum of 100 hours and warranted for a full 6 months.

DM816-UB1
A prototyping card with on-board 5V regulator and address selection. You add the application.

DM816-P8
A 4/8K EPROM card for 2708 or 2716 circuits. On board regulation of all required voltages. Supplied without EPROMS.

DM816-CC15
A 15 position motherboard mounted in a 19" RETMA standard card cage, with power supply. KIM, AIM and SYM versions.

DISK PROGRAM LIBRARY
Offers exchange of user contributed routines and programs for HDE Disk Systems. Contact Progressive Computer Software, Inc. for details.

HDE DISK BASIC
A full range disk BASIC for KIM based systems. Includes PRINT USING, IF . . . THEN . . . ELSE. Sequential and random file access and much more. $175.00

HDE ADVANCED INTERACTIVE DISASSEMBLER (AID)
Two pass disassembler assigns labels and constructs source files for any object program. Saves multiple files to disk. TIM, AIM, SYM, KIM versions. $95.00

HDE ASSEMBLER
Advanced, two pass assembler with standard mnemonics. KIM, TIM, SYM and KIM cassette versions. $75.00 ($80.00 cassette)

HDE TEXT OUTPUT PROCESSING SYSTEM (TOPS)
A comprehensive text processor with over 30 commands to format and output letters, documents, manuscripts. KIM, TIM and KIM cassette versions. $135.00 ($142.50 cassette)

HDE DYNAMIC DEBUGGING TOOL (DDT)
Built in assembler/disassembler with program controlled single step and dynamic breakpoint entry/deletion. TIM, AIM, SYM, KIM AND KIM cassette versions. $65.00 ($68.50 cassette)

HDE COMPREHENSIVE MEMORY TEST (CMT)
Eight separate diagnostic routines for both static and dynamic memory. TIM, AIM, SYM, KIM and KIM cassette versions. $65.00 ($68.50 cassette)

AVAILABLE DIRECT OR FROM THESE FINE DEALERS:

Progressive Computer Software
405 Corbin Road
York, PA 17403
(717) 845-4954

Johnson computers
Box 523
Medina, Ohio 44256
(216) 725-4560

Falk-Baker Associates
382 Franklin Avenue
Nutley, NJ 07110
(201) 661-2430

Perry Peripherals
P.O. Box 924
Miller Place, NY 11764
(516) 744-6462

Lux Associates
20 Sunland Drive
Chico, CA 95926
(916) 343-5033

Laboratory Microcomputer Consultants
P.O. Box 84
East Amherst, NY 14051
(716) 689-7344

ASMLST: Full-Sized Assembler Listing for AIM

ASMLST reads an AIM 20-column assembly listing produced by the AIM ROM assembler and reformats it into a full-sized assembly listing.

Joel Swank
25730 Beach Drive
Rockaway OR 97136

The AIM 4K ROM assembler is one of the most useful features of the AIM. You can enter assembler source code into memory with the AIM editor, assemble it directly from memory, and store the object in memory for quick testing. The AIM assembler can also read source from any system device and write object and assembly listing to any device.

The listing output of the AIM assembler is designed for the AIM onboard 20-column printer. Single statements are broken into as many as three lines for the AIM printer. This format is fine for short programs, but cumbersome for longer ones, and wastes a lot of paper on a TTY or full size printer. As the size of the programs I was writing increased, I realized that I would need full size listings. First I contacted 6502 software vendors, looking for a full-feature AIM compatible assembler. The only one I could find came with disk systems. So, as a last resort I decided to write ASMLST.

ASMLST reads a listing generated by the AIM assembler and constructs a full size listing. The format of the listing produced is similar to the listing format in the AIM monitor program listing manual. ASMLST produces a listing with six fields. The first field is the four character address. This address is taken from the AIM listing lines beginning with '= ='. The address is then updated as each object code line is processed. It is printed on each line, not just on the lines with labels, as with the AIM for-

Listing 1

```
                ;*    ASMLST    *
                ;ZERO PAGE STORAGE
                ;
                INDEXO EPZ $00          ;OUTPUT INDEX
                LOC    EPZ $01          ;LOCATION COUNTER
                LOCL   EPZ $01
                LOCH   EPZ $02
                TEMP   EPZ $03          ;TEMPORARY STORAGE
                CODEIN EPZ $04          ;FLAG TO FORCE PUTLIN
                EOF    EPZ $05          ;END OF FILE FLAG
                LOCUP  EPZ $06          ;LOCATION COUNTER UPDATE
                ;
                ;AIM SUBROUTINES
                ;
                CRLF   EQU $E9F0        ;SEND CRLF TO AOD
                RCHEK  EQU $E907        ;INTERRUPT CHECK
                WHEREO EQU $E871        ;OPEN OUTPUT
                WHEREI EQU $E848        ;OPEN INPUT
                INALL  EQU $E993        ;GET A CHARACTER
                OUTALL EQU $E9BC        ;PUT A CHARACTER
                NUMA   EQU $EA46        ;PUT A HEX BYTE
                PACK   EQU $EA84        ;ASCII TO HEX
                DU12   EQU $E511        ;TAPE CLOSE ROUTINE
                COMIN  EQU $E1A1        ;RETURN TO MONITOR
                ;
                ;EQUATES
                ;
                ;OUTPUT LINE FIELDS
                ;
                ADD    EPZ $00          ;ADDRESS
                CODE   EPZ $05          ;OBJECT CODE
                LABEL  EPZ $0D          ;LABEL
                OPCODE EPZ $14          ;OPCODE
                OPER   EPZ $18          ;OPERAND
                COMM   EPZ $24          ;COMMENT
                ;
                CR     EPZ $0D
                LF     EPZ $0A
                CTLZ   EPZ $1A
                ;
                OUTFLG EQU $A413        ;OUTPUT DEVICE
                ;
                ;      ORG $200
0200 2048E8     ASMLST JSR WHEREI       ;OPEN INPUT DEVICE
0203 2071E8            JSR WHEREO       ;OPEN OUTPUT DEVICE
0206 A905              LDA #$05
0208 8503              STA TEMP         ;SKIP THE FIRST 5 LINES
020A 208103     SKLOOP JSR GETLIN
020D C603              DEC TEMP
020F D0F9              BNE SKLOOP
0211 A900              LDA #$00         ;CLEAR LOCATION INCREMENT
0213 8506              STA LOCUP
0215 8505              STA EOF          ;AND END FLAG
0217 8501              STA LOCL         ;AND LOCATION COUNTER
0219 8502              STA LOCH
021B 20B303            JSR CLROUT       ;INIT OUTPUT BUFFER
021E 2007E9     NXTLIN JSR RCHEK        ;INTERRUPT?
0221 A505              LDA EOF          ;AT END?
0223 D029              BNE CLOSE        ;YES, GO FINISH UP
0225 208103            JSR GETLIN       ;GET A LINE
```

(Continued)

mat. The next field is the object code field. This field contains 0 to 3 bytes of object code generated by the statement. The label field is next, followed by the opcode and operand fields. Last is the comment field. The actual positions of these fields are defined by equates in the assembly source (see listing 1).

Because of the format of the AIM assembly listing, it is not always possible to tell how the source was entered. One problem is that all comments are printed on a separate line. This means that ASMLST cannot tell whether a comment was part of the preceding line or entered on a separate line. But, if you follow a few rules, you can enter your source so that you get the desired listing.

If you want a comment to be placed at the end of a line in the comment field, enter it on that line separated from the operand by a blank, or instead, enter it in column 1 of the following line. Both of these methods generate the same listing from the AIM assembler. This is true except for comments on the '=' directive. The AIM print format for the '=' directive is different. If the comment is entered on the same line as the '=' directive, the assembler will insert a blank line between the directive and the comment. So, if you want a comment on an '=' directive, you must enter it in the first column of the next line. If you want a comment to appear on a line by itself, you may enter it starting in column two or greater. This will generate a blank line and the comment will appear starting in the label field. A comment that follows a comment always appears on a line by itself.

There is also a problem with the data assignment directives (.BYTE, .WORD, etc.). ASMLST may not be able to keep the address field updated properly if the NOGEN listing option is used. When you use NOGEN, the generated data bytes are not printed. ASMLST must count these bytes to keep the address field updated. To avoid this problem you must assemble with the GEN option or specify only one operand on each directive.

ASMLST inserts a form feed ($0C) when it detects a page break (from the .PAGE directive). It deletes the first five lines (PASS 1, PASS 2, etc.) of the listing and copies **ERROR messages are they are. ASMLST is designed to work on debugged listings; an error in your program may foul it up. ASMLST recognizes the .END directive as the end of the listing.

```
0228 ADD103        LDA LBUFI          ;FIRST CHARACTER
022B               ;
022B               ;FIRST CHARACTER DETERMINES ROUTINE
022B               ;
022B C920          CMP '              ;BLANK?
022D D003          BNE NOBL           ;NOPE
022F 4CB102        JMP BLHAN
0232 C93D   NOBL   CMP '=             ;EQUALS?
0234 D003          BNE NOEQ           ;NOPE
0236 4CF102        JMP EQHAN
0239 C93B   NOEQ   CMP ';             ;SEMICOLON?
023B D003          BNE NOCOM          ;NOPE
023D 4C1003        JMP COMHAN
0240 C95F   NOCOM  CMP #$5F           ;UNDERSCORE?
0242 D003          BNE NOPAG          ;NOPE
0244 4C2703        JMP PAGHAN
0247 C92A   NOPAG  CMP '*             ;ASTERISK?
0249 D029          BNE HEXHAN         ;NO, MUST BE HEX CODE
024B 4C4503        JMP ASTHAN
024E               ;
024E               ;CLOSE : PUT LAST LINES, CLOSE TAPE,
024E               ;        AND RETURN TO MONITOR
024E 209503 CLOSE  JSR PUTLIN         ;SEND CURRENT LINE
0251 208103        JSR GETLIN         ;GET LAST LINE
0254 A20D          LDX #LABEL         ;COPY TO OUTPUT BUFFER
0256 206803        JSR COPOVR
0259 209503        JSR PUTLIN         ;SEND IT
025C A91A          LDA #CTLZ          ;SEND A CONTROL-Z
025E 20BCE9        JSR OUTALL
0261 20F0E9        JSR CRLF
0264 20F0E9        JSR CRLF           ;AND A COUPLA CRLFS
0267 AD13A4        LDA OUTFLG         ;CHECK OUTPUT DEVICE
026A C954          CMP 'T             ;IT IS TAPE?
026C D003          BNE NOTAP          ;NO
026E 2011E5        JSR DU12           ;YES, CLOSE FILE
0271 4CA1E1 NOTAP  JMP COMIN          ;RETURN TO MONITOR
0274               ;
0274               ;END OF MAINLINE
0274               ;
0274               ;THE FOLLOWING 6 ROUTINES EACH HANDLE 1 TYPE
0274               ;    OF INPUT LINE
0274               ;
0274               ;HEXHAN : PROCESS LINE OF OBJECT CODE
0274               ;
0274 A504   HEXHAN LDA CODEIN         ;BEEN HERE ON THIS STMT?
0276 D006          BNE PUTC           ;YES
0278 A500          LDA INDEXO         ;GET OUTPUT INDEX
027A C915          CMP #OPCODE+1      ;OPCODE PRESENT?
027C 9003          BCC COPHEX         ;NO, DON'T SEND
027E 209503 PUTC   JSR PUTLIN         ;YES, PUT CURRENT LINE
0281 A000   COPHEX LDY #$00           ;BEGINNING OF INPUT
0283 A205          LDX #CODE          ;OUTPUT INDEX
0285 8604          STX CODEIN         ;SHOW WE BEEN HERE
0287 B9D103 CHLUP  LDA LBUFI,Y        ;GET A CHAR
028A C90D          CMP #CR            ;END OF LINE?
028C F018          BEQ PUTHEX         ;YES, GO SEND
028E C920          CMP '              ;SPACE?
0290 F007          BEQ SKIPSP         ;YES, GO FINISH LINE
0292 9D5104        STA LBUFO,X        ;COPY A CHAR
0295 C8            INY                ;BUMP POINTERS
0296 E8            INX
0297 10EE          BPL CHLUP          ;NEXT CHARACTER
0299 206203 SKIPSP JSR UPLOC          ;UPDATE LOCATION COUNTER
029C A007          LDY #$07
029E A214   COPCOD LDX #OPCODE        ;OPCODE OFFSET
02A0 206A03        JSR COPOV2         ;COPY TO OUTPUT BUFF
02A3 4C1E02        JMP NXTLIN         ;
02A6 206203 PUTHEX JSR UPLOC          ;UPDATE LOCATION COUNTER
02A9 8600          STX INDEXO         ;MARK END OF LINE
02AB 209503        JSR PUTLIN         ;SEND LINE
02AE 4C1E02        JMP NXTLIN         ;NEXT LINE
02B1               ;
02B1               ;BLHAN : PROCESS LINE STARTING WITH BLANK
02B1               ;
02B1 A500   BLHAN  LDA INDEXO         ;WHERE IN OUTPUT LINE?
02B3 C90D          CMP #LABEL         ;LABEL PRESENT?
02B5 9004          BCC NEWL           ;NO, SEND LINE
02B7 C915          CMP #OPCODE+1      ;OPCODE PRESENT?
```

(Continued)

```
02B9 9003          BCC SCANC          ;NO, DON'T SEND
02BB 209503  NEWL  JSR PUTLIN         ;SEND CURRENT LINE
02BE A001    SCANC LDY #$01           ;START SEARCH AT COLUMN 1
02C0 B9D103  BLSCH LDA LBUFI,Y        ;SCAN TO FIRST NON-BLANK
02C3 C90D          CMP #CR            ;WATCH FOR END
02C5 F01C          BEQ EMPLIN         ;NULL LINE
02C7 C920          CMP '              ;BLANK?
02C9 D003          BNE DOTCHK         ;NO, GO CHECK FOR DOT
02CB C8            INY                ;NEXT CHAR
02CC 10F2          BPL BLSCH
02CE C92E    DOTCHK CMP '.            ;CHECK FOR END STMT
02D0 D009          BNE BLCOP          ;NO
02D2 B9D203        LDA LBUFI+1,Y      ;GOT A DIRECTIVE, IS IT .END?
02D5 C945          CMP 'E
02D7 D002          BNE BLCOP          ;NO
02D9 8505          STA EOF            ;YES, FLAG END OF FILE
02DB A214    BLCOP LDX #OPCODE        ;OUTPUT INDEX
02DD 206A03        JSR COPOV2         ;COPY REST OF LINE
02E0 4C1E02        JMP NXTLIN         ;NEXT LINE
02E3 209503  EMPLIN JSR PUTLIN        ;SEND LINE
02E6 A920          LDA '
02E8 20BCE9        JSR OUTALL
02EB 20F0E9        JSR CRLF           ;AND AN EMPTY LINE
02EE 4C1E02        JMP NXTLIN         ;NEXT LINE
02F1               ;
02F1               ;EQHAN : PROCESS A LINE BEGINNING WITH '='
02F1               ;
02F1 209503  EQHAN JSR PUTLIN         ;SEND LINE IF ANY
02F4 A002          LDY #$02           ;POINT TO ADDRESS
02F6 205303        JSR GETHEX         ;CONVERT HIGH BYTE
02F9 8502          STA LOCH           ;SAVE IT IN LOCATION COUNTER
02FB 205303        JSR GETHEX         ;CONVERT LOW BYTE
02FE 8501          STA LOCL           ;SAVE
0300 B9D103        LDA LBUFI,Y        ;ANY MORE?
0303 C90D          CMP #CR
0305 F006          BEQ JNEXT1
0307 C8            INY
0308 A20D          LDX #LABEL         ;PREPARE FOR LABEL
030A 206A03  COPLAB JSR COPOV2        ;COPY LABEL
030D 4C1E02  JNEXT1 JMP NXTLIN        ;NEXT LINE
0310               ;
0310               ;COMHAN : PROCESS A COMMENT
0310               ;
0310 A600    COMHAN LDX INDEXO        ;ANY LINE IN PROCESS
0312 F008          BEQ COMLAB         ;NO
0314 E024          CPX #COMM          ;PAST COMMENT FIELD?
0316 B006          BCS COMCOP         ;YES, USE VALUE
0318 A224          LDX #COMM          ;NO, COPY TO COMMENT FIELD
031A D002          BNE COMCOP
031C A20D    COMLAB LDX #LABEL        ;COPY TO LABEL
031E 206803  COMCOP JSR COPOVR
0321 209503        JSR PUTLIN         ;SEND LINE
0324 4C1E02        JMP NXTLIN
0327               ;
0327               ;PAGHAN : PROCESS A NEW PAGE
0327               ;
0327 209503  PAGHAN JSR PUTLIN        ;SEND CURRENT LINE
032A A90C          LDA #$0C           ;SEND A FORM FEED
032C 20BCE9        JSR OUTALL
032F A20D          LDX #LABEL         ;COPY UNDERSCORES TO LABEL
0331 206803        JSR COPOVR
0334 209503        JSR PUTLIN         ;SEND IT
0337 208103        JSR GETLIN         ;GET TITLE
033A A20D          LDX #LABEL         ;COPY TO OUTPUT LINE
033C 206803        JSR COPOVR
033F 209503        JSR PUTLIN         ;SEND IT
0342 4C1E02        JMP NXTLIN         ;NEXT LINE
0345               ;
0345               ;ASTHAN : PROCESS ERROR STATEMENT
0345               ;
0345 209503  ASTHAN JSR PUTLIN        ;SEND CURRENT LINE
0348 A205          LDX #CODE
034A 206803        JSR COPOVR         ;COPY TO OUTPUT BUFF
034D 209503        JSR PUTLIN         ;SEND IT
0350 4C1E02        JMP NXTLIN
0353               ;
0353               ;
0353               ;SUBROUTINES FOLLOW
```

(Continued)

ASMLST inputs an AIM assembly listing from the AID (Active Input Device) and outputs the new listing to the AOD (Active Output Device). Start ASMLST at $200. You will receive the Standard IN= and OUT= prompts for the input and output devices. You can input from tape and send the output to a printer through the user port or to a TTY or CRT through the AIM TTY port. You can also input from tape and output to tape, if you have two remote controlled tape recorders. For long listings you may want to partition the listing output with the LIST and NOLIST Options. This is fine as long as you are sure that the .END statement is listed so that ASMLST can recognize the end of the listing.

ASMLST works if you have a full sized printer or terminal, but what about those of us with only the AIM 20-column printer? Help is available with 'SPLIT' program, which allows the AIM printer to print listings up 80 characters wide. It inputs the output from ASMLST and splits it into 20-column sections for printing on the AIM onboard printer. You then cut and paste to create a full width listing. Execute SPLIT at $200 and it will first prompt the section to be printed. Each section is 20 columns wide and numbered 1 to 4. Section 1 is columns 1-20, section 2 is columns 21-40, and so on. SPLIT then issues the standard IN= and OUT= prompts. The input file must be read once to print each section. The number of sections allowed can be increased to 7, simply by changing the value in the compare instruction at $221. The size of each section is controlled by the value assigned to the label 'SIZE'. If you would like to use ASMLST with SPLIT, you may want to change the equates in ASMLST for the output line fields as follows:

ADD	= 0
CODE	= 5
LABEL	=12
OPCODE	=19
OPER	=23
COMM	=31

ASMLST then produces a listing format that will fit into 40 columns. This leaves only 10 columns for the comment field. The 40-column format makes the most efficient use of two passes of SPLIT On the AIM printer. ASMLST and SPLIT are especially valuable if you want to publish a program in a magazine or newsletter. The reader will appreciate the increased readability of your listing. You can also use SPLIT to print BASIC files. In fact any ASCII file that ends with a Control-Z can be printed in sections with SPLIT.

Micro Co-op

Micro Co-op is a software co-operative. We publish a bi-monthly newsletter describing various new and useful software products as they become available. We also try to give reasonable comparisons of existing products that help you make more educated buying decisions. Our members may purchase most software through us below retail price.

For more information, write for a free copy of our newsletter.

Micro Co-op
(Specify Apple or Atari) Newsletter
P.O. Box 432
West Chicago, IL 60185
(312) 231-0912

OHIO
SCIENTIFIC
NEW & USED SYSTEMS

- HARDWARE
- SOFTWARE
- PARTS
- ACCESSORIES

-SERVICE AVAILABLE-

SUNSET ELECTRONICS
2254 TARAVAL ST.
SAN FRANCISCO, CA 94116
(415) 665-8330


```
0353            ;
0353            ;
0353            ;GETHEX : GET A BYTE OF ASCII HEX FROM INPUT LINE
0353            ;
0353 B9D103     GETHEX LDA LBUFI,Y          ;GET HIGH NYBBLE
0356 2084EA            JSR PACK             ;CONVERT TO BINARY
0359 C8                INY
035A B9D103            LDA LBUFI,Y          ;GET LO NYBBLE
035D 2084EA            JSR PACK             ;CONVERT TO BIN
0360 C8                INY
0361 60                RTS
0362            ;
0362            ;UPLOC ; SAVE LOCATION COUNTER INCREMENT
0362            ;
0362 C8         UPLOC  INY                  ;BUMP POINTER
0363 98                TYA
0364 4A                LSR                  ;DIVIDE BY 2
0365 8506              STA LOCUP            ;SAVE
0367 60                RTS
0368            ;
0368            ;COPOVER : COPY ALL OR PART OF THE INPUT LINE TO THE
0368            ;          OUTPUT BUFFER AT OFFSET IN X
0368            ;
0368 A000       COPOVR LDY #$00             ;BEGINNING OF INPUT LINE
036A            ;ENTRY TO COPY PARTIAL LINE
036A B9D103     COPOV2 LDA LBUFI,Y          ;GET A CHAR
036D C90D              CMP #CR              ;END?
036F F00D              BEQ COPOUT           ;YES, QUIT
0371 C93D              CMP '=               ;CHECK FOR '=' DIRECTIVE
0373 D002              BNE STOCH            ;NOPE
0375 8504              STA CODEIN           ;REMEMBER
0377 9D5104     STOCH  STA LBUFO,X          ;STORE IN OUTPUT BUFFER
037A E8                INX                  ;NEXT
037B C8                INY                  ;NEXT
037C 10EC              BPL COPOV2
037E 8600       COPOUT STX INDEXO           ;SAVE LAST
0380 60                RTS
0381            ;
0381            ;GETLIN : GET ONE LINE INTO INPUT BUFFER
0381            ;
0381 A000       GETLIN LDY #$00             ;BEGINNING
0383 2093E9     GETCHR JSR INALL            ;GET A CHAR
0386 C90A              CMP #LF              ;LINE FEED?
0388 F0F9              BEQ GETCHR           ;YES, IGNORE IT
038A 99D103            STA LBUFI,Y          ;SAVE
038D C90D              CMP #CR              ;END?
038F F003              BEQ GOTLIN           ;YES, QUIT
0391 C8                INY
0392 10EF              BPL GETCHR           ;NEXT CHAR
0394 60         GOTLIN RTS
0395            ;
0395            ;PUTLIN : SEND OUTPUT BUFFER TO OUTPUT DEVICE
0395            ;
0395 A500       PUTLIN LDA INDEXO           ;ANYTHING THERE?
0397 F037              BEQ NOPUT            ;NO, SKIPPIT
0399 A502              LDA LOCH             ;GET HI BYTE OF LOC COUNTER
039B 2046EA            JSR NUMA             ;SEND TO OUTPUT
039E A501              LDA LOCL             ;GET LO BYTE
03A0 2046EA            JSR NUMA             ;SEND IT
03A3 A204              LDX #$04             ;INIT INDEX
03A5 BD5104     PUTCHR LDA LBUFO,X          ;GET A CHAR
03A8 20BCE9            JSR OUTALL           ;SEND IT
03AB E8                INX
03AC E400              CPX INDEXO           ;END OF LINE?
03AE 90F5              BCC PUTCHR           ;NO
03B0 20F0E9            JSR CRLF             ;TERMINATE LINE
03B3 A27F       CLROUT LDX #$7F             ;CLEAR OUTPUT BUFFER
03B5 A920              LDA '
03B7 9D5104     CLRLUP STA LBUFO,X
03BA CA                DEX
03BB 10FA              BPL CLRLUP
03BD A506              LDA LOCUP            ;UPDATE LOCATION COUNTER
03BF 18                CLC
03C0 6501              ADC LOCL
03C2 8501              STA LOCL
03C4 9002              BCC NOCY
03C6 E602              INC LOCH
03C8 A900       NOCY   LDA #$00             ;CLEAR DATA
```

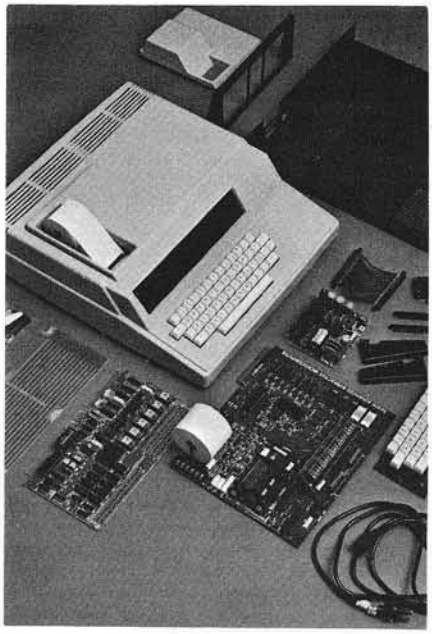
(Continued)

```
03CA 8506           STA LOCUP
03CC 8500           STA INDEXO
03CE 8504           STA CODEIN
03D0 60       NOPUT RTS
03D1          ;
03D1          ;BUFFERS
03D1          ;
0A51          LBUFI  DFS $80
0AD1          LBUFO  DFS $80
04D1          ;
                    END
```

Listing 2

```
                    ;
                    ;*      SPLIT       *
                    ;ZERO PAGE STORAGE
                    ;
                    COUNT  EPZ $00          ;CHARACTER COUNT
                    SECTON EPZ $01          ;CURRENT SECTION
                    ;
                    ;AIM SUBROUTINES
                    ;
                    BLANK  EQU $E83E        ;SEND BLANK TO D/P
                    REDOUT EQU $E973        ;INPUT AN ASCII CHAR
                    CRLF   EQU $E9F0        ;SEND CRLF TO AOD
                    RCHEK  EQU $E907        ;INTERRUPT CHECK
                    WHEREO EQU $E871        ;OPEN OUTPUT
                    WHEREI EQU $E848        ;OPEN INPUT
                    INALL  EQU $E993        ;GET A CHARACTER
                    OUTALL EQU $E9BC        ;PUT A CHARACTER
                    CKER00 EQU $E394        ;DISPLAY 'ERROR'
                    OUTPUT EQU $E97A        ;SEND TO DISPLAY
                    DU12   EQU $E511        ;TAPE CLOSE ROUTINE
                    COMIN  EQU $E1A1        ;RETURN TO MONITOR
                    ;EQUATES
                    CR     EQU $0D
                    CTLZ   EQU $1A
                    SIZE   EQU $14
                    LF     EQU $0A
                    OUTFLG EQU $A413
                           ORG $0200
                           OBJ $0800
                    ;
0200 A000           SPLIT  LDY #$00         ;INIT INDEX
0202 B9AF02         MSGLUP LDA MESG,Y       ;GET A CHAR
0205 F006                  BEQ GSPLIT       ;QUIT ON ZERO
0207 207AE9                JSR OUTPUT       ;DISPLAY IT
020A C8                    INY
020B 10F5                  BPL MSGLUP       ;NEXT
020D 2073E9         GSPLIT JSR REDOUT       ;GET REPLY
0210 C931                  CMP '1           ;LEGAL (1 TO 4)?
0212 9004                  BCC BADNUM       ;NO
0214 C935                  CMP '5
0216 9009                  BCC GOODNM       ;YES
0218 2094E3         BADNUM JSR CKER00       ;PRINT ERROR MSG
021B 20F0E9                JSR CRLF
021E 4C0002                JMP SPLIT        ;TRY AGAIN
0221 2907           GOODNM AND #$07         ;CLEAR HI BYTES
0223 8501                  STA SECTON       ;SAVE IT
0225 203EE8                JSR BLANK
0228 2048E8                JSR WHEREI       ;OPEN INPUT
022B 2071E8                JSR WHEREO       ;OPEN OUTPUT
022E 205802         NXTLIN JSR GETLIN       ;GET A LINE FROM INPUT DEVICE
0231 B009                  BCS FIN          ;QUIT ON CTLZ
0233 209502                JSR SPLCHK       ;CALCULATE SPLIT
0236 207B02                JSR PUTLIN       ;SEND LINE
0239 4C2E02                JMP NXTLIN       ;GET ANOTHER LINE
023C AD13A4         FIN    LDA OUTFLG       ;CHECK OUTPUT DEVICE
023F C950                  CMP 'P           ;PRINTER?
0241 F012                  BEQ NOTAP        ;YES SKIP EOF MARK
0243 A91A                  LDA #CTLZ        ;SEND A CONTROL-Z
0245 20BCE9                JSR OUTALL
0248 20F0E9                JSR CRLF
024B 20F0E9                JSR CRLF         ;AND 2 CRLFS
024E C954                  CMP 'T           ;IS IT TAPE?
0250 D003                  BNE NOTAP        ;NO
0252 2011E5                JSR DU12         ;YES, CLOSE FILE
0255 4CA1E1         NOTAP  JMP COMIN        ;RETURN TO MONITOR
0258                ;END OF MAINLINE
0258                ;SUBROUTINES FOLLOW
0258                ;
0258                ;GETLIN : GET ONE LINE INTO INPUT BUFFER
```

(Continued)



TAKE AIM.

Unique Data Systems has. We've taken Rockwell's AIM 65 Microcomputer, packaged it in a professional enclosure and turned it into a versatile, higher capacity microcomputer system. Complete with a memory-I/O board, modem board or wire wrap prototyping board, power supply, cables and connectors. It makes the AIM 65 a joy to work with, and there's even space inside for your own special circuitry. We'll sell you the whole package or just the bits and pieces you need for your application. We're AIM 65 specialists. We're Unique Data Systems.

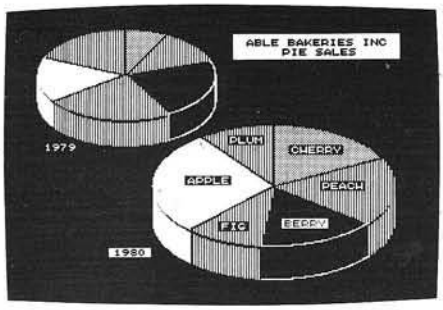
Unique Data Systems
15041 Moran Street
Westminster, CA 92603

(714) 895-3455

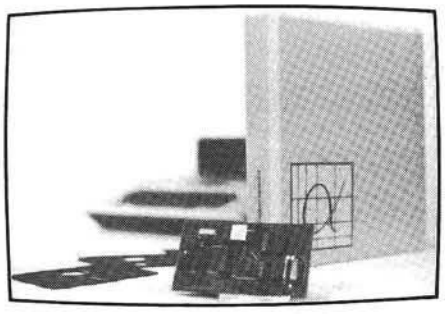
POWER
FOR YOUR APPLE

SPEED II
LANGUAGE SYSTEM

APPLESOFT: 30.3 MIN.
MICROSPEED: 3.9 MIN.



FASTEST: SIX TO SIXTY TIMES FASTER THAN APPLESOFT
MOST POWERFUL: MORE POWER THAN BASIC PASCAL OR FORTRAN
EXPANDABLE: LANGUAGE BASED ON FORTH
CREATIVE: GROW YOUR OWN LANGUAGE
USER-FRIENDLY: EASIEST FOR YOU TO LEARN



REQUIRES
48K APPLE II or II+ SINGLE DISK
SEE YOUR DEALER OR CONTACT:

applied analytics incorporated
8910 Brookridge Dr., Suite 803, Upper Marlboro, Md. 20870
(301) 627-6650
I'm Interested: Please Send
☐ 160 Page Manual $35.00
☐ Detailed Information
Name _____
Address _____
City _____
State _____ Zip _____

```
0258                ;
0258 A080       GETLIN LDY #$80          ;CLEAR BUFFER
025A A920              LDA '
025C 99B802     CLRLUP STA LBUF,Y
025F 88                DEY
0260 D0FA              BNE CLRLUP
0262 2093E9     GETCHR JSR INALL         ;GET A CHAR
0265 C90A              CMP #LF           ;LINE FEED?
0267 F0F9              BEQ GETCHR        ;YES, IGNORE IT
0269 C91A              CMP #CTLZ         ;END OF FILE?
026B F00C              BEQ ZOUT          ;YES
026D 99B802            STA LBUF,Y        ;SAVE
0270 C90D              CMP #CR           ;END?
0272 F003              BEQ GOTLIN        ;YES, QUIT
0274 C8                INY
0275 10EB              BPL GETCHR        ;NEXT CHAR
0277 18         GOTLIN CLC
0278 60                RTS
0279 38         ZOUT   SEC
027A 60                RTS
027B                ;
027B                ;PUTLIN : SEND OUTPUT BUFFER TO OUTPUT DEVICE
027B                ;
027B A914       PUTLIN LDA #SIZE         ;INIT CHAR COUNT
027D 8500              STA COUNT
027F B9B802     PRLUP  LDA LBUF,Y        ;GET A CHARACTER
0282 C90D              CMP #CR           ;END OF LINE?
0284 F008              BEQ EOL           ;YES
0286 20BCE9            JSR OUTALL
0289 C8                INY
028A C600              DEC COUNT         ;COUNT IT
028C D0F1              BNE PRLUP         ;CONTINUE UNTIL SIZE
028E 2007E9     EOL    JSR RCHEK         ;CHECK FOR INTERRUPT
0291 20F0E9            JSR CRLF          ;END LINE
0294 60                RTS
0295                ;
0295                ;SPLCHK : SET Y FOR PROPER SECTION OF LISTING
0295                ;
0295 A601       SPLCHK LDX SECTON        ;GET SECTION NUMBER
0297 A000              LDY #$00          ;BEGINNING OF LINE
0299 CA         SPLUP1 DEX               ;COUNT SECTION
029A F012              BEQ SPOUT         ;DUN
029C A914              LDA #SIZE         ;GET LINESIZE
029E 8500              STA COUNT
02A0 B9B802     SPLUP2 LDA LBUF,Y        ;GET A CHAR
02A3 C90D              CMP #CR           ;END OF LINE?
02A5 F007              BEQ SPOUT         ;YES, QUIT
02A7 C8                INY
02A8 C600              DEC COUNT         ;END OF SECTION?
02AA D0F4              BNE SPLUP2        ;NO, GET ANOTHER CHAR
02AC F0EB              BEQ SPLUP1        ;YES, TRY NEXT SECTION
02AE 60         SPOUT  RTS
02AF 534543     MESG   ASC 'SECTION='
02B2 54494F
02B5 4E3D
02B7 00                HEX 00
02B8            ;BUFFER
02B8                ;
0938            LBUF   DFS $80
                       END
```

Note: When the AIM assembler listing is directed to tape, the assembler does not properly close the file at the end of the assembly: This will cause the last block of data to never be written. To circumvent this problem you must run one of the following programs immediately after the assembly is finished.

If you are using any device EXCEPT tape for the source input, then execute the following routine:

```
CLD
JSR E511 (close tape file)
JMP E1A1 (return to AIM)
```

If you are using tape for both source input and listing output, execute the following program:

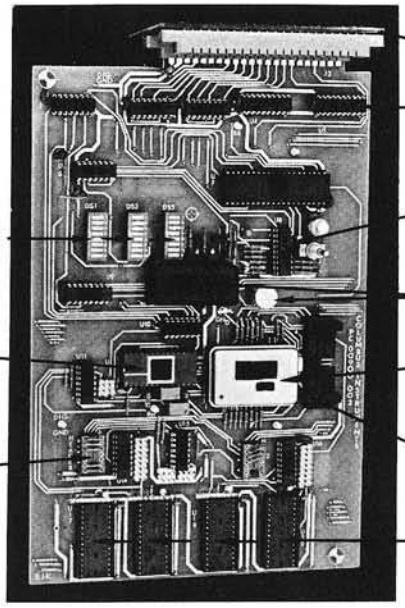
```
CLD
LDA #54     (move 'T' to INFLG
STA A412    and OUTFLG so
            alternate buffer
STA A413    will be used.)
JSR E511    (close tape file)
JMP E1A1    (return to AIM)
```

Either of these programs can be located in any unused spot in memory and can be executed with the GO command.

MICRO™

AIM-65/SYM-PET-KIM-6800

Universal Interface Board Converts AIM-65/SYM Into Professional Data Logger



(Also connects to PET or KIM with adapter cable. Adaptable to other 6502 and 6800 systems)

CONTAINS:

- ★ 12 bits, 16 channels, fast A/D converter
- ★ space for additional 16K RAM memory or 32K EPROM (or combination)
- ★ real time clock/calendar with real time interrupt capability and 10-year lithium battery backup
- ★ plugs directly into AIM-65 expansion connector with the help of a mini-mother board which supports up to three interface boards
- ★ supplied with supportive demonstration and control programs

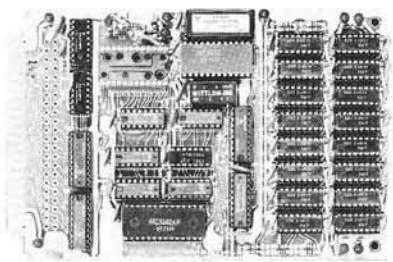
AVAILABLE MODELS:

★ IB-902	Additional Memory Space (only)	$ 390.00
★ IB-902-A	Calendar/Clock plus memory space	$ 690.00
★ IB-902-B	A/D (12 bits, 16 channels plus memory space)	$ 960.00
★ IB-902-AB	A/D, plus memory space and calendar/clock	$1,270.00
Mini mother board to support up to three (3) interface boards		$ 65.00

Quantity Discounts Available

COLUMBUS INSTRUMENTS INTERNATIONAL CORPORATION
Supplier of individual instruments and total measuring systems

950 N. HAGUE AVE., COLUMBUS, OHIO 43204 U.S.A.
PHONE: (614) 488-6176 TELEX: 246514



BETA 32K BYTE EXPANDABLE RAM FOR 6502 AND 6800 SYSTEMS
AIM 65 KIM SYM PET S44-BUS

- Plug compatible with the AIM-65/SYM expansion connector by using a right angle connector (supplied).
- Memory board edge connector plugs into the 6800 S44 bus.
- Connects to PET using an adaptor cable.
- Uses +5V only, supplied from the host computer.
- Full documentation. Assembled and tested boards are guaranteed for one full year. Purchase price is fully refundable if board is returned undamaged within 14 days.

Assembled with 32K RAM	$349.00
& Tested with 16K RAM	329.00
Bare board, manual & hard-to-get parts	99.00
PET interface kit. Connects the 32K RAM board to a 4K or 8K PET	$ 69.00

See our full page ad in BYTE and INTERFACE AGE.

---

wabash

8" or 5¼" flexible diskettes certified 100% error free with manufacturers 5 year limited warranty on all 8" media. Soft sectored in tilt-back* boxes of 10. 5¼" available in 10 Sector.
(Add $3.00 for plastic library cases)

8" single sided, single density	$29.95
8" single sided, double density	37.95
8" double sided, double density	48.95
5¼" single sided, single density	27.95
5¼" single sided, double density	29.95

*Tilt-Back is a trademark of Wabash, Inc.

8" DISK DRIVES

Shugart 801R	$390.00
NEC FD1160 (double sided)	595.00
Memorex MRX-101 8" Winchester style, hard disk drive, 10 megabytes	$2,000.00

TERMS: *Minimum order $15.00. Minimum shipping and handling $3.00. Calif. residents add 6% sales tax. Cash, checks, Mastercard, Visa and purchase orders from qualified firms and institutions are accepted. Product availability and pricing are subject to change without notice.*

INTERNATIONAL ORDERS: *Add 15% to purchase price for all orders. Minimum shipping charge is $20.00. Orders with insufficient funds will be delayed. Excess funds will be returned with your order. All prices are U.S. only.*

16K MEMORY EXPANSION KIT $29.00

For **Apple, TRS-80** keyboard, **Exidy**, and all other systems using 4116 dynamic rams or equivalent. All IC's are prime Mitsubishi MK 4116-3.

- 200 NSEC access, 375 NSEC cycle
- Burned-in and fully tested
- 1 full year parts replacement guarantee

ROCKWELL AIM 65

AIM 65 with 1K ram	$425.00
AIM 65 with 4K ram	485.00
AIM power supply	125.00
Professional AIM enclosure	169.00
Budget AIM enclosure	50.00
KIM enclosure	40.00
SYM enclosure	30.00

BETA COMPUTER DEVICES
1230 W. COLLINS AVE.
ORANGE, CA 92668
(714) 633-7280

master charge VISA

JINSAM™ 8.0

space
age
micro
software

**Used at NASA,
Kennedy Space Center
With Multiple Applications Related
to the Columbia Space Shuttle Project**

DEALER INQUIRY WELCOME.

JINI MICRO-SYSTEMS, Inc.

DATABASE MANAGEMENT SYSTEM DESIGN

BOX 274 KINGSBRIDGE STN., RIVERDALE, N.Y. 10463 (212) 796-6200

MICRO

New Publications

Mike Rowe
New Publications
34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

6809

CHROMASETTE Magazine is a monthly issued on cassette tape and devoted entirely to the TRS-80 Extended BASIC Color Computer. The first issue appeared in July 1981. Each monthly tape contains six to eight programs that directly load and run on the color computer. They include games, tutorials, utilities, and application programs by various authors. The same publisher issues *CLOAD Magazine*, for the TRS-80 Models I and III. Subscriptions are available for one year ($45.00) or half-a-year ($25.00) from CHROMASETTE Magazine, P.O. Box 1087, Santa Barbara, California 93102.

Atari

The Atari Assembler by Don Inman and Kurt Inman. Reston Publishing Company, Inc. (Reston, Virginia), 1981, xii, 270 pages, 78 illustrations, 6¼ × 9¼ inches.
ISBN: 0-8359-0237-4 (cloth) $14.95
ISBN: 0-8359-0236-6 (paper) $ 9.95

This is an introduction to assembly language programming for use with the Atari Assembler Cartridge. The book is written for the beginning assembly language programmer who has some knowledge of BASIC.

CONTENTS: *Introduction*—Computer Architecture; Review of BASIC; Graphics Keywords. *Machine Language from BASIC*—Binary Number Patterns; Hexadecimal Notation; Hexadecimal-to-Decimal Conversion; How the Machine Language Program Works; Summary; Exercises; Answers. *Memory Use*—Atari Memory Map; How BASIC Finds the Machine Language Program; Passing Variables to Machine Language Subroutine; Using the One-Variable Program; Passing More Than One Variable; A Machine Language Loop; New Instructions Used; Tracing Through the Subroutine; Summary; Exercises; Answers. *Getting Started with the Assembler*—The Writer/Editor; The Assembler Program; Executing the Machine Language Program—The Debugger; Summary; Exercises; Answers. *Special-Purpose Registers and Addressing Modes*—The Accumulator; the X and Y Registers; The Processor Status Register; The Stack Pointer Register; Addressing, Modes; Summary; Exercises; Answers. *Branching Out*—Examples Using Forward Branches; Examples Using Backward Branches; Using the Carry Flag; Using the Zero Flag; Using the Negative Flag; The Overflow Flag; Summary; Exercises; Answers. *Assembler Review*—Source Program Format; Methods to Use Operands; The Assembler Writer/Editor Mode; The Debug Mode; Exercises; Answers. *Designing a Program*—Absolute Indexed Addressing; Using the Add Five Pairs of Numbers Program; Using the Add Ten Program; A Variation of the Add Ten Program; Yet Another Variation; Summary; Exercises; Answers. *Addition and Subtraction*—Two-Byte Addition; Two Programs in Memory; Two-Byte Subtraction; Negative Numbers; Multiple-Byte Addition and Subtraction; Decimal Arithmetic; Summary; Exercises; Answers. *Shift and Rotate*—Arithmetic Shift Left; Logical Shift Right; Rotate Left; Rotate Right; Summary; Exercises; Answers. *Multiplication, Division, and Subroutines*—Eight-Bit Multiplication; Using the 8-Bit Multiplication Program; Eight-Bit Division; Subroutines; Using a Subroutine; Summary; Exercises; Answers. *Programming Practice*—Using a Logic Function; Entering the Subroutine; Program to Sound Off; Play Notes Program; Program to Shape Sound; Program to Print on the Screen; You're On Your Own. *Appendix A*-6502 Instructions—Flags Affected. *Appendix B*—6502 Instructions—Addressing Modes. *Appendix C*—Frequency Values for Three-Octave Scale. *Appendix D*—Atari Assembler Error Codes. *Appendix E*—Atari Operating System Errors. *Appendix F*—ATASCII Character Set. *Index*.

AIM

The Take AIM Manual, Volume 1 by James Hoyt Clark. Matrix Publishers, Inc. (11000 S.W. 11th, Beaverton, Oregon 97005). 1981, xii, 388 pages, diagrams, drawings, listings, 8½ × 11 inches, paperbound.
ISBN: 0-916460-29-0 $16.95

This manual is designed as an addition to the AIM 65 documentation that is enclosed with the AIM 65. To use this manual successfully, a reader must first be able to use the AIM 65 documentation.

CONTENTS: *How To Use This Manual. The Formal Introduction*—The AIM 65 Hardware; The PERSON; Some Useful Steps—Software Example. *In The Beginning*—A Short History; The Sixteen Commandments of Microcomputers; 0 0000 CAUTION; 1 0001 POWER; 2 0010 WIRING; 3 0011 The Foundation; 4 0100 INSIGHT/OUTSIGHT; 5 0101 & 6 0110 Info Sources; 7 0111 Micro Chauvinist; 8 1000 NOP; 9 1001 Other Programs—the LAW; A 1010 Checking with the Neighbor; B 1011 & C 1100 Q & A's; D 1101 Storage; E 1110 The Every Half Hour Be Carefuls; F 1111 An Understanding. *The Language of the AIM 65*—The Hex Format; The Mnemonic Format; The ASCII Text Format; The EDITOR and ASSEMBLER; The Cost of Assembling; How to Input the TAKE AIM Programs. *Microcomputer Basics*—Memory; Some Programming Information; The Brains—the 6502 Microprocessor; The Heart—Clocks and Timing. *AIM 65 Documentation Explained*—Manuals, Cards, and a Chart; The Chicken or the Egg; User's Guide; The Monitor Program Listing; 6500 Microprocessor Programming Reference Card; The AIM 65 Wall Chart; The Monitor and Editor Subroutines. *The AIM 65 Documentation Index. The AIM 65 Display*—The Display Hardware; DISPLAYING; *The AIM 65 Keyboard*—The Keyboard Hardware; KEYING; KBINT Program; ASCII Program; DEMCU Program. *Versatile Interface Adaptor (VIA)*—The 6522 Homologous Homuncular Warehouse; Diagram of the VIA 6522; VIA Data Output Register Schematics; Summary of VIA Control Registers; Summary of VIA Commands. *Glossary. The GAIMS Programs*—BINARY TO HEXADECIMAL CONVERSION Program—BINHX; CARDS Program—CARDS; REACTION TIMER Program—REACT; GOLLUM'S CAVERNS Program—GC; ESP Program—ESP; BAGELS Program—BAGEL; BRICKS Program—BRICK; HANGMAN Program—HANGM; TIC TAC TOE Program—TIC; STARWAY 090 Program—STAR9. *The UTILITY Programs*—ROTATING BILLBOARD—ROTBB; PRINTER WAVES Program—PRTWV; ADDITION & SUBTRACTION—ADDSB; TOTAL Program—TOTAL; TIMER Program—TIMER; Printer Paper Programs; MEMORY TEST Program—MTEST; Hexadecimal Input/Output Programs; EDITOR RESTORE Program—ESTOR; SUPER SIGNS Program—SIGNS; CONVERT DECIMAL TO HEX Program—CNVRT; SYMBOL TABLE—SYMTB; FIELD SORT Program—FSORT; RELOCATE MEMORY Program—RLOC; DISASSEMBLY 1-STEP Program—K1STP; FIBBONACCI NUMBER SERIES Program—FIBBN. *The Appendix—Contents*—ROCKWELL Specification Sheets; Questions for the TAKE AIM Manual; Worksheets and Memory Page Samples; Reply Form.

Pascal

Pascal Programs for Scientists and Engineers by Alan R. Miller. Sybex Inc. (2344 Sixth Street, Berkeley, California 94710), 1981, xxii, 378 pages, 134 listings and illustrations, 7 × 9 inches, paperbound.
ISBN: 0-89588-058-X $16.95

This book was written to help readers gain a proficiency in Pascal and to pro-

(Continued on next page)

New Publications

(Continued from page 45)

vide a library of programs useful for solving problems frequently encountered in science and engineering. It contains over 60 of the most frequently used scientific algorithms, along with their program implementation in Pascal. The book is designed not only for the practicing scientist or engineer but is also suitable for a junior- or senior-level engineering course in numerical methods. Users need a working knowledge of an applications language (Pascal, FORTRAN, or BASIC). Experience with vector operations and differential and integral calculus is also recommended by the authors.

CONTENTS: *Evaluation of a Pascal Compiler*—Introduction; Precision and Range of Floating-Point Operations; Pascal Program: A Test of the Floating-Point Operations; Pascal SIN and COS Functions; Pascal Program: Testing the SIN Function; Other Pascal Functions; External Files; A Power-of-10 Function; Pascal Program: Calculating Powers of 10; Summary. *Mean and Standard Deviation*—Introduction; The Mean; The Standard Deviation; Pascal Program: Mean and Standard Deviation; Random Numbers; Pascal Function: A Random Number Generator; Pascal Program: Evaluation of a Random Number Generator; Pascal Function: Gaussian Random Number Generator; Pascal Program: Evaluating RANDG; Summary. *Vector and Matrix Operations*—Introduction; Scalars and Arrays; Vectors; Matrices; Pascal Program: Matrix Multiplication; Determinants; Pascal Program: Determinants; Inverse Matrices and Matrix Division; Summary. *Simultaneous Solution of Linear Equations*—Introduction; Linear Equations and Simultaneous Equations; Solution by Cramer's Rule; Pascal Program: A More Elegant Use of Cramer's Rule; Solution by Gauss Elimination; Pascal Program: The Gauss Elimination Method; Solution by Gauss-Jordan Elimination; Pascal Program: Gauss-Jordan Elimination; Multiple Constant Vectors and Matrix Inversion; Pascal Program: Gauss-Jordan Elimination, Version Two; Ill-Conditioned Equations; Pascal Program: Solving Hilbert Matrices; A Simultaneous Best Fit; Pascal Program: The Best Fit Solution; Equations with Complex Coefficients; Pascal Program: Simultaneous Equations with Complex Coefficients; The Gauss-Seidel Iterative Method; Pascal Program: The Gauss-Seidel Method; Summary. *Development of a Curve-Fitting Program*—Introduction; The Main Program; A Printer Plotter Routine; A Simulated Curve Fit; The Curve-Fitting Algorithm; The Correlation Coefficient; Pascal Program: Least-Squares Curve Fitting for Simulated Data; Summary. *Sorting*—Introduction; Handling Experimental Data; A Bubble Sort; Pascal Program: The Bubble Sort and TSTSORT; Pascal Procedure: Bubble Sort with SWAP; A Shell Sort; Pascal Procedure: The Shell-Metzner Sort; The Quick Sort; Pascal Procedure: A Recursive Quick Sort; Pascal Procedure: A Nonrecursive Quick Sort; Incorporating Sort into the Curve-Fitting Program; Summary. *General Least-Squares Curve Fitting*—Introduction; A Parabolic Curve Fit; Pascal Program: Least-Squares Curve Fit for a Parabola; Curve Fits for Other Equations; Pascal Program: The Matrix Approach to Curve Fitting; Pascal Program: Adjusting the Order of the Polynomial; Pascal Program: The Heat-Capacity Equation; Pascal Program: The Vapor Pressure Equation; A Three-Variable Equation; Pascal Program: An Equation of State for Steam; Summary. *Solution of Equations by Newton's Method*—Introduction; Formulating Newton's Method; Pascal Program: A First Attempt at Newton's Method; Pascal Programs: Solving Other Equations; Pascal Program: The Vapor Pressure Equation; Summary. *Numerical Integration*—Introduction; The Definite Integral; The Trapezoidal Rule; Pascal Program: The Trapezoidal Rule with User Input for the Number of Panels; Pascal Program: An Improved Trapezoidal Rule; Pascal Program: Trapezoidal Rule with End Correction; Pascal Program: Simpson's Integration Method; Pascal Program: The Simpson Method with End Correction; The Romberg Method; Pascal Program: Integration by the Romberg Method; Functions that Become Infinite at One Limit; Pascal Program: Adjustable Panels for an Infinite Function; Summary. *Nonlinear Curve-Fitting Equations*—Introduction; Linearizing the Rational Function; Pascal Program: The Clausing Factor Fitted to the Rational Function; Linearizing the Exponential Equation; Pascal Program: An Exponential Curve Fit for the Diffusion of Zinc in Copper; Direct Solution of the Exponential Equation; Pascal Program: A Nonlinearized Exponential Curve Fit; Summary. *Advanced Applications: The Normal Curve, the Gaussian Error Function, the Gamma Function, and the Bessel Function*—Introduction; The Normal and Cumulative Distribution Functions; The Gaussian Error Function; Pascal Program: Evaluating the Gaussian Error Function Using Simpson's Rule; Pascal Program: Evaluating the Gaussian Error Function Using an Infinite Series Expansion; The Complement of the Error Function; Pascal Program: Evaluating the Complement of the Error Function; Pascal Program: A Faster Implementation of the Error Function; The Gamma Function; Pascal Program: Evaluation of the Gamma Function; Bessel Functions; Pascal Program: Bessel Functions of the First Kind; Pascal Program: Bessel Functions of the Second Kind; Summary. *Appendix A: Reserved Words and Functions. Appendix B: Summary of Pascal*—Minimum Standard Character Set; Variable Names; Numbers; Comments; Operations; Syntax; Conditional Statements; Iterative Statements; Transfer-of-Control Statements; Input and Output; Data Types. *Bibliography. Index.*

CBM/PET? SEE SKYLES ... CBM/PET?

"Should we call it Command-O or Command-O-Pro?"

That's a problem because this popular ROM is called the Command-O-Pro in Europe. (Maybe Command-O smacks too much of the military.)

But whatever you call it, this 4K byte ROM will provide your CBM BASIC 4.0 (4016, 4032) and 8032 computers with 20 additional commands including 10 Toolkit program editing and debugging commands and 10 additional commands for screening, formatting and disc file manipulating. (And our manual writer dug up 39 additional commands in the course of doing a 78-page manual!)

The Command-O extends Commodore's 8032 advanced screen editing features to the ultimate. You can now SCROLL up and down, insert or delete entire lines, delete the characters to the left or right of the cursor, select TEXT or GRAPHICS modes or ring the 8032 bell. You can even redefine the window to adjust it by size and position on your screen. And you can define any key to equal a sequence of up to 90 key strokes.

The Command-O chip resides in hexadecimal address $9000, the rightmost empty socket in 4016 and 4032 or the rearmost in 8032. If there is a space conflict, we do have Socket-2-ME available at a very special price.

Skyles guarantees your satisfaction: if you are not absolutely happy with your new Command-O, return it to us within ten days for an immediate, full refund.

Command-O from Skyles Electric Works **$75.00**

Complete with Socket-2-Me **95.00**

Shipping and Handling ..... *(USA/Canada)* **$2.50** *(Europe/Asia)* **$10.00**

California residents must add 6%/6½% sales tax, as required.

Skyles Electric Works
231E South Whisman Road
Mountain View, California 94041
(415) 965-1735

Visa/Mastercard orders: call tollfree **(800) 227-9998** (except California). California orders: please call **(415) 965-1735.**

SEE SKYLES ... CBM/PET? SEE SKYLES ... CBM/PET?

APPLE DISK & MEMORY UTILITY

- **Repairs Blown Disks**
- **Reads Nibbles**
- **Maps Disk Space**
- **Searches Disks**
- **Searches Memory**
- **Edits Disk Sectors**
- **Outputs Screen to Printer**
- **Displays Memory In HEX/ASCII**

THE INSPECTOR

These utilities enable the user to examine data both in the Apple's memory and on disks. Simple commands allow scanning through RAM and ROM memory as well as reading, displaying and changing data on disk.

Read and rewrite sections of Random Access files. Reconstruct a blown VTOC. Weed out unwanted control characters in CATALOG listings. UnDELETE deleted files or programs. Repair files that have erroneous data. All without being under program control. and more.....

You may transfer sectors between disks. This allows you to transfer DOS from one disk to another thereby saving a blown disk when all that's blown is DOS itself; or to restore a portion of a blown disk from its backup disk.

Its unique NIBBLE read routine provides a Hi-Res graphical representation of the data on any track allowing you to immediately ascertain whether your disk is 13 sector or 16 sector. Get an I/O error...is it because you have the wrong DOS up? is it because of a bad address field? or a bad data field? or because a track was erased? This will allow you to tell in an instant without blowing away any program in memory.

The **INSPECTOR** even lets you search through an entire disk or through on-board memory for the appearance of a string. Now you can easily add lower case to your programs (with LCA).

Do you want to add so-called illegal line numbers into your program? or have several of the same line numbers in a program (like the professional programmers do)? or input unavailable commands (like HIMEM to Integer Basic)? or put quotation marks into PRINT statements? Here's the easy way to do them all!

AND MORE

The **INSPECTOR** provides a USER exit that will interface your own subroutines with those of the **INSPECTOR** itself. For example, just put a screen dump routine (sample included in documentation) at HEX 0300 and press CTRL-Z. The contents of the screen page will print to your printer.

ROM RESIDENT ROUTINES

The **INSPECTOR** utilities come on an easily installed EPROM. This makes them always available for instant use. No need to load a disk and run a program.

FULLY DOCUMENTED

Unlike other software of its kind, The **INSPECTOR** comes with an EASY to understand manual and reference card. Examples and graphics help even the uninitiated use the power of these utilities. And furthermore, we offer the kind of personal service which you have never experienced from a software vendor before.

See your LOCAL DEALER OR . . . Mastercard or Visa users call TOLL FREE 1-800-835-2246. Kansas residents call 1-800-362-2421. Or send $49.95. Illinois residents add $3 sales tax.

SYSTEM REQUIREMENTS

All Apple II configurations that have access to Integer Basic (either in ROM or RAM) will support The **INSPECTOR**. Just place the chip in empty socket D8 either on the mother board or in an Integer firmware card. Apple II+ systems with RAM expansion boards or language systems will receive the **INSPECTOR** on disk to merge and load with INTBASIC.

And...if you have an Apple II+, without either RAM or ROM access to Integer Basic, you will still be able to use The **INSPECTOR** because we are making available 16k RAM expansion boards at a very affordable price. Not only will you be able to use The **INSPECTOR**, but you will also have access to Integer Basic and other languages. Our price for BOTH the **INSPECTOR and our 16k RAM board is $169.95**, less than most RAM boards alone. Call our office for details.

Another Quality Product from
Omega Software Products, Inc.
222 S. Riverside Plaza, Chicago, IL 60606
Phone (312) 648-1944

© 1981 Omega Software Products, Inc.
Apple is a registered trademark of Apple Computer, Inc.

Your Pascal too slow? Not anymore...

with the PASCAL SPEED-UP KIT, which includes THE MILL: the easiest way to give your Pascal system a tremendous performance boost.

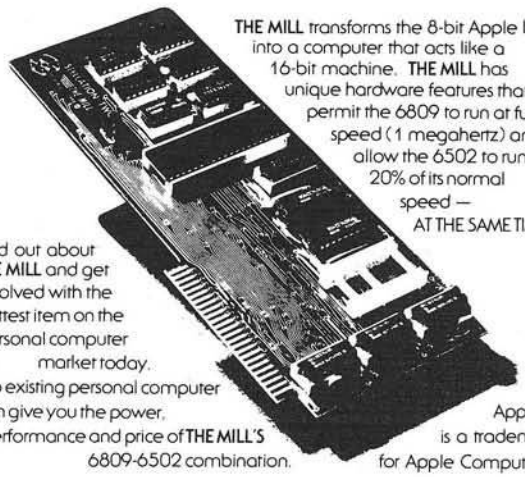
Here is how it works:

1) Plug in THE MILL
2) Run our configuration program one time
3) That's all

You now have a 30 to 300% faster Pascal P-machine, and you don't have to recompile, reprogram or relink. FORTRAN users may also take advantage of THE PASCAL SPEED-UP KIT. Contact your local Apple dealer for more information.

"Coming June 1, 1981 to your local Apple dealer"
THE ASSEMBLER DEVELOPMENT KIT

STELLATION TWO makes available the tools necessary to take full advantage of THE MILL. Enter the world of true MULTIPROCESSING with THE PASCAL SPEED-UP KIT and THE ASSEMBLER DEVELOPMENT KIT, available only from STELLATION TWO.



THE MILL transforms the 8-bit Apple II into a computer that acts like a 16-bit machine. **THE MILL** has unique hardware features that permit the 6809 to run at full speed (1 megahertz) and allow the 6502 to run at 20% of its normal speed — AT THE SAME TIME!

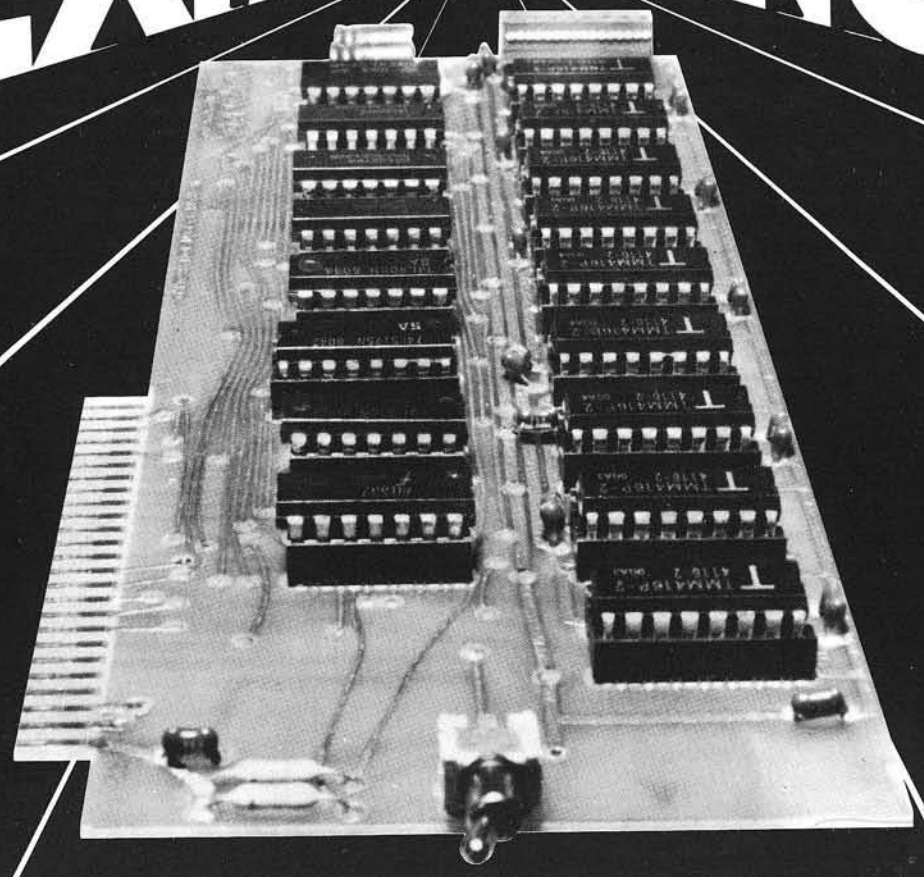
Find out about **THE MILL** and get involved with the hottest item on the personal computer market today. No existing personal computer can give you the power, performance and price of **THE MILL'S** 6809-6502 combination.

Apple II is a trademark for Apple Computer, Inc.

STELLATION TWO

P.O. BOX 2342– **N2**
SANTA BARBARA, CA. 93120
(805) 966-1140

EXPAND ☆ ENHANCE



16K RAM EXPANSION BOARD FOR THE APPLE II* $195.00

The Andromeda 16K RAM Expansion Board allows your Apple to use RAM memory in place of the BASIC Language ROMs giving you up to 64K of programmable memory. Separate Applesoft* or Integer BASIC ROM cards are no longer needed. The 16K RAM Expansion Board works with the Microsoft Z-80 card, Visicalc, DOS 3-3, Pascal, Fortran, Pilot, and other software. A switch on the card selects either the RAM language or the mainboard ROMs when you reset your Apple.

The Andromeda 16K RAM Expansion Board has a proven record for reliability with thousands of satisfied customers.

Now with One Year Warranty.

*Apple II and Applesoft are trademarks.

ANDROMEDA ☆ INCORPORATED

P.O. Box 19144
Greensboro, NC. 27410
919 852-1482

Distributed By:

COMPUTER DATA SERVICES

P.O. Box 696
Amherst, NH. 03031
603 673-7375

Apple Graphics

This month's Apple bonus section features a special concentration on graphics. The versatile graphical capabilities of the Apple lend themselves easily to many useful and unusual applications. These applications can range from rather ordinary ones, like graph plotting, to some very sophisticated uses like solid area shading and hidden surface removal. In this section, the MICRO staff has collected five graphics-oriented articles, each of which highlights a different use of the Apple's graphical capabilities.

"SHAPER," by C. Osborne, is an easy-to-use, powerful utility designed to create and maintain Applesoft shape tables. While several shape table utilities have appeared before, both in the pages of MICRO and in other magazines, none has worked quite as quickly and easily as SHAPER. If you have not yet explored the possibilities offered by shape tables, SHAPER is the perfect place to start.

"Lo-Res Graphics and Pascal," by C.D. Heth, unlocks the power of the Apple's low resolution graphics from within the Pascal environment. One of the most common complaints about Apple Pascal has been that there are no Lo-Res graphics instructions on capabilities. With the routines explained in this article, the Apple Pascal user can install these instructions and take advantage of the 16-color graphics for which the Apple is famous. With low resolution graphics installed, the structure and versatility of Apple Pascal will become even more alluring than it already is.

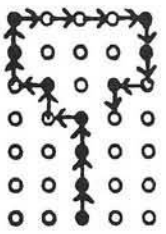
"Paddle Hi-Res Graphics," by K. Woodward, interfaces Applesoft high resolution graphics to the game paddle control. Designed to enable drawing of backgrounds on the high resolution screen, the programs provided in this article demonstrate the power of interactive graphic cursor control, and show how easily such control is implemented on the Apple. Additionally, the article explains some of the graphical transformations used in the program — transformations which are universally essential to any computer graphics system.

"True 3-D Images on the Apple II," by A. Radcliffe, illustrates how 3-dimensional views can be generated on the Apple Hi-Res screen by the use of stereoscopic pairs. The blue Hi-Res "roller-coaster" depicted on this month's cover represents one such pair and many others can be generated by the "noisy coaster" routine explained in the article. Besides providing a lucid description of the theory and techniques underlying 3-D view generation, this article and its accompanying programs should provide hours of fun and amusement. (They did for the MICRO staff.)

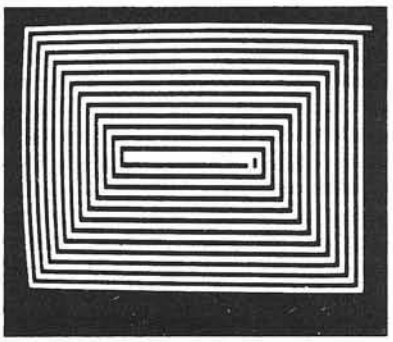
"Apple Bits," by R. Vile, discusses techniques for displaying patterns in low resolution graphics mode. The first part of this three-part article, presented in this month's issue, describes a fast, machine language routine which converts compact, numeric data files into low resolution pictures. A demonstration program, also described, illustrates how the routine is used to produce animation on the Lo-Res screen. The next two parts of the article, which will appear in October and November, will illustrate methods of generating and using the numeric data files. Together, the three parts of the series will enable more effective use of the Apple's low resolution graphics.

Each of these graphics articles tackles a different capability of Apple graphics. By reading through them, and studying the program material provided, it should be an easy matter to unlock the hidden power of your Apple's video display.

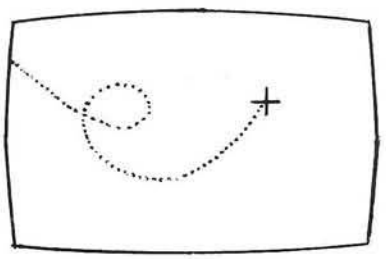
The graphics capability is, without doubt, one of the most powerful of the Apple, and is certainly one of the most interesting areas of computer science. We hope this special section helps to stimulate exploration of this exciting area by all MICRO readers.



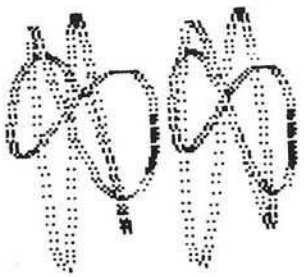
SHAPER — P. 50



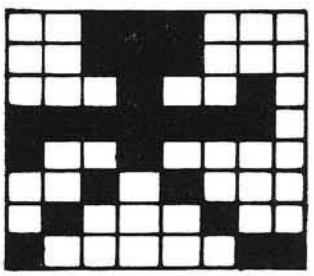
Lo-Res and Pascal — P. 62



Paddle Hi-Res — P. 68



3-D Images — P. 71



Apple Bits — P. 75

MICRO™

SHAPER: A Utility Program for Managing Shape Tables

"Shaper" describes a utility program for building and managing shape tables for the Apple II computer. The program allows the user to build shape tables by vector input, and view the shape while it is being built. The user can then change shapes, add shapes, delete shapes, review a shape table, and save or load a shape table.

Clement D. Osborne
949 S. Kingston St.
Aurora, Colorado 80012

Shape tables are a very valuable tool of the Applesoft language. Programmers are allowed to define their own shapes, and then by implementing the Applesoft commands DRAW, XDRAW, ROT, SCALE, and SHLOAD, the shapes can be drawn, erased, rotated, enlarged, and loaded from tape. The shape tables can be used in programs to create dynamic effects on the high resolution graphics screen. Animating, displaying game objects, and mixing text and graphics on the high resolution graphics screen create only a subset of the possible uses of shape tables. The real problem of utilizing shape tables is the creation of the table. This program relieves the programmer of the tedious task of defining the shapes and setting up the table.

SHAPER is a complete program for building and modifying shape tables. The first function, BUILD, allows shape tables to be constructed. All details are handled for creating the index portion and shape definition portion of the table. The user types in vector definitions by using the keys 0 through 7. By hitting the ESC key the high resolution screen will display the shape as it is being typed in. The ADD function allows more shapes to be added to an already existing table. All new shape definitions are added at the end of the table, and the indexes are added in the index portion of the table. The function CHANGE allows the user to replace existing shapes in a table with a new shape definition. The actual shape definition is added at the end of the table, the old shape definition is removed, and the indexes are updated to point to the proper locations.

DELETE allows the user to remove unwanted shapes from a shape table. After the shapes are removed, the table is compressed. The remaining shapes will then have different numbers when referring to them with the DRAW and XDRAW commands. The REVIEW function displays information about a shape table and allows the user to view the shapes in the table. The SAVE/LOAD function allows shape tables to be loaded from tape and saved or loaded to and from disk. Creating shape tables with SHAPER is an enjoyable process, and using the shapes in Applesoft programs adds a new dimension to programming the Apple II computer.

To use the shape table in Applesoft programs requires two steps. The first step is loading the shape table into memory. SHLOAD can be used to read the table in from tape, or the code in line 7110 can be implemented to read the table in from disk. The second step is to store the lower two digits of the starting location of the table in $E8 (248), and the upper two digits in $E9 (249).

The code presented is full of REMarks to aid in the understanding of how the algorithms work. Each function is logically grouped into a set of lines. Please contact the author if there are any problems, suggestions, or questions about SHAPER. The text and example shape tables are available on cassette or disk. Contact the author for details.

The remainder of the article is presented in a "user's manual" format, so it can easily be referred to while learning to use the program. After running the program once or twice, the user's manual will seldom be needed.

Shape tables are a unique feature of the Apple II. However, the method of building shape tables described in chapter 9 of the Applesoft manual is time consuming, prone to error, and difficult to master. SHAPER utilizes an automated approach which provides the same sophisticated results in a shorter time with less chance of error.

SHAPER not only allows the user to build tables, but also to add shapes to the table, change shapes in the table, delete them from the table, display them from a table, and save shape tables to disk or load from disk or tape.

Executing SHAPER

SHAPER 2 is written in Applesoft BASIC. The program takes up about 6.5K and uses HGR2. SHAPER can be run in a 32K Apple, but 48K is recommended. Once SHAPER is loaded, type RUN to execute.

While running SHAPER a "Yes/No" question can be answered with any word beginning with a "Y" or "N." If an answer begins with any other letter, then the question is re-asked.

An "APPLESOFT ERROR" is an error caused by Applesoft. SHAPER intercepts the error and prints "APPLESOFT ERROR xx," where xx is the error code as listed on page 81 of the Applesoft Manual. To restart SHAPER hit any key. Each cause of an "APPLESOFT ERROR" is discussed under the function in which it can occur.

A "DOS ERROR xx" is an error that occurs when a table is being saved or loaded using the disk. The xx is the DOS error code as listed on page 114 of the DOS 3.3 manual.

Warning: Do not hit "Reset" during the operation of any of the functions of SHAPER, or the table could be destroyed.

Selecting the Starting Location

Selecting the correct starting location for the table is very important, and because of the importance, SHAPER makes the user double check it. In selecting a starting location, the amount of memory available in the computer and the length of the shape table being manipulated must be considered.

SHAPER 2 overwrites part of HGR1 and uses HGR2, so the logical location for the table is immediately above HGR2 at memory location 24576 (6000 hex). Using this location will allow a table length of 24476 bytes without DOS booted and 13724 bytes with DOS. Both these lengths allow for 100 bytes of character strings stored after HIMEM.

Question: What is the starting location of the table (in decimal)?

Response:	*Action of SHAPER:*
n	Set n as the starting location of the table.
0	Set starting location of the table to 24576.

If n is greater than the highest memory location in the Apple or lower than LOMEM an "APPLESOFT ERROR" will occur. Hit any key to restart SHAPER, and select a different starting location. After selecting the starting location a menu will appear:

Functions available:

0. Exit from SHAPER
1. Build shape table
2. Add shapes to table
3. Change shape in table
4. Delete shapes from table
5. Review shape table
6. SAVE/LOAD shape table

Question: Function?

Response:	*Action of SHAPER:*
0 - 6	Execute the desired function.
> 6	Re-ask question.

If a shape table is not in memory, trying to execute the Add, Change, Delete or Review function will give unpredictable results and usually end with an "APPLESOFT ERROR." Each function is described in the rest of the article.

Build

Build is used to construct a shape table. SHAPER-BUILD will configure the table in the proper format needed to utilize the Applesoft shape table commands. SHAPER-BUILD builds the index portion of the table along with the shape definition portion.

Question: Number of shapes going into table?

Response:	*Action of SHAPER-BUILD:*
0	Return to menu.
1 - 255	Set table for number of shapes.
> 255	Re-ask question. (Maximum number of shapes is 255.)

SHAPER-BUILD is now ready to accept vector definition numbers to define shapes that are going into the table.

Definition of Vectors:

0 - move up
1 - move right
2 - move down
3 - move left
4 - plot and move up
5 - plot and move right
6 - plot and move down
7 - plot and move left

Question: (Will repeat until shape definition is completed.) Vector 1 – 1 =

Response:	*Action of SHAPER-BUILD:*
0 - 7	Use as vector definition in shape.
8 - 9	Display definition of vectors.
ESC	Switch from TEXT to HGR2 and *vice versa.*
X	Erase last vector input.

Anything else is ignored.

To end a shape definition type in three vectors of zero. Because of how Applesoft handles a shape definition, a shape cannot be defined to move up three times in a row, or move up twice and use a plotting vector. Example (Vector n, Vector n + 1, Vector n + 2):

n	**n+1**	**n+2**	
0	0	0	Will end the shape definition.
0	0	(4-7)	Will end the shape definition.
0	0	(1-3)	Move up twice and move the last vector direction.

After the shape has been defined it will be displayed on high-resolution graphics page two at the coordinates: X = 139, Y = 79. These coordinates can be changed by altering line one of SHAPER, which also changes the point that the REVIEW function uses.

The "ESC" key will switch the display between TEXT and high-resolution graphics page two, so the shape can be viewed while it is being built. If a move vector is used to go over a point that has been plotted, the point will disappear, but the point still exists in the shape definition and it will be displayed when the shape is completed.

The "X" key can be used to erase the last vector input. The input buffer will only hold 100 inputs. This allows for 100 vectors to be erased per shape. If more than 100 mistakes are made on a shape, then the shape can be ended and restarted.

When the shape is completed, the shape will be displayed on HGR2, and the bell will sound. SHAPER is waiting for a "Y" or "N" for approval or disapproval of the shape. The question will not be seen and the ESC key is inoperative at this time.

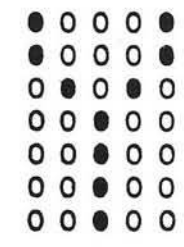
Question: Is shape OK?

Response:	*Action of SHAPER-BUILD:*
Y	Allow user to define the next shape.
N	Allow the user to redefine the last shape.

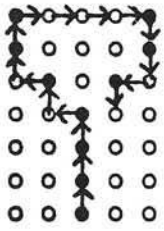
Warning: Do not start a shape definition with 0 0 0 or 0 0 (4-7). This will put one byte of zero in the table to define the shape. Applesoft will ignore this zero and use the next bytes in the table to define the shape until another byte of zero is reached. Later, when using Delete and Change functions, more problems could arise. An "APPLESOFT ERROR" will occur if one attempts to build a long table past the end of memory available in the computer.

Example of Build Function

Suppose one wishes to build a shape table of one shape. The shape looks like a "Y" in a 5 × 7 format. For starting location reply: 0 (sets starting location to 24576 — 6000 hex). The shape looks like this:



The shape is then drawn with direction vectors:



The vectors would be laid out as shown in figure 1. Type in the vector definition numbers followed by three zeros. Location 6000 hex will show the table as in figure 2.

Figure 1

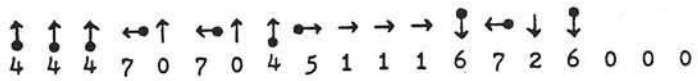
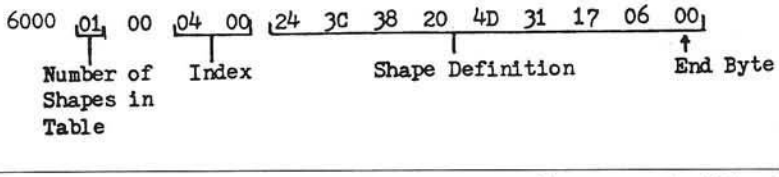


Figure 2



Add

Add allows one to add shapes at the end of an already existing shape table. The number of shapes added cannot make the total shape count in the table go over 255.

Question: Number of shapes adding to table?

Response:	*Action of SHAPER-ADD:*
0	Return to menu.
1 - n	Allow n number of shapes to be added to the table.
NS + n>255	Re-ask question.

(n is the number of shapes adding. NS is the number of shapes in table. NS + n is the number of shapes in the table after the add.)

After telling SHAPER-ADD how many shapes are being added, the vector definitions are typed in as in SHAPER-BUILD.

Change

Change allows a shape definition to be redefined. The new shape is defined as in SHAPER-BUILD and it replaces the one being changed.

Question: Shape to be changed?

Response:	*Action of SHAPER-CHANGE:*
0	Return to menu.
1 - NS	The requested numbered shape is changed.
> NS	Re-ask question.

(NS is the number of shapes in the table.)

After telling SHAPER-CHANGE what shape is to be changed, the vector definitions are typed in as in SHAPER-BUILD.

Delete

Delete allows unwanted shapes to be removed from the table. SHAPER-DELETE removes the shape definition, removes the index, and compresses the table. This compression causes all shapes after the deleted shape to have a smaller definition number in the table.

Example: Suppose shape number 3 is to be deleted.

Before

Shape Number	Shape
1	A
2	B
3	C
4	D
5	E

After

Shape Number	Shape
1	A
2	B
3	D
4	E

Notice after shape 3 is deleted, that shape 3 now defines D, and shape 4 is now E.

Question: (Will repeat until function is completed.) Shape to be deleted?

Response:	*Action of SHAPER-DELETE:*
0	Return to menu if no shapes have been deleted. Compress table and return to menu if shapes have been deleted.
1 - NS	Delete that shape from the table.
>NS	Give error message and re-ask question.
PS	Give error message and re-ask question.

(NS is number of shapes in the table.)
(PS is a previous deleted shape in this execution of SHAPER-DELETE.)

Many shapes may be deleted in one execution of SHAPER-DELETE. The table is not compressed until 0 is typed to end the function. So, in the above example if shape 2 and shape 4 were to be deleted, then 2, 4, and 0 would be typed in to answer the questions. B and D would be deleted.

If there is only one shape in the table, the Delete function cannot be used. Trying this will result in an error message and the question being re-asked. Type in zero to exit from SHAPER-DELETE.

Depending on the size of the table, there will be a delay when typing in zero to end the function. This is when the table is being compressed.

Review

Review will give information about the table and allow the shape table to be displayed. Information given: 1. The starting location of the table; 2. The ending location of the table; 3. The length of the table; 4. How many shapes are in the table.

Question: Shape number (first, last)?

Response:	*Action of SHAPER-REVIEW:*
0 ,L	Return to menu.
F,L (F = L)	Display shape.
F,L (F < L)	Display shape F through shape L.
F,L (F > L)	Re-ask question.

(F is the first shape to be displayed.)
(L is the last shape to be displayed.)

If L is greater than the number of shapes in the table then the shapes from F to the end of the table will be displayed.

If an "APPLESOFT ERROR" occurs when the function is being executed for the first time, then a table does not exist at the given starting location. Insure there was a table at the starting location.

If the table was read in from tape, make sure there was not an I/O error during the read, and that the length of the table was correct.

The point that the shape is drawn at is: X = 139, Y = 79. This point can be changed by altering line one in the program. Altering the coordinates also changes them for the Build function.

While a series of shapes is being displayed, a zero can be typed in to exit the function.

Save/Load

Save/Load allows a shape table to be loaded from cassette tape, loaded from disk, or saved to disk. A function menu will appear:

0. Exit
1. Load from tape
2. Save to disk
3. Load from disk
4. List catalog

Question: Function?

Response:	*Action of SHAPER-SAVE/LOAD:*
0	Return to menu.
1	(See cassette function below.)
2,3	(See disk function below.)
4	Display Catalog from the disk.

Cassette Tape:

Question: How long is table (in decimal)?

Response:	*Action of SHAPER: (Load from tape)*
0	Return to menu.
n	Use as length of table.

Next, the tape should be started. Hit any key to start SHLOAD command, which reads the tape.

If the wrong length is given, then the table will not be loaded at the starting location given. This will cause an "APPLESOFT ERROR" later in the execution of another function. Reload the table using the correct length.

If "ERR" appears on the screen, then an I/O error has occurred during the read. An "APPLESOFT ERROR" will occur. Hit any key to restart the program. Check the tape and recorder for problems.

Disk:

Question: What is the (input/output) file name?

Response:	*Action of SHAPER (Disk):*
Legal File Name	Perform the save or load function.

An illegal file name as defined in the DOS manual will result in a "DOS ERROR" or an "APPLESOFT ERROR." A "DOS ERROR" will return to the Save/Load menu and the function can be retried. An "APPLESOFT ERROR" will return to the beginning of the program. The shape table is not lost. Use the same starting location and the table is recovered.

SHAPER Listing

```
0  DIM BF%(100)
1 X = 139:Y = 79
10  ONERR  GOTO 9000
14  REM  : COMMENTS SHOULD NOT
15  REM  : BE INCLUDED IN THE
16  REM  : EXECUTABLE MODULE.
18  REM
19  REM : SET CONSTANTS
20 X1 = X:Y1 = Y:TW = 2:OE = 1:TF
      = 256:ZE = 0:TR = 3: GOTO 6
     000
1000 KB = BF%(B3):B3 = B3 + OE: IF
     B3 = 101 THEN B3 = ZE
1010  IF B1 = TW THEN  RETURN
1020  GOSUB 1030: RETURN
1027  REM : * SUBROUTINE FOR    *
1028  REM : * VECTOR INPUT      *
1030 N = N + OE
1040  PRINT "VECTOR ";SN;"-";N;"
     = ";
1049  REM :  GET KEYBOARD INPUT
     AND DETERMINE ACTION
1050  HCOLOR= TR: HPLOT X1,Y1:KI =
      PEEK ( - 16384) - 176: HCOLOR=
     ZE: HPLOT X1,Y1: IF (KI < ZE
      AND KI <  >  - 21) OR (KI >
     9 AND KI <  > 40) GOTO 1050
1060  POKE  - 16368,ZE: IF KI = 4
     0 GOTO 1190
1070  IF KI =  - 21 GOTO 1170
1080  IF KI > 7 GOTO 1330
1090  PRINT KI: IF KI > TR
      THEN  HCOLOR= TR: HPLOT X1,Y1

1099  REM : PLOT NEW VECTOR POINT

1100 T6 = KI:B4 = OE: GOSUB 1240
1110 BF%(BP) = KI:BP = BP + OE: IF
     KI = ZE THEN B2 = B2 + OE
1120  IF B2 = TR OR (KI > TR AND
     B2 = TW) THEN B1 = TW: RETURN

1130  IF BP = 100 THEN B1 = OE
1140  IF BP = 101 THEN BP = ZE
1150  IF KI <  > ZE THEN B2 = ZE
1160  RETURN
1169  REM : SWITCH SCREEN MODE
1170  IF S = OE THEN  POKE  - 162
     99,ZE: POKE  - 16297,ZE: POKE
      - 16304,ZE:S = ZE: GOTO 105
     0
1180  TEXT :S = OE: GOTO 1050
1189  REM : ERASE OLD VECTOR
1190 BP = BP - OE: IF BP = B3 - O
     E OR N = OE GOTO 1230
1200  IF BP < ZE THEN BP = 100
1210 T6 = BF%(BP):B4 =  - OE: GOSUB
     1240
1220  HPLOT X1,Y1:N = N - OE: PRINT
     "ERASED": GOTO 1040
1230 BP = BP + OE: PRINT "CAN'T E
     RASE": GOTO 1040
1239  REM : SET UP NEW X AND Y
     VALUES FOR PLOTTING
1240  IF T6 = ZE OR T6 = 4 THEN Y
     1 = Y1 - B4
1250  IF T6 = OE OR T6 = 5 THEN X
     1 = X1 + B4
1260  IF T6 = TW OR T6 = 6 THEN Y
     1 = Y1 + B4
1270  IF T6 = TR OR T6 = 7 THEN X
     1 = X1 - B4
1280  IF X1 < ZE THEN X1 = 279
1290  IF X1 > 279 THEN X1 = ZE
1300  IF Y1 < ZE THEN Y1 = 191
1310  IF Y1 > 191 THEN Y1 = ZE
1320  RETURN
1329  REM  :     PRINT  VECTOR
     DEFINITIONS
1330  PRINT : PRINT "0 : MOVE UP
           4 : PLOT & MOVE UP": PRINT
```

(Continued)

```
    "1 : MOVE RIGHT     5 : PLOT
    & MOVE RIGHT": PRINT "2 : MO
    VE DOWN      6 : PLOT & MOVE
    DOWN": PRINT "3 : MOVE LEFT
       7 : PLOT & MOVE LEFT";: GOTO 1050
1338  REM : * BUILD ROUTINE     *
1340  PRINT "**  BUILD  **": PRINT
    : INPUT "NUMBER OF SHAPES GO
    ING INTO TABLE ? ";NS: IF NS
     < OE GOTO 6050
1350  IF NS >  = TF GOTO 1340
1359  REM : SET UP BEGINNING OF
    SHAPE TABLE
1360  VTAB 24: POKE PP,NS:PP = PP
     + OE: POKE PP,ZE:PP = PP +
    OE:SS = NS * TW + TW + SL:R =
    ZE
1369  REM : SET UP INDEXES INTO
    TABLE
1370 T2 =  INT ((SS - SL) / TF):T
    1 =  INT (SS - SL - T2 * TF)

1380 PP = SN * TW + SL: POKE PP,T
    1:PP = PP + OE: POKE PP,T2:P
    P = SS
1390  VTAB 24:BP = ZE:B3 = ZE:B1 =
    ZE
1400  IF B1 = ZE THEN  GOSUB 1030
    : GOTO 1400
1408  REM : ALL SHAPES ARE BUILT
     AFTER RECEIVING 3 VECTORS
1409  REM : GET 1 OF 3 VECTOR
1410  GOSUB 1000:T1 = KB
1419  REM : GET 2 OF 3 VECTOR
1420  GOSUB 1000:T2 = KB
1430 PE = T1 + T2 * 8
1439  REM : GET 3 OF 3 VECTOR
1440  GOSUB 1000: IF PE = ZE AND
    (KB > TR OR KB = ZE) GOTO 15
    10
1449  REM : CONVERT VECTORS FOR
    EACH BYTE IN THE TABLE
1450 T1 = KB: IF KB > TR GOTO 149
    0
1460 PE = PE + KB * 64: POKE PP,P
    E:PP = PP + OE: IF T1 = ZE AND
    T2 = ZE GOTO 1430
1470  IF T1 = ZE GOTO 1420
1480  GOTO 1410
1490  POKE PP,PE:PP = PP + OE: IF
    T2 = ZE THEN T2 = T1:T1 = ZE
    : GOTO 1430
1500  GOTO 1420
1509  REM : REVIEW THE SHAPE
1510  POKE PP,ZE:PP = PP + OE:N =
    ZE: HGR2 :S = ZE: HCOLOR= TR
    : ROT= ZE: SCALE= OE: DRAW S
    N AT X,Y
1520 BP = ZE:B3 = ZE:B1 = ZE:X1 =
    X:Y1 = Y:B2 = ZE
1530  VTAB 24: INPUT "IS SHAPE OK
     ? ";A$: IF  LEFT$ (A$,OE) =
    "Y" GOTO 1560
1540  IF  LEFT$ (A$,OE) <  > "N" GOTO
    1530
1550  CALL 62450:N = ZE:PP = SS: GOTO
    1400
1560  IF SN = NS GOTO 1580
1570  CALL 62450: PRINT :SS = PP:
    SN = SN + OE: GOTO 1370
1580  IF R <  > ZE THEN  TEXT : PRINT
    "$$ TABLE IS BEING CHANGED $
    $": GOTO 3080
1590  GOTO 6050
1998  REM : * ADD ROUTINE      *
2000  PRINT "**  ADD  **": PRINT

2010  INPUT "NUMBER OF SHAPES ADD
    ING TO TABLE ? ";ND: IF ND <
    OE GOTO 6050
2020 NS =  PEEK (SL):SN = NS + ND
    : IF SN > 255 GOTO 2010
2030  POKE SL,SN:PP = SL
2039  REM : RECOMPUTE INDEXES AND
     MAKE ROOM FOR NEW INDEXES
2040  FOR I = OE TO NS:PP = PP +
    TW:T1 =  PEEK (PP) +  PEEK (
    PP + OE) * TF:T2 = T1 + ND *
    TW:T3 =  INT (T2 / TF):T2 =
     INT (T2 - T3 * TF): POKE PP
    ,T2: POKE PP + OE,T3: NEXT
2050 EF = PP + TW:T2 = T1 + SL
2060 T2 = T2 + OE: IF  PEEK (T2) <
     > ZE GOTO 2060
2070 PP = T2 + ND * TW:SS = PP +
    OE
2080  POKE PP, PEEK (T2):T2 = T2 -
    OE:PP = PP - OE: IF T2 >  =
    EF GOTO 2080
2089  REM : SET UP VARIABLES FOR
    TRANSFER TO BUILD
2090 SN = NS + OE:NS =  PEEK (SL)
    : GOTO 1370
2998  REM : * CHANGE ROUTINE    *

3000  PRINT "**  CHANGE  **": PRINT
    :NS =  PEEK (SL):T1 = NS * T
    W + SL:PP =  PEEK (T1) +  PEEK
    (T1 + OE) * TF + SL
3010  INPUT "SHAPE TO BE CHANGED
    ? ";ND: IF ND < OE GOTO 6050

3020  IF ND > NS GOTO 3010
3029  REM :  REPLACE THE LAST
    SHAPE IN THE TABLE
3030  IF ND = NS THEN R = ZE: GOTO
    3070
3040 PP = PP + OE: IF ND = NS THEN
    R = ZE: GOTO 3070
3049  REM : AN INTERNAL SHAPE IS
    BEING CHANGED
3050 PP = PP + OE: IF  PEEK (PP) <
     > ZE GOTO 3050
3060 PP = PP + OE:T2 = ND * TW +
    SL:T3 =  PEEK (T2) +  PEEK (
    T2 + OE) * TF:T4 =  PEEK (T2
     + TW) +  PEEK (T2 + TR) * T
    F:T5 = T4 - T3:SS = PP - SL:
    SN =  INT (SS / TF):SS =  INT
    (SS - SN * TF): POKE T2,SS: POKE
    T2 + OE,SN:EF = PP:R = OE
3069  REM : INITIALIZE VARIABLES
    SO BUILD CAN CREATE SHAPE
3070 SS = PP:SN = ND:NS = SN: GOTO
    1390
3079  REM : REMOVE OLD SHAPE AND
    ADJUST INDEXES
3080 R = ZE:T1 = PP - EF:T5 = T1 -
    T5:SN =  INT (T3 / TF):SS =
     INT (T3 - SN * TF):NS = ND *
    TW + SL: POKE NS,SS: POKE NS
     + OE,SN: IF T5 > ZE GOTO 31
    10
3090  IF T5 = ZE GOTO 3130
3100 T2 = PP - OE:PE = T4 + SL -
    OE:PP = T4 + SL + T5 - OE:SS
     = OE: GOTO 3120
3110 PE = PP:PP = PP + T5:SS =  -
    OE:T2 = T4 + SL
3120 PE = PE + SS:PP = PP + SS: POKE
    PP, PEEK (PE): IF PE <  > T2
     GOTO 3120
3130 PE = EF + T5:PP = SL + T3:T1
     = PP + T1
3140  POKE PP, PEEK (PE):PP = PP +
    OE:PE = PE + OE: IF PP < T1 GOTO
    3140
3150 PP = NS:EF =  PEEK (SL + TW)
     +  PEEK (SL + TR) * TF + SL - TW
3160 PP = PP + TW:T2 =  PEEK (PP)
     +  PEEK (PP + OE) * TF:T2 =
```

(Continued)

```
      T2 + T5:T1 =  INT (T2 / TF):
      T2 =  INT (T2 - T1 * TF): POKE
      PP,T2: POKE PP + OE,T1: IF P
      P <  > EF GOTO 3160
 3170  GOTO 6050
 3998  REM : * DELETE ROUTINE   *
 4000  PRINT "**  DELETE  **": PRINT
      :EF =  PEEK (SL):PE = EF * T
      W + SL:PE =  PEEK (PE) +  PEEK
      (PE + OE) * TF + SL:NS = EF
 4010  INPUT "SHAPE TO BE DELETED
      ? ";ND: IF ND <  = ZE THEN  PRINT
      "$$ TABLE IS BEING COMPRESSE
      D $$": GOTO 4080
 4019  REM : ERROR CHECK SO TABLE
      WONT BE DESTROYED
 4020  IF (ND > EF) OR ND < ZE GOTO  4220
 4030  IF NS = OE GOTO 4220
 4040 SN = SL + ND * TW:PP =  PEEK
      (SN) +  PEEK (SN + OE) * TF +
      SL: IF  PEEK (SN) = ZE GOTO  4220
 4049  REM : ZERO INDEX TO SHAPE
 4050  POKE SN,ZE: POKE SN + OE,ZE
 4060  IF  PEEK (PP) = ZE GOTO 4210
 4069  REM : ZERO SHAPE DEFINITION
 4070  POKE PP,ZE:PP = PP + OE: GOTO  4060
 4080 T1 = PE:PP = NS * TW + SL +
      TW:PE = SL + EF * TW + OE
 4088  REM : COMPRESS ZEROS OUT OF TABLE
 4089  REM : LEAVE A SINGLE BYTE
      OF ZEROS BETWEEN SHAPES
 4090 T1 = T1 + OE: IF  PEEK (T1) <
       > ZE GOTO 4090
 4100 PE = PE + OE: IF  PEEK (PE) =
      ZE GOTO 4100
 4110  POKE PP, PEEK (PE):PP = PP +
      OE:PE = PE + OE: IF  PEEK (P
      E) <  > ZE GOTO 4110
 4120  IF PE = T1 GOTO 4170
 4130  IF  PEEK (PE + OE) <  > ZE GOTO
      4110
 4140  POKE PP, PEEK (PE):PP = PP +
      OE
 4150 PE = PE + OE: IF  PEEK (PE) =
      ZE GOTO 4150
 4160  IF PE < T1 GOTO 4110
 4170  POKE PP,ZE: POKE SL,NS:EF =
      ZE:PP = SL + TW:T2 = ZE:T3 =
      NS * TW + TW:T1 = T3 + SL
 4180 T4 = T3 + T2:T5 =  INT (T4 /
      TF):T4 =  INT (T4 - T5 * TF)
      : POKE PP,T4: POKE PP + OE,T
      5:PP = PP + TW:EF = EF + OE:
       IF EF = NS GOTO 6050
 4190 T1 = T1 + OE:T2 = T2 + OE: IF
       PEEK (T1) <  > ZE GOTO 4190
 4200 T1 = T1 + OE:T2 = T2 + OE: GOTO 4180
 4210  PRINT "SHAPE DELETED": PRINT
      :NS = NS - OE: GOTO 4010
 4220  PRINT "** ERROR ** INVALID
      SHAPE NUMBER": PRINT : GOTO  4010
 4998  REM : * DISPLAY ROUTINE *
 5000 NS =  PEEK (SL):T1 = NS * TW
       + SL:T2 =  PEEK (T1 + OE):T
      1 =  PEEK (T1):T1 = T2 * TF +
      T1 + SL
 5009  REM : FIND THE END OF THE
      TABLE
 5010 T1 = T1 + OE: IF  PEEK (T1) <
       > ZE GOTO 5010
 5019  REM : COMPUTE ENDING ADDRS
      (T1), AND LENGTH (T2)
 5020 T1 = T1 + OE:T2 = T1 - SL
 5030  HOME : TEXT : VTAB TR: PRINT
      "   *****   SHAPE TABLE REVI
      EW   *****": PRINT : PRINT "
      TABLE STARTING LOCATION -> "
      ;SL;" DECIMAL": PRINT : PRINT
      "TABLE ENDING LOCATION --->
      ";T1;" DECIMAL"
 5040  PRINT : PRINT "LENGTH  OF T
      ABLE  -------> ";T2;" BYTES"
      : PRINT : PRINT : PRINT "NUM
      BER OF SHAPES IN TABLE-> ";N
      S
 5050  VTAB 22: INPUT "SHAPE NUMBE
      RS (FIRST,LAST) ? ";SN,T5: IF
      SN > NS OR SN > T5 GOTO 5050
 5060  IF SN = 0 GOTO 6050
 5070  IF T5 > NS THEN T5 = NS
 5079  REM : DRAW SHAPE (I) FROM
      THE TABLE
 5080  FOR I = SN TO T5: HGR2 :S =
      ZE: COLOR= TR: SCALE= OE: ROT=
      ZE: DRAW I AT X,Y: VTAB 23: PRINT
      "SHAPE NUMBER ";I
 5089  REM  : CHECK FOR "0" TO QUI T
 5090  FOR T4 = OE TO 150: IF  PEEK
      ( - 16384) = 176 GOTO 5110
 5100  NEXT : NEXT
 5110  POKE  - 16368,ZE:S = OE: GOTO
      5030
 5997  REM : ********************
 5998  REM : * MAIN PROGRAM      *
 5999  REM : ********************
 6000 S = OE: HGR2 : HOME : TEXT :
       VTAB 2: PRINT "**********
       S H A P E R  2  **********
      ": VTAB 3: PRINT  TAB( 2);"*
      "; TAB( 39);"*": PRINT  TAB(
      3);"*"; TAB( 38);"*": PRINT
       TAB( 4);"*"; TAB( 12);"A UT
      ILITY PROGRAM"; TAB( 37);"*"
 6010  PRINT  TAB( 5);"*"; TAB( 14
      );"FOR MANAGING"; TAB( 36);"
      *": PRINT  TAB( 4);"*"; TAB(
      14);"SHAPE TABLES"; TAB( 37)
      ;"*": PRINT  TAB( 3);"*"; TAB(
      38);"*": PRINT  TAB( 2);"*";
      " COPYRIGHT 1980  CLEMENT D.
       OSBORNE"; TAB( 39);"*"
 6020  FOR I = OE TO 40: VTAB 10: HTAB
      I: PRINT "*": NEXT
 6028  REM : GET STARTING LOCATION
       AND POKE FOR DRAW COMMANDS
 6029  REM : GET STARTING LOCATION
 6030  PRINT : PRINT "WHAT IS STAR
      TING LOCATION": INPUT "OF TH
      E TABLE (IN DECIMAL) ? ";SL:
       PRINT : PRINT "DOUBLE CHECK
       STARTING LOCATION !": PRINT
      : INPUT "IS IT CORRECT ? ";A
      $: IF  LEFT$ (A$,OE) <  > "Y
      " GOTO 6030
 6035  IF SL = 0 THEN SL = 24576
 6040 PP =  INT (SL / TF): POKE 23
      3,PP:PP =  INT (SL - PP * TF
      ): POKE 232,PP
 6049  REM : MAIN MENU
 6050  TEXT : HOME : VTAB 3: PRINT
      "FUNCTIONS AVAILABLE:": PRINT
      : PRINT "  0. EXIT FROM SHAP
      ER": PRINT : PRINT "  1. BUI
      LD SHAPE TABLE": PRINT : PRINT
      "  2. ADD SHAPES TO TABLE": PRINT
      : PRINT "  3. CHANGE SHAPE I
      N TABLE"
 6060  PRINT : PRINT "  4. DELETE
      SHAPES FROM TABLE": PRINT : PRINT
      "  5. REVIEW SHAPE TABLE": PRINT
      : PRINT "  6. SAVE/LOAD SHAP
      E TABLE": PRINT : VTAB 20: INPUT
```

(Continued)

```
      "FUNCTION ? ";T1: IF T1 < ZE
       OR T1 > 6 GOTO 6050
6070   CALL 62450:PP = SL:SN = OE:
       HOME : ON T1 GOTO 1340,2000
      ,3000,4000,5000,7000: GOTO 9
      999
6997   REM : ********************
6998   REM : *  I/O ROUTINE     *
6999   REM : ********************
7000   VTAB 2: PRINT "**  SAVE/LOA
      D  **": PRINT : PRINT " 0. E
      XIT": PRINT : PRINT " 1. LOA
      D FROM TAPE": PRINT : PRINT
      " 2. SAVE TO DISK": PRINT : PRINT
      " 3. LOAD FROM DISK": PRINT

7010   PRINT " 4. LIST CATALOG": PRINT
7020 D4$ =  CHR$ (4): INPUT " FUN
      CTION ? ";T2: IF T2 < ZE OR
      T2 > 4 GOTO 7000
7030   HOME : ON T2 GOTO 7040,7060
      ,7100,7120: GOTO 6050
7040   PRINT : INPUT "HOW LONG IS
      THE TABLE (IN DECIMAL) ?";T1
      : IF T1 = ZE GOTO 6050:
7050 T3 =  PEEK (116) * TF +  PEEK
      (115): HIMEM: T1 + SL + OE: PRINT
      : PRINT "START TAPE, WHEN RE
      ADY HIT ANY KEY": GET A$: SHLOAD
      : PRINT : INPUT "HIT RETURN
      TO RETURN TO MENU";A$: HIMEM:
      T3: GOTO 6050
7060   PRINT : PRINT "WHAT IS THE
      OUTPUT FILE NAME ? ": INPUT
      A$:T1 =  PEEK (SL) * TW + SL
      :T2 =  PEEK (T1 + OE):T1 =  PEEK
      (T1):T1 = T2 * TF + T1 + SL
```

```
7070 T1 = T1 + OE: IF  PEEK (T1) <
      > ZE GOTO 7070
7080 T2 = T1 + OE - SL
7090   PRINT D4$;"BSAVE ";A$;",A";
      SL;",L";T2: PRINT : PRINT A$
      : PRINT "  HAS BEEN WRITTEN
      TO DISK": PRINT : PRINT "STA
      RTING LOCATION : ";SL,"LENGT
      H : ";T2: GOTO 7140
7100   PRINT : PRINT "WHAT IS THE
      INPUT FILE NAME ?": INPUT A$

7110   PRINT D4$;"BLOAD ";A$;",A";
      SL: PRINT : PRINT A$: PRINT
      "  HAS BEEN LOADED AT ";SL: GOTO
      7140
7120   HOME : PRINT D4$;"CATALOG":
       PRINT D4$
7130   PRINT : PRINT "HIT ANY KEY
      TO CONTINUE": GET A$: HOME :
       GOTO 7000
7140   PRINT : PRINT "HIT ANY KEY
      TO CONTINUE": GET A$: GOTO 6
      050

8998   REM : * ERROR ROUTINES *

9000   IF  PEEK (222) = 254 THEN  RESUME
9005   IF  PEEK (222) > 0 AND  PEEK
      (222) < 16 GOTO 9040
9010   PRINT : PRINT "** APPLESOFT
       ERROR ** "; PEEK (222): PRINT
      : PRINT "HIT ANY KEY TO CONT
      INUE": GET A$: GOTO 1
9040   PRINT : PRINT "** DOS ERROR
       ** "; PEEK (222): GOTO 7130

9999   END
```

MICRO™

DROWNING IN PAPERWORK?

The MAGIC WINDOW word processing system lets you breath easier.

MAGIC WINDOW is the professional tool that will lessen the efforts of communication and improve your productivity!

Secretaries at APPLE, Engineers, writers of leading computer magazines and U.S. government officials have selected and use MAGIC WINDOW over all the other word processors available for the APPLE II computer.

MAGIC WINDOW's overwhelming appeal among experts and novice computer users originates from its simulation of a standard typewriter. Add three types of disk file storage, four way scrolling providing full viewing of 80 column documents, logically placed and easy to learn single key editing commands, and you have a word processor that is truly magical.

Take a relaxing deep breath and ask your local computer store for MAGIC WINDOW.

ANNOUNCING BASIC MAILER

BASIC MAILER is a mailing list merge system designed to take MAGIC WINDOW document files and replace names, addresses or any other section of the document with individual data, creating customized letters, invoices etc. BASIC MAILER uses the same human engineered menu systems used by MAGIC WINDOW.

Together MAGIC WINDOW and BASIC MAILER create an affordable, powerful and professional word processing mailing system. The uses for either of these systems are almost unlimited.

SOFTAPE

10432 Burbank Boulevard • North Hollywood, California 91601 • (213) 985-5763

sensible T.M. software

SENSIBLE SOFTWARE, INC. IS PLEASED TO INTRODUCE...
OUR 1981 COLLECTION OF SUPERIOR SOFTWARE FOR THE APPLE COMPUTER...

APPLESOFT-PLUS STRUCTURED BASIC [APLUS] **$25.00**

32K+, Disk II, ROM/RAM Applesoft, Apple II/Apple II+

APLUS is a 4K machine language utility that adds the following structured programming commands to Applesoft basic: 1) WHEN..ELSE..FIN, 2) UNTIL, 3) WHILE, 4) UNLESS, 5) CASE, 6) SELECT (variable), and 7) (OTHERWISE). Multi-line IF..THEN statements are also supported. APLUS allows the use of "named" subroutines or "procedures". The programmer can now instruct a program to "DO CURVE-FIT" without worrying about the location of the subroutine. APLUS automatically indents "&LIST"ed programs to clarify the logic flow. The APLUS "&CONVERT" command replaces the above structured programming commands with "GOTO"'s and "GOSUB"'s to provide a standard Applesoft program as output. New programs can now be written using "GOTO"-less logic.

APPLESOFT PROGRAM OPTIMIZER [AOPT] **$20.00**

32+, Disk II, ROM/RAM APPLESOFT, Apple II/Apple II+

AOPT is a 2.2K machine language utility that will substantially reduce the size of an Applesoft program without affecting the operation of the program. AOPT automatically: 1) Shortens variable names, 2) Removes remarks, 3) Removes unreferenced lines, 4) Appends short lines together, 5) Removes extra colons, and 6) Renumbers line numbers. AOPT will convert a verbose, well documented, development version of a program into a memory-efficient, more secure, production version of the same program. This is the ORIGINAL and the BEST optimizer on the software market today!

DOS PLUS **$25.00**

32+, Disk II, DOS 3.3, Apple II/Apple II+

DOS PLUS is the software solution for living with both 13-sector (DOS 3.1, 3.2, and 3.2.1) and 16 sector (DOS 3.3) Apple diskettes. DOS PLUS adds 8 new commands to Apple DOS. Three of these are built-in and five are user definable. The built in commands include: 1) ".F" to "flip" between DOS 3.2 and 3.3 (The user need not re-boot and any program that resides in memory will not be affected by the flip. The DOS version can even be changed within a program!), 2) ".S" status command informs you what DOS version is currently active, and 3) ".B" BLOAD- analysis is also provided to inform the user of the starting address and length of the last accessed binary file. DOS PLUS also includes a DOS COMMAND CHANGER program to allow easy customization of Apple DOS commands to suit individual tastes.

DISK ORGANIZER II **—NEW—** **$30.00**

48K, Disk II, Apple II/Apple II +

DO II is the fastest and friendliest utility available today for organizing files on an Apple II diskette. DO II provides the following functions: 1) TITLING in Normal, Inverse, Flashing, Lower case, and other characters normally not available, 2) CUSTOM REORDERING of the directory, 3) ALPHABETIZING, 4) DYNAMIC DISPLAY of ALL filenames on a diskette (including deleted files), 5) RENAMING files with the same character options as TITLING, 6) UNDELETING, 7) DELETING, 8) PURGING deleted files, 9) LOCKING (all or some), 10) UNLOCKING (all or some), 11) USE of DOS sectors for increased data storage, and 12) a SIMULATED CATALOG to show the modified directory **before** it is written to the diskette. DO II is completely MENU DRIVEN and attains it's speed by altering a RAM version of the catalog. DO II uses a very powerful SMART KEY to automatically locate the next valid filename for any specified disk operation. Compatible with DOS 3.1, 3.2, 3.2.1, and 3.3 as well as MUSE DOS to allow manipulation of SUPER TEXT files! (Note: Updates available for $5.00 and original diskette.)

PASCAL LOWER CASE **—NEW—** **$25.00**

48K+, Disk II, Apple II/Apple II+, Language System

This is the most recent commercially available LOWER CASE MOD for Pascal for the Apple II. It is the only currently available modification that is compatible with both versions of Pascal (1.0 and 1.1). The Pascal version is automatically checked prior to updating system Apple. If you have any of the hardware lower case adapters you can now input the following characters directly from the keyboard: | ~ ` { } _ and \. This modification does NOT interfere with any of the 'Control' character functions implemented by the Pascal environment and will 'undo' any alterations made by other commercially released modifications.

QUICKLOADER **$25.00**

48K+, Disk II, Apple II/Apple II+ . . . (2 Disks)

If you find yourself doing the same things over and over -- QL will help you do it faster! QL is a unique disk that lets you load DOS, a language card (optionally), and an application program of your choice extremely rapidly. QL boots as a 13 or 16 sector diskette and is easy to set up and use. To change the setup, you merely load your Apple RAM with the new data and use the "RECONFIGURE" option of QL. The next time you boot your QL disk, it will quickly load your new setup (Language Card, DOS, Application program) into your Apple! QL can reduce the time to perform these functions by up to 80%! Now that you've read this, you say "But I can already do all of that!" QL doesn't do anything new -- it just does it MORE CONVENIENTLY and FASTER! Try it, you'll like it!

DISK RECOVERY ["THE SCANNER"] **$30.00**

48K+, Disk II, Apple II/Apple II+

This program is long overdue. You need no longer be concerned with the problem of physically damaged disks. Just as "Apple Pascal" provides a "BAD BLOCK SCAN", DISK RECOVERY will do a complete scan of your Apple diskettes' recording surface. Damaged areas will be "marked" as used in the disk directory so that no attempts will be made to "WRITE" to a bad sector. The VTOC will be completely redone to reflect both the bad sectors and actual disk usage. A complete report is generated advising the user of all corrections. A resulting "DISK MAP" is presented for your review. The greatest advantage of this program over the other versions is that it can be used on either NEWLY INITIALIZED DISKS or disks that ALREADY CONTAIN PROGRAMS as well as the SPEED of analysis. THE SCANNER is fully compatible with both 13 and 16 sector diskettes. This is a must for all Disk II owners!

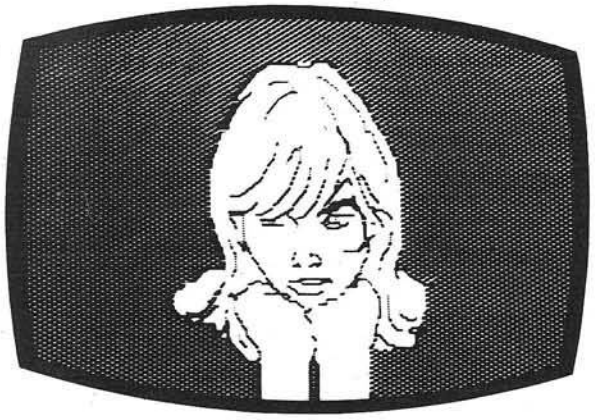
ALSO AVAILABLE:

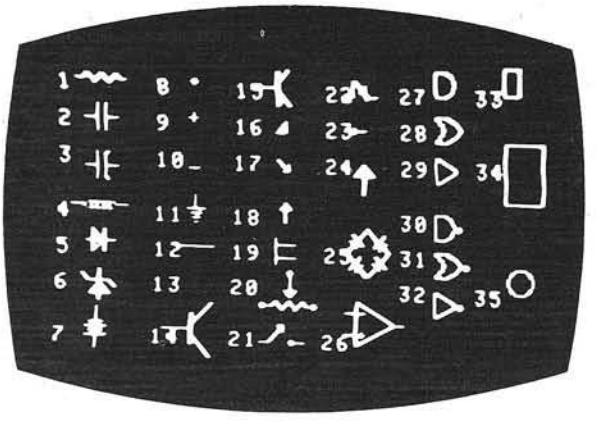
SUPER DISK COPY III **$30.00**
MULTI-DISK CATALOG III **$25.00**
THE NEW PROTECTOR **$250.00**
(Call or Write for Information)
LUNAR LANDER II **$15.00**
MASTER MAZE **$15.00**

SENSIBLE SOFTWARE, INC.
6619 PERHAM DRIVE / W. BLOOMFIELD, MICHIGAN 48033
313-399-8877

VISA and MASTERCARD WELCOME
Michigan Residents add 4% Sales Tax
Please add $1.00 postage & handling for each item ordered.

VersaWriter





What is VersaWriter?

- VersaWriter is an inexpensive drawing tablet for the APPLE II that lets you trace a picture and have it appear on TV display.
- VersaWriter is a comprehensive software drawing package which lets you color in drawings with over **100** different colors.
- VersaWriter is a shape compiler that converts anything on the screen automatically into a standard shape table.
- VersaWriter is a text writer for labeling pictures with text in six colors and five sizes. Use English or Greek, upper or lower case letters.
- VersaWriter is much more! Draw with brush, create schematic drawings, compute area and distance, edit pictures, save, recall and more.

VersaWriter requires ROM APPLESOFT and 48K memory.

$299 Suggested Retail

UNIQUE OFFER

Send us YOUR disk and $1. We will promptly return the disk with a slide package of 10 color pictures drawn with VersaWriter.

☐ Enclosed is $1 and my disk. Send me the slide package.
☐ Send more information including VersaWriter dealers in my area.

DEALER INQUIRIES INVITED.

NAME

ADDRESS

CITY STATE ZIP

Send To: Versa Computing, Inc. • 887 Conestoga Circle • Newbury Park, CA 91320 • (805) 498-1956

List Controller

LIST CONTROLLER provides flexibility and ease of operation when using the LIST command in Applesoft and Integer BASIC and the TRACE command when using the Apple II System Monitor. The programmer can control the speed of the output to the text screen using the game paddle. As an added bonus those programmers using Integer BASIC and the Monitor can interrupt output to the text screen without pressing the RESET key.

Preston R. Black
16 Durham Street
Boston, MA 02115

The process of debugging a program is particularly tedious on the Apple, since there is limited control of the LIST and TRACE functions. Applesoft allows the TRACE and LIST speed to be changed, and the screen output can be interrupted with 'CTRL-C'. However, both Integer BASIC and the monitor lack even these primitive capabilities.

While Applesoft does provide the programmer with these useful debugging aids, there is little flexibility in the system. Once a particular speed is set in Applesoft, there is no way to alter the speed without interrupting the listing and starting over again. In addition, Applesoft does not provide any means by which the programmer can interrupt the output of his listing for his perusal and then continue in a simple fashion. A program which would give the programmer the debugging aids of Applesoft in Integer BASIC and in the Apple monitor, and also provide all three langauges with the flexibility mentioned above would be very useful.

LIST CONTROLLER is a short assembly language program which does this. LIST CONTROLLER uses the Apple game paddles to control the speed of output to the text screen. The programmer thus has the ability to change his output speed from the equivalent of SPEED = 0 to SPEED = 255 at any time during his output. The programmer can therefore speed over those portions of his program which have been debugged, and then slow down to concentrate on those portions of the program with which he is having difficulty. LIST CONTROLLER also allows the programmer the option to output his listing to the text screen one line at a time, or an entire page (i.e. one full screen) at once. All of this can be done without interrupting the LISTing or the TRACE.

LIST CONTROLLER also allows the programmer using Integer BASIC and the Apple monitor to interrupt his listing at any time without the necessity of pressing the RESET key.

How it Works

LIST CONTROLLER consists of four interconnected routines. The first of these is PDDLRD, which controls output speed, using paddle #0 as the controller. The Apple game paddles are analog inputs connected to 150K ohm variable resistors. The variable resistance between each input and the +5 volt power supply can be used as a timing circuit. As the resistance of the input varies, the timing characteristics of its corresponding time circuit changes accordingly. When the timing loops of the paddles are reset, all the paddle locations (-16284 ($C064) to -16281 ($C067)) become greater than 128 (that is, their high order bit is set). The time for these values to drop below 128 is directly proportional to the setting of the game paddle associated with that location. By polling the game paddle location and counting until it goes below 128, we can get a number relative to the setting of the game paddle.

The good old Apple monitor provides us with a routine which does exactly what we want. PREAD (located at $FB1E) polls the paddle pointed to by the x-register, and returns a value from 0 to 255 in the y-register, depending upon the setting of the paddle. The value in the y-register can then be used to initiate the accumulator before jumping to the monitor WAIT ($FCA8) subroutine. This is another useful subroutine which will initiate a delay of a specific amount of time, depending upon the value of the accumulator when this routine is called. The delay, in microseconds, is given by the equation

$$DELAY = 13 + 13.5*A + 2.5*A^2$$

where A is the contents of the accumulator. By interrelating the PREAD subroutine and the WAIT subroutine, output speed is controlled by the game paddle.

The second routine in LIST CONTROLLER is the PAGE routine. This routine outputs the listing one text screen page at a time. The text screen is first cleared by using the HOME subroutine in the Apple monitor. The listing is then output to the text screen. When the screen has been filled, the listing stops until another command is given. To understand how it is determined that the text screen has been filled, one must understand how characters are output to the screen.

Pages 14-17 of the *Apple II Reference Manual* tell us that the text screen occupies pages four through seven of memory. If we examine the diagram on page 16, we see that the lines of the text screen are not ordered sequentially. However, this is not important since this diagram also shows us that the last line of the text screen is at $7D0. And by further knowing that the address of the next line to be used for outputting to the text screen is calculated in BASCALC ($FBC1) and is then stored in BASL($28) and BASH($29), we can poll these addresses until the end of the text screen

is reached (i.e. BASL = $D0 and BASH = $07). We then wait for the next command.

The STEP routine is a very simple routine which polls the output to the text screen until a 'carriage return' is detected, indicating the end of a line of output. We then go to the DELAY routine which waits for another command.

The final routine in this program is the DONE routine. When this routine is called, the output hooks at CSWL($36) and CSWH($37) are reset to the original monitor output routine at COUT1. It then determines which language the user is using by polling PROMPT($33), the address at which the monitor holds the prompt character. A jump to the warm start of the particular language is then executed.

The well-documented program which is listed should be self explanatory and should help the interested programmer to fully understand the routines. The program can be relocated with a few changes.

How to Use 'LIST CONTROLLER'

Because of the idiosyncracies of the three languages in the standard Apple II, I have had to use three different methods for using this program—one for each language. The program is first BLOADed from disk. If you are in Applesoft, type '&:LIST' to activate LIST CONTROLLER. The colon is necessary for proper handling of the command. In Integer BASIC typing 'CALL 768' will activate LIST CONTROLLER and begin listing your program automatically. To use LIST CONTROLLER from the monitor type (CTRL-Y)xxxxT where xxxx is the address with which you wish to begin tracing. The way the program is written, game paddle #0 is used to control output speed. To change to paddle #1, change the 'LDX #$0' in the PDDLRD routine to 'LDX $#1'.

When I'm not programming or playing with My Apple II, I am a research fellow in the Department of Surgery at the Brigham and Women's Hospital in Boston. I have had my Apple for two years and have taught myself assembly language over the last twelve months. Although I use my Apple primarily for personal entertainment, I have had occasion to use it in my research projects.

Listing 1

```
                ; ***********************************
                ; *                                 *
                ; *        LIST CONTROLLER          *
                ; *             BY                  *
                ; *        PRESTON R BLACK          *
                ; *                                 *
                ; *                                 *
                ; *      TO USE 'BLOAD' FROM THE    *
                ; *      DISK. THEN ACTIVATE BY     *
                ; *      USING THE FOLLOWING        *
                ; *      COMMANDS:                  *
                ; *                                 *
                ; *      APPLESOFT--'&:LIST'        *
                ; *      INTEGER BASIC--'CALL 768'  *
                ; *      MONITOR--'(CTRL-Y)XXXX'    *
                ; *        WHERE XXXX IS THE ADD-   *
                ; *        RESS FROM WHICH YOU      *
                ; *        WOULD LIKE TO START      *
                ; *        TRACING                  *
                ; *                                 *
                ; ***********************************
                ;
                ;
                BASL   EPZ $28
                BASH   EPZ $29
                PROMPT EPZ $33
                CSWL   EPZ $36
                CSWH   EPZ $37
                RETURN EQU $8D
                ESCAPE EQU $9B
                SPACE  EQU $A0
                IPRMPT EQU $BE
                PGBTM  EQU $D7
                APRMPT EQU $DD
                KBRD   EQU $C000
                STRB   EQU $C010
                ABASIC EQU $D43C
                IBASIC EQU $E003
                INLIST EQU $E04B
                PREAD  EQU $FB1E
                HOME   EQU $FC58
                WAIT   EQU $FCA8
                COUT1  EQU $FDF0
                RSTORE EQU $FF3F
                SAVE   EQU $FF4A
                MON    EQU $FF69
                ;
                ; SET AMPERSAND (&) JUMP VECTOR FOR APPLESOFT
                ;
03F5                    ORG $3F5
03F5            AMPRSD:
03F5 4C0003             JMP BEGIN
03F8            ;
03F8            ; SET CONTROL-Y JMP VECTOR FOR THE MONITOR
03F8            ;
03F8                    ORG $3F8
03F8            CTRLY:
03F8 4C0003             JMP BEGIN
03FB            ;
03FB            ;
03FB            ; ********* LIST CONTROLLER *********
03FB            ;
03FB            ;
0300                    ORG $300
0300            ;
0300            BEGIN:
0300 A912               LDA #PDDLRD         ; RESET OUTPUT HOOKS
0302 8536               STA CSWL            ; TO ROUTINE AT PDDLRD
0304 A903               LDA /PDDLRD
0306 8537               STA CSWH
0308 A533               LDA PROMPT
030A C9BE               CMP #IPRMPT         ; IS PROMPT '>'?
030C D003               BNE RETRN           ; NO--CONTINUE
030E 4C4BE0             JMP INLIST          ; YES--TO INTEGER BASIC LIST
0311            RETRN:
0311 60                 RTS
0312            ; *
0312            ; *
0312            ; * PADDLE READING ROUTINE
0312            ; *
0312            ; *
0312            PDDLRD:
0312 204AFF             JSR SAVE            ; SAVE REGISTERS
0315 A200               LDX #$00            ; SELECT PADDLE #0
0317 201EFB             JSR PREAD           ; READ PADDLE VALUE IN Y REGISTER
031A 98                 TYA                 ; MOVE VALUE TO ACC
031B 20A8FC             JSR WAIT            ; TO MONITOR DELAY ROUTINE
031E 203FFF             JSR RSTORE          ; RESTORE REGISTERS
0321 20F0FD             JSR COUT1           ; OUTPUT CHARACTER
0324 AD00C0             LDA KBRD            ; POLL KEYBOARD
0327 C99B               CMP #ESCAPE         ; ? 'ESCAPE'
0329 F050               BEQ DONE            ; YES--FINISHED
032B C9A0               CMP #SPACE          ; ? 'SPACE'
032D F06B               BEQ STEPS           ; YES--OUTPUT ONE LINE
032F C98D               CMP #RETURN         ; ? 'CR'
0331 F004               BEQ PAGE            ; YES--OUTPUT ONE PAGE
```

(Continued)

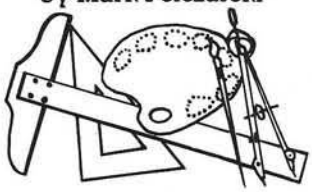
Listing 1 *(Continued)*

```
0333 2C10C0             BIT STRB            ; CLEAR KEYBOARD
0336 60                 RTS                 ; REJOIN OUTPUT
0337                ; *
0337                ; *
0337                ; * ROUTINE TO OUTPUT ONE PAGE TO THE SCREEN
0337                ; *
0337                ; *
0337                PAGE:
0337 2C10C0             BIT STRB
033A A94C               LDA #PAGE1          ; RESET OUTPUT HOOKS
033C 8536               STA CSWL            ; TO ROUTINE AT PAGE
033E A903               LDA /PAGE1
0340 8537               STA CSWH
0342 204AFF             JSR SAVE            ; SAVE REGISTERS
0345 2058FC             JSR HOME            ; CLEAR SCREEN
0348 203FFF             JSR RSTORE          ; RESTORE REGISTERS
034B 60                 RTS                 ; REJOIN OUTPUT
034C                PAGE1:
034C 20F0FD             JSR COUT1           ; OUTPUT A CHARACTER
034F A528               LDA BASL            ; ARE WE AT THE END
0351 4529               EOR BASH            ; OF THE SCREEN PAGE?
0353 C9D7               CMP #PGBTM
0355 F001               BEQ DELAY           ; YES--WAIT FOR ANOTHER INSTRUCTION
0357 60                 RTS                 ; ELSE REJOIN OUTPUT
0358                ; *
0358                ; *
0358                ; * DELAY ROUTINE AND KEYBOARD SERVICING ROUTINE
0358                ; *
0358                ; *
0358                DELAY:
0358 AD00C0             LDA KBRD            ; ?KEY PRESSED
035B 10FB               BPL DELAY           ; NO--WAIT
035D C9A0               CMP #SPACE          ; ? 'SPACE'
035F F039               BEQ STEPS           ; YES--OUTPUT ONE LINE
0361 C98D               CMP #RETURN         ; ? 'CR'
0363 D006               BNE DELAY1
0365 2C10C0             BIT STRB            ; YES--OUTPUT ONE PAGE
0368 4C3703             JMP PAGE
036B                DELAY1:
036B C99B               CMP #ESCAPE         ; ?'ESCAPE'
036D F00C               BEQ DONE            ; YES--FINISHED
036F 2C10C0             BIT STRB            ; IF ANY OTHER KEY
0372 A912               LDA #PDDLRD         ; IS PRESSED THEN
0374 8536               STA CSWL            ; RESET OUTPUT HOOKS
0376 A903               LDA /PDDLRD         ; TO ROUTINE AT PDDLRD
0378 8537               STA CSWH
037A 60                 RTS                 ; REJOIN OUTPUT
037B                ; *
037B                ; *
037B                ; * LIST CONTROLLER EXIT ROUTINE
037B                ; *
037B                ; *
037B                DONE:
037B 2C10C0             BIT STRB
037E A9F0               LDA #COUT1          ; RESET OUTPUT HOOKS
0380 8536               STA CSWL            ; TO REGULAR OUTPUT ROUTINE AT COUT1
0382 A9FD               LDA /COUT1
0384 8537               STA CSWH
0386 A533               LDA PROMPT
0388 C9BE               CMP #IPRMPT         ; IS PROMPT '>'
038A D003               BNE DONE2
038C 4C03E0             JMP IBASIC          ; YES--WARMSTART INTEGER BASIC
038F                DONE2:
038F C9DD               CMP #APRMPT         ; IS PROMPT ']'
0391 D003               BNE DONE3
0393 4C3CD4             JMP ABASIC          ; YES--WARMSTART APPLESOFT
0396                DONE3:
0396 4C69FF             JMP MON             ; DEFAULT TO MONITOR WARMSTART
0399 60                 RTS
039A                ; *
039A                ; *
039A                ; * ROUTINE TO OUTPUT ONE LINE
039A                ; *
039A                ; *
039A                STEPS:
039A 2C10C0             BIT STRB
039D A9A6               LDA #STEP           ; RESET OUTPUT HOOKS
039F 8536               STA CSWL            ; TO STEP ROUTINE
03A1 A903               LDA /STEP
03A3 8537               STA CSWH
03A5 60                 RTS                 ; REJOIN OUTPUT
03A6                STEP:
03A6 C98D               CMP #$8D            ; ?CARRIAGE RETURN
03A8 F004               BEQ STEP1
03AA 20F0FD             JSR COUT1           ; NO--CONTINUE OUTPUT
03AD 60                 RTS                 ; REJOIN OUTPUT
03AE                STEP1:
03AE 20F0FD             JSR COUT1           ; YES--OUTPUT CARRIAGE RETURN
03B1 D0A5               BNE DELAY           ; THEN TO DELAY ROUTINE
03B3 60                 RTS                 ; REJOIN OUTPUT
                        END
```

MICRO™

THE COMPLETE GRAPHICS SYSTEM

by Mark Pelczarski



- **108 color choices for automatic filling and hi-res characters.**
- **Drawing modes include lines, circles, ellipses, automatic filling, and user definable "paintbrushes".**
- **Design hi-res characters and text fonts that can be plotted at any x, y location on the screen.**
- **Design shape tables point-by-point with keystrokes or paddles.**
- **Easily create, edit, and manipulate 3-D objects on the screen. 3-D figures may be entered by coordinates, or by drawing the figure's flat surfaces on the screen, then assembling the parts visually.**
- **Thorough, easy to use manual even shows how to use graphics and machine language routines in your programs. only $59.95**

Also available:.

THE 100-COLOR DRAWING SYSTEM contains all of the 2-dimensional functions of the above.

THE 3-D DRAWING SYSTEM contains the 3-dimensional utilities. $32.95 each

DOS 3.2 or 3.3, requires Applesoft firmware or language system.

CO-OP software

Post Office Box 432
West Chicago, IL 60185
(312) 231-0912

VISA and MASTERCARD accepted

COMING SOON: more graphics utilities and a tablet version!

Lo-Res Graphics and Pascal

The Apple language card precludes the use of the normal low resolution plotting routines. This article offers a library of assembly language procedures that allows you to plot low resolution graphics using Pascal.

C. Donald Heth
Department of Psychology
The Unversity of Alberta
Edmonton, Alberta, Canada
TG6 2E9

One of the principal attractions of the Apple II is its capacity for expansion. Not only can the Apple owner add a plethora of peripheral devices, but he can also change the very "personality" of his machine. Perhaps the most significant development in this latter connection was the Apple II Language Card. With it, the Apple owner can turn a hobbyist machine into a sophisticated computer system using one of the foremost languages of the computer industry, Pascal. As discussed by John Mulligan (MICRO, 29:59), Pascal provides the programmer with a very powerful set of programming tools. The Apple II owner interested in serious applications of his machine will find Pascal well worth considering.

However, he may find one feature disappointing. Because the Language Card functionally replaces the Apple monitor, it gates out many of Apple's intrinsic features. The feature that the new Pascal owner is likely to miss is the Apple's low resolution graphics capability. While Pascal has an excellent graphics handler, it is restricted to the standard high resolution color set. There are no defined procedures to produce the rainbow of colors for which the unmodified Apple is famous. So, while the Pascal user will undoubtedly conclude that Pascal's power is worth the sacrifice, its lack of low resolution graphics is likely to keep him forever nostalgic, or turn him into a closet BASIC user.

To help advance the Pascal movement, I'd like to report on one way I have found to generate low resolution graphics from Apple Pascal. It permits plotting on either page of the low resolution screen.

As readers of the *Apple Reference Manual* know, low resolution graphics in the BASIC Apple are generated from RAM data in the range $400 to $BFF, with the first $400 hex locations designated as "page one" and the second as "page two." This is also the area which contains text data when the Apple is displaying text. Whether text or data are displayed is determined by whether location $C054 or $C055 was last referred to.

This same memory range is reserved in Pascal to display text. The old page one contains the first 40 columns of text; the old page two contains the second. Hitting "Ctrl-A" from the keyboard simply switches between the two pages. But, despite the use of this range for text, the soft switches controlling the display of text or graphics are still functional. To verify this, perform the following experiment.

Type in the program of listing 1. This is a series of assembly language instructions that refer to the soft switches controlling the display of text

Listing 1

```
;-----------------------------------------------;
;                                               ;
; THE FOLLOWING PROCEDURES "THROW" THE          ;
; APPROPRIATE SOFT SWITCHES TO EFFECT THEIR     ;
; RESPECTIVE FUNCTIONS. "COLOR1" AND "COLOR2"   ;
; DISPLAY LOW RESOLUTION GRAPHICS ON THE FULL   ;
; SCREEN OF THE PRIMARY AND SECONDARY PAGE,     ;
; RESPECTIVELY. "TEXT1" DISPLAYS ALL TEXT ON    ;
; THE PRIMARY PAGE.  OTHER COMBINATIONS OF      ;
; SWITCHES COULD BE USED TO DISPLAY MIXED TEXT  ;
; AND GRAPHICS ( SEE PAGE 13 OF THE APPLE ][    ;
; REFERENCE MANUAL ).                           ;
;                                               ;
;-----------------------------------------------;

        .PROC COLOR1
        LDA 0C050
        LDA 0C052
        LDA 0C054
        LDA 0C056
        RTS

        .PROC COLOR2
        LDA 0C050
        LDA 0C052
        LDA 0C055
        LDA 0C056
        RTS

        .PROC TEXT1
        LDA 0C051
        LDA 0C054
        RTS

        .END
```

or graphics. The instructions under the heading .PROC COLOR1 turn on the graphics of page one; those under .PROC COLOR2 and .PROC TEXT1 turn on page two graphics and page one text, respectively. For those readers who have not yet used the assembly langauge capabilities of Pascal, these routines will ultimately be integrated into the main Pascal program and called as procedures.

After you have typed listing 1, save it in a disk file (e.g., under the name ASMDEMO). Then type in listing 2, which is the Pascal program to use these routines. Notice that there are procedure declarations corresponding to each assembly language routine. These procedures are then called in the main body of the program. Listing 2 should likewise be saved in a disk file.

Now you must use the system assembler, compiler, and linker to integrate the two programs. Follow the instructions given for the example in section 1.9.1.1 of the *Pascal Reference Manual*. Run the linked program. The system responds with its usual "RUNNING..." message. Hit carriage return, and you should see an immediate switch to the bar pattern of a non-cleared low resolution screen. The irregular colors in the upper left, of course, are the color translations of the message characters. The next carriage return switches to page two; the pattern in the upper left disappears. Finally, the next carriage return will return you to text mode and end the program.

Listing 2

```
PROGRAM SWITCHDEMO;

VAR
  RESPONSE : CHAR;

PROCEDURE COLOR1; EXTERNAL;

PROCEDURE COLOR2; EXTERNAL;

PROCEDURE TEXT1; EXTERNAL;


BEGIN

  READLN(RESPONSE);
  COLOR1;
  READLN(RESPONSE);
  COLOR2;
  READLN(RESPONSE);
  TEXT1;

END.
```

Listing 3

```
PROGRAM COLORDEMO;

VAR
  RESPONSE : CHAR;
  X : 0..39;
  Y : 0..47;
  COLOR : 0..15;

PROCEDURE COLOR1; EXTERNAL;

PROCEDURE COLOR2; EXTERNAL;

PROCEDURE TEXT1; EXTERNAL;

PROCEDURE PLOT2(X,Y,COLOR: INTEGER); EXTERNAL;

BEGIN

  COLOR2;

  FOR X := 0 TO 39 DO
    FOR Y := 0 TO 47 DO
      PLOT2(X,Y,0);

  TEXT1;

  REPEAT
    GOTOXY(0,3);
    WRITELN('          ');
    GOTOXY(0,1);
    WRITELN('ENTER X, Y, AND COLOR');
    WRITELN('HIT <CR> TO DISPLAY PLOT; <CR> TO RETURN');
    READLN(X,Y,COLOR);
    COLOR2;
    PLOT2(X,Y,COLOR);
    READLN(KEYBOARD,RESPONSE);
    TEXT1;
    WRITELN('TYPE ''0'' TO QUIT; SPACE TO CONTINUE');
    WRITELN('THEN HIT <CR>');
    READLN(RESPONSE);
  UNTIL RESPONSE = '0';

END.
```

So, Apple Pascal can generate low resolution colors—if we can put the right data into the right locations of the low resolution pages. Normally, this computation is performed by the Apple monitor, but again, it has been disabled by the Language Card.

The solution I suggest is an assembly language routine like the ones used to switch the colors on and off. Listing 3 contains such a procedure, labeled as PLOT2. It is set up to accept the X and Y coordinates, and the color to be plotted, as parameters. This gives the user closer

Listing 4

```
          .MACRO POP     ; SAMPLE MACRO TO POP 16 BIT
          PLA            ; WORD FROM TOP OF STACK.
          STA %1
          PLA
          STA %1+1
          .ENDM

          .PROC PLOT2,3
;-----------------------------------------;
;                                         ;
;                PLOT2                    ;
;                                         ;
; THIS PROCEDURE TAKES AS ITS PARAMETERS AN X ;
; COORDINATE (0..39), A Y COORDINATE (0..47), ;
; AND A COLOR (0..15) AND PLOTS A LOW RESOLU- ;
; TION BLOCK ON THE SECONDARY PAGE ( SEE PAGES ;
; 17-18 OF THE APPLE ][ REFERENCE MANUAL FOR  ;
; COLOR AND COORDINATE ASSIGNMENTS ).         ;
;                                         ;
```

(Continued)

Listing 4. *(Continued)*

```
; THE ROUTINE CAN BE USED TO PLOT ON THE PRI-  ;
; MARY PAGE BY ALTERING THE INSTRUCTION        ;
; LABELED "PAGENUMB" TO READ "ORA #04".        ;
;                                              ;
; LOCATIONS 0005 AND 0006 ARE USED FOR         ;
; TEMPORARY STORAGE.                           ;
;                                              ;
;----------------------------------------------;

RETURN    .EQU 0          ; USE LOCATION 0000 TO
          POP RETURN      ; STORE RETURN ADDRESS.

          PLA             ; PUT LOWER-ORDER BYTE OF
          STA CLR         ; COLOR ARGUMENT IN CLR,
          PLA             ; AND DISCARD HIGH BYTE.

          LDA #0F0        ; SET UP BINARY MASK
          STA MASK        ; AT MASK.

          PLA             ; GET LOWER-ORDER BYTE
          PHA             ; OF Y COORDINATE AND
          PHA             ; DUPLICATE ON STACK.
          AND #30         ; STRIP BITS 0-3 AND 6-7.
          LSR A           ; TRANSFER BITS 4-5 TO
          STA 5           ; POSITIONS 3-4.
          ASL A           ; THEN DUPLICATE PATTERN
          ASL A           ; IN POSITIONS 5-6,
          ORA 5           ; AND STORE IN 0005.
          STA 5

          PLA             ; GET LOW BYTE OF Y COOR.
          AND #0E         ; STRIP BITS 0 AND 4-7.
          ROR A           ; TRANSFER RESULT TO
          ROR A           ; CARRY AND BITS 0-1.
PAGENUMB  ORA #08         ; COMBINE WITH PAGE LIMIT
          STA 6           ; AND STORE IN 0006.

          ROR A           ; TRANSFER ORIGINAL BIT 1
          AND #80         ; TO POSITION 7 AND
          ORA 5           ; COMBINE WITH CONTENTS
          STA 5           ; OF 0005.
          PLA             ; GET LOW BYTE OF Y COOR.
          AND #01
          BEQ SKIP        ; IF ODD,
          ASL CLR         ; THEN SHIFT COLOR CODE
          ASL CLR         ; TO POSITIONS 4-7
          ASL CLR         ; OF CLR,
          ASL CLR
          LDA #0F         ; AND CHANGE MASK.
          STA MASK

SKIP      PLA             ; DISCARD HIGH BYTE OF Y.
          TYA             ; SAVE Y REGISTER.
          STA YSTOR
          PLA             ; GET LOW BYTE OF X COOR,
          TAY             ; TRANSFER TO Y REGISTER,
          PLA             ; AND DISCARD HIGH BYTE.

          LDA @5,Y        ; GET BYTE ON SCREEN.
          AND MASK        ; ERASE OLD COLOR.
          ORA CLR         ; COMBINE NEW COLOR
          STA @5,Y        ; AND PLOT.
          LDA YSTOR       ; RETURN Y REGISTER.
          TAY
          LDA RETURN+1    ; GET RETURN ADDRESS
          PHA
          LDA RETURN
          PHA
          RTS             ; AND RETURN.

CLR       .BYTE
MASK      .BYTE
YSTOR     .BYTE

          .END
```

control over the color than that afforded by the routine which comes with Applesoft. I won't go into the details of listing 3, except to note that, through bit manipulations, it transforms the parameters passed to it by the host program into the appropriate addresses in the page two memory range. If you'd like to plot on page one, change the line labeled PAGENUMB to ORA #04. You could even rewrite listing 3 with the modification and put it under the heading .PROC PLOT1. Then you could plot a point on either page by executing PLOT2 or PLOT2.

Save and assemble listing 3 under some name. It can now be linked, along with listing 1, into a Pascal program which uses the procedure PLOT2 (X,Y,COLOR). I've written listing 4 as an example. To use it, type, save, and compile it, and then link it (as the "host file") to the assembled versions of listings 1 and 3 (as the "lib files"). When you run it, it will begin by visibly clearing the screen. Then it will switch back to text to ask for the coordinates and color of the point you wish to plot. Type the X coordinate, Y coordinate, and color number, each separated by spaces. Hit carriage return, and the program will plot the low resolution point on the screen. The next carriage return will bring you back to text to plot another point. The Pascal program isn't very elegant, but it should help to demonstrate how these routines can be used.

Hitting "Ctrl-A" during the program seems to interfere with the plot; I haven't located the reason yet. Also, the plotting routine could be written in such a way that the color of the point is determined by a global variable in the Pascal host program, rather than by a procedure parameter. This would be similar to the way plotting routines are handled in Applesoft. This could be done by the assembler directive .PUBLIC, which allows communication between the assembler and Pascal programs through common variables. Finally, I suggest that listing 3 be adapted as BASIC routine. Using it, one could plot on page two directly, without having to move chunks of data from page one. Just how it could be adapted is left as an exercise for the reader.

I've found Pascal to be an extremely attractive language. Before I developed these routines, however, I occasionally missed the low resolution graphics of Applesoft. Now, it's nice to know you can have your Apple and eat it too.

Donald Heth is an Assistant Professor at the University of Alberta. He is interested in microcomputers as tools for psychological research.

MICRO™

the BEST keyboard buffer

Lazer MICRO SYSTEMS INC

& SHIFT KEY UPPER/LOWER CASE CONTROL

$119.95

+ More buffer than others.
+ Clear buffer control.
+ SHIFT key entry of upper/lower case.
+ Easy CTL key access to special chars " | ~ ` { } _ | \ □".
+ Allows BASIC programs with standard INPUT to support Lower Case without software modification.
+ A lower case adapter is required to display lower case.

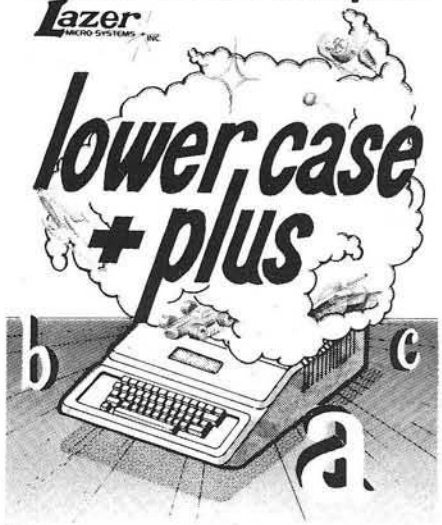
◁ AND ▷

Separately, they have more features and out perform all the rest. But together as a team they perform even better. Look for the Graphics +Plus soon. It's a RAM based character generator to compliment the Lower Case +Plus. It will allow you to define the character set to your needs. You could load German, French, Scientific, Engineering or any other special characters into the Graphics +Plus and use it as if the Apple II was designed specially for that application. And that's not all. If you define the characters as graphics, you can do extremely fast HI-RES type graphics on the text screen without all those cumbersome and slow HI-RES routines and 8K screen. For all the details on this triad of products, send for our free booklet "Lower case adapters and keyboard buffers from the inside out". This booklet gives all the details about lower case adapters and keyboard buffers in general. It also has a section on the Graphics +Plus (RAM based character generator).

Lazer MICRO SYSTEMS INC.
1791-G Capital
Corona, CA 91720
(714)735-1041

the BEST lower case adapter

Lazer MICRO SYSTEMS INC



GRAPHICS & LOWER CASE CHARACTER GENERATOR FOR THE APPLE II COMPUTER

$69.95

+ Normal & Inverse Lower Case.
+ 2 complete character sets on board.
+ Graphics character font built in.
+ Expansion socket allows access to external character sets.
+ 2716 EPROM compatible char generator.
+ More supporting software. (on diskette)
+ Keyboard +Plus & Graphics +Plus designed around the Lower Case +Plus.

DOSOURCE 3.3 for the Apple II

A source listing of DOS 3.3

Disassembled & commented by Randy Hyde

We took our DISASM/65 disassembler program, disassembled Apple's DOS 3.3, and added meaningful lables and comments to create DOSOURCE 3.3, a perfect companion to "Beneath Apple DOS" by Don Worth and Pieter Lechner*. DOSOURCE clearly lists each routine used by Apple DOS.

DOSOURCE is a LISA 2.5 compatible source listing of DOS 3.3. LISA 2.5 owners can load and reassemble DOS at other locations for special applications (such as in a RAM card). DOSOURCE is also a text file that can be loaded into your favorite assembler and converted for use with it. DOSOURCE is also an assembled listing that you can dump to a printer for reference purposes.

With DOSOURCE you can:

-> Reassemble DOS 3.3 at different addresses.
-> Utilize several useful routines found within DOS, such as decimal input and output. Many routines within DOS are as useful as routines found within the Apple monitor...only you didn't know about them until now!!
-> Remove portions of DOS, that you may not need, freeing memory for program use. Most programs do not need the "RENAME", "INIT", "BSAVE", "BRUN", "BLOAD", "CATALOG", etc. commands while they are running. As much as 4K can be removed from DOS without affecting your programs operation. Think about it the next time you get a MEM FULL error or need to declare an array that's just a little bit too big.
-> Learn lot's of 6502 programming tricks - DOS 3.3 is full of 'em. And you can learn them by studying the source listing.
-> Make "Patches" to DOS 3.3 and understand exactly whats going on. No more "guessing game" resulting in unreliable software.

SPECIAL INTRODUCTORY PRICE $39.95
with "Beneath Apple DOS" $55.00

* Beneath Apple DOS is published by Quality Software. Suggested list $19.95

DISASM/65

by Randy Hyde

DISASM/65 is a LISA compatible 6502 disassembler for the Apple II. DISASM/65 takes unadorned machine code and converts it to an understandable assembly language text file. DISASM/65 allows users to disassemble 6502 instruction codes, HEX data, string data, address data, stack data, and more! DISASM/65 is by far the most powerful 6502 disassembler available for the Apple II. In fact, we used it to disassemble DOS 3.3 for our DOSOURCE package. Over 500 happy users bought DISASM/65 for $24.95 without the source listing (The source listing was available for $35.00 extra). Now, for a limited time, you get both the DISASM/65 program and the source listing for $29.95 (DISASM/65 sources are in a LISA 2.x compatible format). Complete documentation included.

Lazer MicroSystems' products are in computer stores all across the country. However, if you cannot locate one, you can order direct from us at 1719-G Capital, Corona, Calif. 91720.

* California residents must add 6% sales tax.
* Master Card & Visa (W/all vital info) welcome.
* Allow 2 weeks additional for checks to clear.
* Orders outside U.S.A. add $15.00 for shipping & handling.

Lower Case +Plus, Keyboard +Plus, and Graphics +Plus are trademarks of Lazer MicroSystems, Inc., Corona, Calif.

Apple II and Applewriter are trademarks of Apple Computer, Inc., Cupertino, Calif.

Dollars & Sense Revisited

This article describes print formatting in Applesoft BASIC using a MID$ statement.

David T. Delli Quadri
Box 1736
Steamboat Springs, Colorado 80477

One of the great joys of programming in BASIC is the way that one can solve the same problem in many different ways. Sometimes the thinking set one brings to the problem dictates the kind of solution one seeks. One often finds the simple solution to a problem while working on something quite different. I have been writing a program to act as a ski race secretary — shuffling, sorting, and printing the various lists of names and times generated during the course of a two-day ski race. One of the problems which I had to address was the output of times to the hundredth's decimal place, a problem similar to that discussed by Barton M. Bauers in his article "Business Dollars and Sense in Applesoft" which appeared in the August 1980 issue of MICRO (27:65). I was surprised to find how dissimilar our solutions were.

The problem, you will remember, is two-fold. First, in Applesoft one does not have a PRINT USING command or a print formatting capacity. Zeros trailing a decimal point, and the decimal point itself, will be omitted where no value follows the decimal. Both of our solutions involve tailoring an alphanumeric variable to suit our needs. Second, the tendency of rounding errors to crop up in the seventh significant digit must be overcome in handling cents. It would never do to output .2999997, rounded to .29 when the answer was really .30.

Bauers chose to handle this problem at input, converting and carrying his variables as integers. They must, of course, be reconverted at output. I chose to eliminate this step, and put the correction in as I set up the alphanumeric-tailored variable.

My solution is, where N is the number to manipulate, and N$ is the tailored alphanumeric to be printed:

```
10 N$ = STR$(N + 1.005 -
   SGN(N) * INT(ABS(N)))
20 N$ = STR$(SGN(N) *
   INT(ABS(N))) +
   MID$(N$,2,3)
```

The Applesoft BASIC will return a 5E-03 when 0 + .005 is used, and this will be carried into the string. To protect against this I use a 1.005 adder for correction, then ignore the first digit. It is also advisable to use a fix-rounding situation rather than depend on the INT() function which will round *down* with negative values.

Listing 1 is a simple program which illustrates the above two lines as used in a program.

Dave Delli Quadri works as a contractor in the summer and a micro-programmer in the winter. Specializing in computer applications for ski competition, his programs have been used by the U.S. and Canadian Ski Associations for both jumping and alpine tournaments.

Listing 1

```
10  REM     A CHECKBOOK BALANCING
20  REM     PROGRAM TO ILLUSTRATE
30  REM     A SOLUTION TO A PRINT
40  REM     FORMATTING PROBLEM...
50  REM
60  HOME
70  INPUT "ENTER CURRENT BALANCE  ";BALANCE
80  PRINT : PRINT "ENTER CHECK AS     -NN.NN"
90  PRINT : PRINT "ENTER DEPOSIT AS    NN.NN"
100  PRINT : HTAB 20: INPUT "";CHECK
105 BALANCE = BALANCE + CHECK
110 BA$ =  STR$ (BALANCE + 1.005 -  SGN (BALAN
CE) *  INT ( ABS (BALANCE)))
120 BA$ =  STR$ ( SGN (BALANCE) *  INT ( ABS (
BALANCE))) +  MID$ (BA$,2,3)
130  PRINT : PRINT "NEW BALANCE IS  $ ";BA$
140  PRINT : INPUT "ANOTHER ENTRY?  ";AN$
150  IF  LEFT$ (AN$,1) = "Y" THEN 80
160  END

]RUN
ENTER CURRENT BALANCE  156.89
ENTER CHECK AS     -NN.NN
ENTER DEPOSIT AS    NN.NN
```

MICRO™

A-STAT

A STATISTICAL ANALYSIS AND FILE MAINTENANCE SYSTEM FOR THE APPLE II™ MICROCOMPUTER

As a Subset Language of **P-STAT™** 78...
A-STAT™ 79 computes:

FREQUENCIES
BI-VARIATE TABLES - CHI SQUARES
CORRELATION MATRICES
MULTIPLE REGRESSION
RESIDUALS
APPLE PLOT INTERFACE
APPLE FILE CABINET INTERFACE
FILE SORT
AGGREGATION
REPORT WRITING
COMPLETE TRANSFORMATION LANGUAGE
READS VISICALC FILES

A-STAT™ 79
Uses Standard DOS Text File and EXEC's
48K Version — All programs in Applesoft™

A-STAT™ 79 is available from:

ROSEN GRANDON ASSOCIATES
296 PETER GREEN ROAD
TOLLAND, CONNECTICUT 06084
(203) 875-3541

A-STAT™ 79 on Disk with 95-page manual... **$125.00**

Apple II tm is a trademark of the Apple Computer Inc.
P-STAT tm 78 is a trademark of P-STAT Inc., Princeton, N.J.
A-STAT tm 79 is copyrighted by Gary M. Grandon, Ph.D.

Singing the file transfer blues? Then...

Get B.I.T.S.™!

Use your Micromodem II,[1] A I O[2] Card, or Apple Comm Card[3] to:

Send data files, BASIC programs, even machine code

to most computers over phone lines.

Copy anything you see

into a 31K buffer then save it on disk and/or print it under your complete control.

Many more features!

See it at your favorite computer store today.

Trademarks held by:
1 - Hayes Microcomputer Products Inc.
2 - S S M
3 - Apple Computer Inc.

B.I.T.S. is a trademark of:

MicroSoftware Systems
7927 Jones Branch Dr. Suite 400
McLean, Virginia 22102
(703) 385-2944

NIKROM TECHNICAL PRODUCTS PRESENTS A DIAGNOSTIC PACKAGE FOR THE APPLE II AND APPLE II + COMPUTER.

"THE BRAIN SURGEON"

Apple Computer Co. has provided you with the best equipment available to date. The Diagnostic's Package was designed to check every major area of your computer, detect errors, and report any malfunctions. ***The Brain Surgeon*** will put your system through exhaustive, thorough procedures, testing and reporting all findings.

The Tests Include:

- MOTHERBOARD ROM TEST
- APPLESOFT ROM CARD TEST
- INTEGER ROM CARD TEST
- MOTHERBOARD RAM TESTS
- DISK DRIVE SPEED CALIBRATION
- DISK DRIVE MAINTENANCE
- DC HAYES MICROMODEM II TEST (HARDWARE & EPROM)
- MONITOR & MODULATOR ROUTINES
- MONITOR SKEWING TESTS
- MONITOR TEST PATTERN
- MONITOR TEXT PAGE TEST
- MONITOR & TV YOKE ALIGNMENT
- LO-RES COLOR TESTS
- HI-RES COLOR TESTS
- RANDOM HI-RES GENERATOR
- SPEAKER FUNCTION TESTS
- SQUARE WAVE MODULATION
- PADDLE & SPEAKER TEST
- PADDLE & BUTTON TEST
- PADDLE STABILITY
- INTERNAL MAINTENANCE
- GENERAL MAINTENANCE
- ON BOARD "HELP"

NEW!

The Brain Surgeon allows you to be confident of your system. This is as critical as the operating system itself. You *must* depend on your computer 100% of it's running time. ***The Brain Surgeon*** will monitor and help maintain absolute peak performance.

Supplied on diskette with complete documentation and maintenance guide.

PRICE: $49.95
REQUIRES: 48K, FP in ROM
1 Disk Drive, DOS 3.2 or 3.3

MasterCard VISA

Nikrom Technical Products
25 PROSPECT STREET • LEOMINSTER, MA 01453

Call Toll-Free Now!
Master Charge & VISA users call: 1-800-835-2246
Kansas Residents call: 1-800-362-2421

Dealer Inquires Invited

APPLE is Registered Trademark of Apple Computer Co
MICROMODEM II is Registered Trademark of D.C. HAYES

DISKETTE UPDATE
IF SERIAL NUMBER IS BELOW 20,000 OR DATED BEFORE 2/15/81, THEN RETURN DISKETTE PLUS $7.00
U.S. $9.00 FOREIGN

SBCS PUTS YOU ON TOP

Organize your business with accounting software from SBCS:

- General Ledger
- Accounts Receivable
- Accounts Payable

The above programs can be used alone or integrated. They include extensive error checking and data entry prompting, numerous reports, departmentalizing, and budgeting. Detailed documentation included.

Get on top of things! Call or write today.

SMALL BUSINESS COMPUTER SYSTEMS
4140 Greenwood, Lincoln, NE 68504 (402) 467-1878

Paddle Hi-Res Graphics

This program lets you draw a background scene on the Apple Hi-Res Graphics display using paddles. Data points are gathered according to the user-definable coordinate space. A series of DATA statements are created on a disk text file which can then be EXECed into any program.

Kim G. Woodward
6526 Delia Drive
Alexandria, Virginia 22310

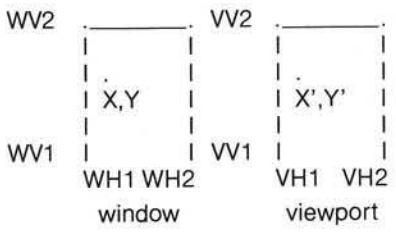
Apple owners are continually searching for shape maker programs. With a shape maker, you can define a shape to meet the needs of your program and then XDRAW the shape on a landscape background, creating exciting games. There has been a myrid of programs to create shapes, but few to create the backgrounds. This program allows you to create that background.

Paddle 0 controls the *X* movement, and paddle 1 controls the *Y* movement. The *X*-, *Y* coordinates from the paddles, in the user's coordinate system, are displayed as well as the status of the "pen." As the movement of the paddles draws a picture on the screen, the coordinates and the pen status are maintained in a vector array. At the conclusion of the background drawing the Apple will respond with a request for a starting line number and a text file name, in which data statements will be stored. In this data statement mode the Apple will create a series of DATA statements starting with the given line number and increment and place them onto the given text file. From the text file the statements may then be EXECd into any program to provide a permanent storage of the required background.

This program makes use of the windowing transform which takes a series of points in a defined "window" and transforms them into a series of points on a defined "viewport." For example, if I have a series of points from a plot whose *X* axis goes from 10 to 20 and whose *Y* axis goes from −15 to +35, this is my defined "window." If I wish to plot them on the upper right portion of the Apple's screen, in a defined "viewport" whose *X* axis goes from 140 to 279 and whose *Y* axis goes from 80 to 0, then I would use the windowing transform:

$$X' = 13.90 * X + 1$$
$$Y' = -1.60 * Y + 56$$

The transform is defined as follows:



$$X' = A * X + B$$
$$Y' = C * Y + D$$

where:

$$A = (VH2 - VH1) / (WH2 - WH1)$$
$$B = VH1 - A * WH1$$
$$C = (VV2 - VV1) / (WV2 - WV1)$$
$$D = VV1 - C * WV1$$

The windowing transform is used to go from a user's coordinate system, the "window," to a device's (plotter, graphics screen, etc.) coordinate system, the "viewport." As you can see from the example, it is not necessary to fill the entire device; the example uses one-fourth of the screen. As a matter of fact, the windowing transform can be used to "zoom" in on a particular area, using clipping to get rid of unplottable lines.

Listing 1

```
10  REM  ENTER X(I),Y(I),P(I),I=1..N
20  REM  X(I) IS X COMPONENT
30  REM  Y(I) IS Y COMPONENT
40  REM  P(I)=1 PEN UP BEFORE MOVE
50  REM  P(I)=0 PEN DOWN BEFORE MOVE
60  POKE 232,192: POKE 233,3
70  FOR I = 960 TO 970: READ Z: POKE I,Z: NEXT I
80  DATA  1,0,4,0,54,196,111,32,86,53,0
90  DIM X(200),Y(200),P%(200)
100  REM  DRAW SCREEN FROM PADDLE
110  TEXT : HOME : ROT= 0: SCALE= 1: HCOLOR= 3
120  INPUT "XMIN VALUE (LEFT SIDE)? ";X1
130  INPUT "XMAX VALUE (RIGHT SIDE)? ";X2
140  INPUT "YMIN VALUE (BOTTOM)? ";Y1
150  INPUT "YMAX VALUE (TOP)? ";Y2
160 AZ = (X2 - X1) / 279:BZ = X1:CZ = (Y2 - Y1) / ( - 159
    ):DZ = Y1 - 159 * CZ
170  HGR :F = 0
180  VTAB 22: HTAB 1: PRINT "PEN IS UP"
190  GOSUB 870: REM  GET PADDLE POSITION
200  XDRAW 1 AT X,Y
210 XL = X:YL = Y
220 N = 0:G = 0
230  REM  REPEAT START
240  GOSUB 870: REM  GET PADDLE POSITION
250  XDRAW 1 AT XL,YL: XDRAW 1 AT X,Y:XL = X:YL = Y
260  IF  PEEK ( - 16384) <  = 127 THEN 840: REM  WAS KEY
    PRESSED?
```

(Continued)

The Program

The program begins by POKEing the "plus" cursor shape into locations starting at $3C0 which are above most page 3 utility routines and out of the way. The program then requests the user's min/max user screen values. If the values entered are 0, 279, 159 and 0, then the data recorded are the screen values themselves. However, if you are entering only one-fourth of an actual background then these numbers must change. The heart of the program is a REPEAT-UNTIL loop; the ESC key is the trigger to leave. Within the loop, as long as no keys on the keyboard are pressed, the loop continuously reads the current paddle cursor position, XDRAWS the cursor over the old cursor position, and XDRAWS the cursor at the new position. In this way the cursor appears to move over the entire screen without disrupting anything that has already been drawn.

The subroutine at the bottom of the program listing reads the cursor position from the paddles. Note that the paddles normally read from 0 to 255, clockwise. The subroutine reads each paddle and converts it through the windowing transform to appropriate screen coordinates.

Note that paddle 1 values are reversed so that with a clockwise rotation the cursor goes "up" instead of "down." The program locks out all but three keys. The space bar changes the "pen" status: if the "pen" is up, no lines are drawn; if it is "down," lines are drawn and data is taken. If the "pen" is down and the return is pressed, then the current cursor position is taken as the data point. If the "pen" was "up," and is now down previous to pressing the return key, then data is stored with an indicator, P%(), of 1, indicating to move to this point, i.e. HPLOT X,Y. If the "pen" was "down," and is now "down" previous to pressing the return key, then data is stored with an indicator, P%(), of 0 indicating to draw a line to this point, i.e. HPLOT TO X,Y. To start taking data, the space bar must be pressed until the "pen" is down and then the return key is pressed. When the ESC key is pressed, further recording of data ceases and the creation of DATA statements begins.

Kim G. Woodward works as an electronic engineer for the U.S. Coast Guard in Washington, D.C. He has been in the computer field for ten years. Woodward's current interests include software engineering and utility type programs.

```
270 K =  PEEK ( - 16384) - 128: POKE  - 16368,0
280  IF K <  > 32 THEN 340: REM  KEY IS SPACE?
290  CALL  - 1059: REM  BEEP SPEAKER
300  IF F = 0 THEN F = 1: VTAB 22: HTAB 1: PRINT "PEN IS
    DOWN": GOTO 320
310  IF F = 1 THEN F = 0: VTAB 22: HTAB 1: PRINT "PEN IS
    UP  ":G = 0
320  REM  CONT.
330  GOTO 840
340  REM  CONT.
350  IF K <  > 27 THEN 640: REM  KEY IS ESC?
360  TEXT : HOME
370  INPUT "NAME OF FILE? ";A$
380  PRINT  CHR$ (4);"OPEN ";A$
390  PRINT  CHR$ (4);"DELETE ";A$
400  PRINT  CHR$ (4);"OPEN ";A$
410  INPUT "STARTING LINE NUMBER? ";I1
420  INPUT "INCREMENT? ";I2
430  PRINT  CHR$ (4);"WRITE ";A$
440  PRINT I1;" REM ";A$
450 I1 = I1 + I2
460  PRINT I1;" DATA ";N;",";X1;",";X2;",";Y1;",";Y2:I1 =
    I1 + I2
470 B$ = " DATA "
480 ZZ = 1
490  FOR I = 1 TO N
500 N1 = I - 1
510  IF ZZ <  > 4 OR I = 1 THEN 570
520 ZZ = 1
530  PRINT I1; LEFT$ (B$, LEN (B$) - 1)
540 I1 = I1 + I2
550 B$ = " DATA "
570 B$ = B$ +  STR$ (X(I)) + "," +  STR$ (Y(I)) + "," +
    STR$ (P%(I)) + ","
580 ZZ = ZZ + 1
590  REM  CONT.
600  NEXT I
610  PRINT I1; LEFT$ (B$, LEN (B$) - 1)
620  PRINT  CHR$ (4);"CLOSE ";A$
630  GOTO 840
640  REM  CONT.
650  IF K <  > 13 THEN 830: REM  KEY IS CR?
660  CALL  - 1059: REM  BEEP SPEAKER
670 N = N + 1:X(N) = AZ * X + BZ:Y(N) = CZ * Y + DZ
680  IF F = 0 THEN 810
690  IF G <  > 0 THEN 750
700  XDRAW 1 AT X,Y
710 G = 1: HPLOT X,Y:XZ = X:YZ = Y
720 P%(N) = 1
730  XDRAW 1 AT X,Y
740  GOTO 800
750  XDRAW 1 AT X,Y
760  HPLOT XZ,YZ TO X,Y:XZ = X:YZ = Y
770 P%(N) = 0
780  XDRAW 1 AT X,Y
790 G = 1
800  REM  CONT.
810  REM  CONT.
820  GOTO 840
830  REM  CONT.
840  REM  CONT.
850  IF K <  > 27 THEN 230
860  END
870  REM  CURSER SUBROUTINE
880 X =  PDL (0) * (279 / 255)
890 Y = 159 - ( PDL (1) * (159 / 255))
900 XP = AZ * X + BZ:YP = CZ * Y + DZ
910  VTAB 22: HTAB 20: PRINT  INT (XP + .5);"     ": VTAB
     23: HTAB 20: PRINT  INT (YP + .5);"     "
920  RETURN
```

Software for the Apple II and Apple II Plus*

BENEATH APPLE DOS

A Technical Manual

By Don Worth and Pieter Lechner

Become an expert on the intricacies of Apple's DOS (Disk Operating System). BENEATH APPLE DOS is the perfect companion to Apple's DOS 3.3 Manual. Containing eight chapters, three appendices, a glossary, an index, and over 160 pages, this manual will serve to completely fill in the many gaps left by Apple's DOS 3.3 Manual. Written for Apple users with DOS 3.3, 3.2 or earlier versions, any Apple disk user would welcome having this carefully written manual at his fingertips.

LEARN . . .

- How DOS 3.3 differs from other DOS versions.
- How disks are protected.
- How to reconstruct a damaged diskette CATALOG.
- How tracks are formatted.
- How to use the disk directly, *without* DOS.
- How to call DOS's file manager.
- How *every* routine in DOS works.
- How to customize DOS to your needs.
- How to overcome DISK I/O ERRORS.
- About the "secret" file types — S and R.

INCLUDES . . .

- Large quantities of excellent diagrams and tables.
- Source listings of useful disk utilities.
- Glossary of over 150 technical terms.
- Exhaustive description of DOS program logic.
- Handy reference card.
- Useful patches to DOS.
- Many programming examples.

Book - $19.95

CROSS-REF by Jim Aalto

Applesoft programmers will be delighted to have this cross reference utility program in their 'tool kit' of software aids. What can CROSS-REF do to speed and facilitate your Applesoft program development? Consider these functions:

VARIABLE CROSS REFERENCE	LINE CROSS REFERENCE
FIND VARIABLE	FIND LINE NUMBER
REPLACE VARIABLE	VARIABLE ONLY LISTING
LINE ONLY LISTING	

Features that make CROSS-REF easy to use include:

- Written in machine language, occupies less than 3K.
- Resides passively in memory while DOS or Applesoft is active.
- Can be loaded with your Applesoft program already resident.
- Very fast — a VARIABLE CROSS REFERENCE for a 16K Applesoft program can start printing in 5 seconds.
- Contains printer format controls and headers for documentation.
- Prints English language error messages.

Cassette - $22.95 **Diskette - $24.95**

LINKER by Don Worth.

Turn your Apple II or Apple II Plus into a powerful and productive software development machine with this superb linking loader/editor package. LINKER does the following and much more:

- Dynamically loads and relocates suitably prepared machine language programs anywhere in RAM.
- Combines a main program with subroutines. You can assemble a subroutine once and then use it with as many main programs as you wish.
- Produces a map of all loaded routines, giving their location and the total length of the resulting module.
- Contains a library of subroutines including binary multiplication and division, print text strings, delay, tone generator, and random number generator.

Linker works with virtually any assembler for the Apple II. Requires 32K of RAM and one disk drive.

Diskette - $49.95
Manual Only - $19.95

FASTGAMMON™ By Bob Christiansen.

Sound, hi res, color, and musical cartoons have helped make this the most popular backgammon playing game for the Apple II. But don't let these entertaining features fool you — FASTGAMMON plays serious backgammon. Runs on any Apple II with at least 24K of RAM.

Cassette - $19.95 **Diskette - $24.95**

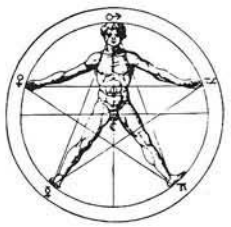
METEOROIDS IN SPACE™

By Bruce Wallace

We have taken our popular space game, formerly called Asteroids in Space, and made some important improvements. To accent these improvements we have given it a new name — METEOROIDS IN SPACE. Your space ship travels through a shower of deadly meteoroids. If your ship is hit, it will be destroyed, so you use your laser gun to blast the meteoroids. Big meteoroids shatter into smaller meteoroids when hit, and the smaller ones are usually faster and just as deadly. From time to time you will encounter an alien space ship whose mission is to destroy you, so you'd better destroy it first. All the action is displayed in fast, smooth, high resolution graphics, accompanied by sound effects. You now can control your ship using one of two options — the Apple game paddles or the keyboard. One of the game paddle buttons controls the laser fire. In METEOROIDS IN SPACE, the spaceship's velocity gradually decreases unless more thrust is applied, adding an element of control. Also new to this version is a hyperspace feature — translate instantly to another spot in the galaxy. The game is over when five of your ships have been destroyed. An additional ship is added for every 10,000 points you score. Runs on any Apple II with at least 32K of RAM and one disk drive.

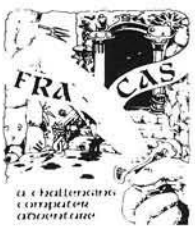
Diskette - $19.95

ASTROAPPLE™ by Bob Male.



Your Apple computer becomes your astrologer, generating horoscopes and forecasts based on the computed positions of the heavenly bodies. This program offers a delightful and stimulating way to entertain friends. ASTROAPPLE produces natal horoscopes (birth charts) for each person based on his or her birth data. Any two people may be compared for physical, emotional, and intellectual compatibility. The program is written in Applesoft BASIC with machine language subroutines. It requires either RAM or ROM Applesoft and at least 32K of memory.

Cassette - $14.95 **Diskette - $19.95**



FRACAS™ by Stuart Smith.

A fantastic adventure game like no other! Up to eight players can participate in FRACAS at the same time. Journey in the land of FAROPH, searching for hidden treasure while warding off all sorts of unfriendly and dangerous creatures. You and your friends can compete with each other or you can join forces and gang up on the monsters. Your location is presented graphically and sound effects enliven the battles. Save your adventure on diskette or cassette and continue it at some other time. Both integer BASIC and Applesoft versions included. Requires at least 32K of RAM.

Cassette - $19.95 **Diskette - $24.95**

BATTLESHIP COMMANDER™ by Erik Kilk and Matthew Jew.

A game of strategy. You and the computer each start out by positioning five ships of different sizes on a ten by ten grid. Then the shooting starts. Place your volleys skillfully — a combination of logic and luck are required to beat the computer. Cartoons show the ships sinking and announce the winner. Sound effects and flashing lights also add to the enjoyment of the game. Both Applesoft and integer BASIC versions are included. Requires at least 32K of RAM.

Cassette - $14.95 **Diskette - $19.95**

Also by Don Worth . . .

BENEATH APPLE MANOR — Adventure. Uses Integer BASIC.
Cassette - $14.95 **Diskette - $19.95**

BABBLE - Fun with words, sound, and graphics.
Cassette - $19.95 **Diskette - $24.95**

QS QUALITY SOFTWARE

6660 Reseda Blvd., Suite 105, Reseda, CA 91335
(213) 344-6599

Now exclusive distributors for products from The Software Factory, Newhall, California

*Apple II and Apple II Plus are trademarks of Apple Computer, Inc.

WHERE TO GET IT: Call us at (213) 344-6599 for the name of the Quality Software dealer nearest you. If necessary you may order directly from us. Mastercard and Visa cardholders may place orders by telephone. Or mail your check or bankcard number to Quality Software, 6660 Reseda Blvd., Suite 105, Reseda, CA 91335. California residents add 6% sales tax. SHIPPING CHARGES: Within North America orders must include $1.50 for first class shipping and handling. Outside North America the charge for airmail shipping and handling is $5.00. Pay in U.S. currency.

True 3-D Images on Apple II

This article describes a program which creates stereo-pair images for viewing without accessory devices. The pair of images can be fused into a three dimensional pattern by placing a piece of paper between the viewer's eyes and the viewing screen so that each eye sees only the appropriate image. With practice the paper is no longer needed. The object used for demonstration is a three dimensional Lissajous figure.

Art Radcliffe
1612 Ferndale
Ann Arbor, Michigan 48104

What we are talking about here is a genuine three-dimensional image such as seen through my grandparents' stereoptican or through more recent systems, such as require colored eye filters or polarizing filters for viewing. The present technique involves not a single projection of the object, a perspective view, but a pair of images which can be fused into one 3-D image without auxiliary contrivances.

The *Scientific American* has published articles accompanied by stereo-pair images, which can be fused into a stereo scene with a little practice. This program was inspired by success with such viewing. Some eye training is required, and some eye strain may be felt initially. What is required is that you stare off into the distance (eyeball axes essentially parallel) while focussing nearby. The muscles which direct your eyeball and the muscles which focus your lens are accustomed to working in a coordinated way for distant or for nearby objects; this muscular habit can readily be broken. It is not at all difficult for me now to glance at a pair of images on the screen from anywhere in the room, and see the 3-D pattern.

Listing 1

```
0  REM NOISY COASTER BY ART RADCLIFFE
10  HOME : POKE 36,12: PRINT "NOISY COASTER"
20  DIM A%(299): DIM B%(299): DIM H%(299): DIM S(299)
30 A = B = C = D = E = F = G = H = I = J = 0
40 K = L = M = N = O = P = Q = T = U = V = 0
50 W = X = Y = Z = 0:R =  - 16336:S = .5:LL = 0
60  GOTO 630
65  REM ------------------------
70  PRINT  CHR$ (7): PRINT  CHR$ (7): FOR A = 0 TO 1000:
 NEXT : PRINT  CHR$ (7)
80  FOR P = 0 TO 299
90 A =  PEEK (R)
100  HCOLOR= 3: REM  FRONT OF TRAIN
110 B = A%(P):C = B%(P):D = H%(P)
120 E = B + 1:F = C + 1:G = D + 1
130  HPLOT B,F: HPLOT E,C: HPLOT E,F
140  HPLOT D,F: HPLOT G,C: HPLOT G,F
150 Q = P - 10
160 A =  PEEK (R)
170  IF Q < 0 THEN Q = P + 289: REM 0<=Q<=360DEG
180  HCOLOR= 0: REM END OF TRAIN
190 B = A%(Q):C = B%(Q):D = H%(Q)
200 E = B + 1:F = C + 1:G = D + 1
210  HPLOT B,F: HPLOT E,C: HPLOT E,F
220  HPLOT D,F: HPLOT G,C: HPLOT G,F
230 A =  PEEK (R): REM REPLOT TRACK ->
240  HCOLOR= 3: HPLOT B,C: HPLOT D,C
250 A =  PEEK (R)
260  FOR Z = 0 TO LL - B%(P): NEXT : REM  TRAIN SPEED
270 A =  PEEK (R)
280  NEXT P
290  PRINT  CHR$ (7)
300  RETURN
305  REM ------------------------
310  FOR P = 0 TO 299: REM ESTABLISH PATTERN
320 X = S(I) + L:Y = 2 * S(J) + T:Z = S(K)
330 M = (C - Z) / (G - Z)
340 A =  INT (S + X + M * (E - X)):A%(P) = A: REM  LEFT X
350 B =  INT (S + Y + M * (F - Y)) - 50:B%(P) = B: REM Y
360 H =  INT (S + X + M * (D - X)):H%(P) = H: REM  RIGHT X
370  HPLOT A,B: HPLOT A + 2,B: HPLOT H,B: HPLOT H + 2,B
380  IF LL < B THEN LL = B
390 I = I + U: IF I > 299 THEN I = 0
400 J = J + V: IF J > 299 THEN J = 0
410 K = K + W: IF K > 299 THEN K = 0
420  NEXT P
430  RETURN
```

(Continued)

The viewing images are produced by running rays from each defined point of the object to points which correspond to eye locations, with the object being behind the screen and the eyes in typical viewing positions. Points are plotted where these rays intercept the display plane.

The object is defined near the origin of an *X, Y, Z* coordinate system, behind the screen plane. We can define object points using the notation: *(X1,Y1,Z1)*, define screen points with: *(X2,Y2,Z2)* and define the eye locations using: *(X3,Y3,Z3)*. *Z2*, the screen distance from the origin, is set at 200 in the program and *Z3*, the eye distance from the origin, is set at 300. *Y3* is the same for each eye: 40; and the *X3* values for the two eyes are 40 and 120. The direction from which the object is viewed can be altered by offsetting *X1* and *Y1*.

Use of proportions leads us to the conclusion that $(X2-X1)/(Z2-Z1) = (X3-X1)/(Z3-Z1)$ and similarly, $(Y2-Y1)/(Z2-Z1) = (Y3-Y1)/(Z3-Z1)$. From these equations we can derive $X2 = X1 + M(X3-X1)$ and $Y2 + Y1 + M(Y3-Y1)$ where $M = (Z2-Z1)/(Z3-Z1)$.

Listing 1 is an embellishment, with sound effects, of the program as originally written, which appears in listing 2.

Within the program there are variable substitutions: (X,Y,Z) = *(X1,Y1,Z1)*, (A,B,C) = *(X2,Y2,Z2)* and (D,F,G),(E,F,G) = *(X3,Y3,Z3)*. A Lissajous pattern was chosen for viewing because it has the convenient property of being restricted to a rectangular area, derived from the property of the sine function, being bounded by 1 and – 1. In the program a raised sine is used by adding 1 (line 64) to avoid negative values. Thus, the *X*-coordinates of the object vary according to one sine function, the *Y*-coordinates of the object vary in a coordinated manner according to a second sine function, and the *Z*-coordinate varies according to a third sine function.

Random numbers are used to achieve an almost infinite variety of patterns. It is fun to watch the pattern take shape; the eye can go on a roller-coaster ride with the leading edge of the pattern as it develops on the screen.

There is an inherent limitation to this method in that the display area is limited to the space between the primary pair of images. Use of prismatic glasses might increase the available object size. The program is written for viewing on a twelve inch diagonal screen. Users with other size displays may want to alter program parameters, first increasing or decreasing the *X* dimension for eye position by altering

Listing 1 *(Continued)*

```
435  REM ------------------------
440 O = 8 *  ATN (1) / 300: REM 360DEG/300
450 N = 40: REM OBJECT SCALE FACTOR
460  FOR A = 0 TO 299
470 S(A) = N * (1 +  SIN (A * O)): REM SINE+1>0
480  NEXT A
490 C = 200: REM X COOR'S OF EYES
500 D = 120
510 E = 40: REM Y COOR'S OF EYES
520 F = 40
530 L = 150: REM  X,Y,Z COOR'S OF OBJECT
540 T = 250
550 G = 300: REM # CYCLES IN X,Y,Z ->
560 U =  INT (1 + 5 *  RND (1))
570 V =  INT (1 + 5 *  RND (1)): IF V = U THEN 570
580 W =  INT (1 + 5 *  RND (1)): IF W = V OR W = U THEN
580
590 I =  INT (300 *  RND (1)): REM START POINTS
600 J =  INT (300 *  RND (1))
610 K =  INT (300 *  RND (1))
620  RETURN
625  REM ------------------------
630  PRINT : PRINT : PRINT "  CREATED BY ART RADCLIFFE,
ANN ARBOR   ": PRINT
640  PRINT : PRINT "PLACE 8 INCH BY 12 INCH CARDBOARD
    "
650  PRINT "BETWEEN SCREEN AND TIP OF NOSE SO EACH  "
660  PRINT "EYE SEES ONLY IT'S IMAGE.  SOME EYE     "
670  PRINT "TRAINING IS NECESSARY.                  "
680  PRINT : PRINT : PRINT : PRINT : PRINT
690  PRINT "PLEASE BE PATIENT WHILE I MEDITATE TO   "
700  PRINT "GET MYSELF READY FOR THIS.............."
705  REM ------------------------
710  GOSUB 440 REM INITIALIZE
720  HOME : HGR : HCOLOR= 3
730 LL = 0: REM LOWEST POINT
740  GOSUB 310 REM LAY TRACK
750  FOR A = 0 TO 999: NEXT
760  GOSUB 70 REM HOLD TIGHT!
770  FOR A = 0 TO 3000: NEXT
780  GOSUB 490 REM REINITIALIZE
790  GOTO 720 REM START OVER
800  END
```

Listing 2

```
0  REM LISSAJOUS FIGURES IN TRUE 3D FOR APPLE-II.
2  REM  PLEASE SHARE YOUR COMMENTS WITH:
4  REM  ART RADCLIFFE
6  REM  1612 FERNDALE
8  REM  ANN ARBOR, MICHIGAN 48104
10  REM  (313)-995-2485
12  REM  SEE REMARKS AT END OF PROGRAM.
14  HGR : HCOLOR= 3: PRINT : PRINT : PRINT "WAIT"
16  DIM S(199)
18 A = B = C = D = E = F = G = H = I = S =. 0
20 J = K = L = M = N = O = P = X = Y = Z = 0
22  GOTO 56
24  FOR P = 0 TO 199
26 X = S(I) + L
28 Y = S(J) + T
30 Z = S(K)
32 M = (C - Z) / (G - Z)
34 A =  INT (S + X + M * (E - X))
36 B =  INT (S + Y + M * (F - Y))
38 H =  INT (S + X + M * (D - X))
40  HPLOT A,B: HPLOT H,B
```

(Continued)

Listing 2 *(Continued)*

```
 42 I = I + U: IF I > 199 THEN I = 0
 44 J = J + V: IF J > 199 THEN J = 0
 46 K = K + W: IF K > 199 THEN K = 0
 48  NEXT P
 50  FOR Z = 0 TO 5000: NEXT Z
 52  HGR
 54  GOTO 22
 56 O = .04 *  ATN (1)
 58 N = 40
 60  FOR A = 0 TO 199
 62 B = A * O
 64 S(A) = N * (1 +  SIN (B))
 66  NEXT A
 68 C = 200
 70 D = 120
 72 E = 40
 74 F = 40
 76 G = 300
 78 T = 250
 80 L = 150
 82 U =  INT (1 + 5 *  RND (1))
 84 V =  INT (1 + 5 *  RND (1)): IF V = U THEN 84
 86 W =  INT (1 + 5 *  RND (1)): IF W = V OR W = U THEN 86
 88 I =  INT (199 *  RND (1))
 90 J =  INT (199 *  RND (1))
 92 K =  INT (199 *  RND (1))
 94 S = .5
 96  POKE 49234,0
 98  GOTO 24
100  REM TO VIEW, SIT WITH FACE ABOUT A FOOT IN FRONT OF THE
SCREEN AND STARE THROUGH THE SCREEN OFF INTO THE DISTANCE.
102  REM THIS PROGRAM PLOTS A LEFT-EYE IMAGE AND A RIGHT-EYE
IMAGE WHICH MUST BE FUSED INTO A SINGLE IMAGE.
104  REM THIS TAKES PRACTICE, AS THE EYES ARE POINTED OFF INT
O THE DISTANCE WHILE THEY ARE FOCUSSED ON THE SCREEN.
106  REM WHEN THE IMAGES ARE FUSED YOU WILLSEE A THREE DIMENS
IONAL PATTERN IN THE CENTER WITH IRRELEVANT IMAGES ON EACH SIDE.
108  REM IN TIME YOU WILL BE ABLE TO GLANCE AT THE SCREEN FRO
M ANYHERE IN THE ROOM AND SEE A 3-D IMAGE.
110  REM PAY ATTENTION TO YOUR EYES; QUIT IF THEY FEEL STRAIN
ED.
112  REM THE PROGRAM GENERATES A 3-D PATTERN BEHIND THE SCREE
N AND RUNS A RAY FROM EACH POINT ON THE PATTERN TO EACH OF YOUR
     EYES IN FRONT OF THE SCREEN.
114  REM AT THE POINT WHERE EACH RAY INTERCEPTS THE SCREEN A
POINT IS PLOTTED.
116  REM THIS IS A SIMPLE MATTER OF PROPORTIONS; YOU MAY READ
 UP ON IT IN AN OPTICS TEXT.  YOU MIGHT ALSO READ ON DIRECTION C
     OSINES IN A SOLID ANALYTIC GEOMETRY TEXT
118  REM OBJECT IS AT ORIGIN; OBJECT IS ORIGINALLY DEFINED WI
TH (X1,Y1,Z1) AND IS REPRESENTED IN PROGRAM BY (X,Y,Z).
120 SIMILARLY FOR THESCREENPLANE:(X2,Y2,Z2) AND (A,B,C)
122  REM EYE LOCATIONS ORIGINALLY DEFINED BY (X3,Y3,Z3) AND B
Y (D,F,G) AND (E,F,G)IN PROGRAM.
124  REM THE OBJECT IS VIEWED FROM OFFSET POINT DEFINED BY T AND
L.
126  REM  THE OBJECT IS CREATED BY DEFINING THE X, Y, AND Z C
OORDINATES BY THREE SEPARATE SINE FUNCTIONS.
128  REM THE POINTS ARE PLOTTED AT ANGULAR INCREMENTS SET BY
'O', LINE 56.
130  REM  THE PROGRAM IS SPEEDED BY PRECALCULATIN A SINE TABL
E WITH SCALE FACTOR 'N' BUILT IN: LINES 60-66.
132  REM I, J, AND K START THE THREE SINE FUNCTIONS AT RANDOM
 PHASES IN THE SINE TABLE.
134  REM U, V, AND W ARE SMALL INTEGERS WHICH DEFINE THE PERI
OD (OR FREQUENCY) OF THE SINE FUNCTIONS.
```

one or both of parameters D and E. It may also be useful to alter the scale factor N.

Interesting 3-D motion displays could be written in machine language; I can also imagine game possibilities, including visual 3-D Tic Tac Toe. Please note the remarks appended to the program (written in Applesoft), which complement the explanatory remarks above.

I have experimented with more general systems using color filters for viewing, and may report on this at some future time. I hope that readers will experiment with this viewing system, perhaps altering parameters of the given program or substituting another object. Data points in three dimensions might be seen as a 3-D swarm of points in which local clusters or correlations could be detected. This is a new way of seeing things.

Art Radcliffe has worked 25 years for IT&T, Radiation-Inc., and Burroughs, during which time he has acquired 32 patents in computer and communciation circuits and systems. He has also worked in optics and holography, whence his interest in generating 3-D images. He has a 48K Apple with twin disks, Zenith color monitor and Silentype printer which he uses as a medium for creative and artistic expression.

MICRO

Need a solution for Floppy Disk or R/W Head problems?

FDL

Floppy Disk Lube

Just THREE drops can:

- Prolong useful disk life.
- Increase head life.
- Allow initialization of "problem" disks.
- Save 'unbootable' disks.
- Reduce 'glitching' problems.
- Cut nuisance problems.

FLOPPY DISK LUBE - ½ oz. WITH APPLICATOR. **$4.00**

Add $1.50 shipping and handling. Ohio residents add 5½% sales tax.

DOSWARE, INC.
P.O. Box 10113
Cleveland, Ohio 44110

LOGICAL SOFTWARE, INC.

ANNOUNCES:

MAIL EXPRESS

A NEW MAIL LIST UTILITY FOR THE APPLE II.

— Up to 2,200 Names per File
— Sort by Company Name, Customer Name, City, State Zip
— Prints Return Addresses
— Merge up to 16 Files
— Easy User Definable Codes for City, State and Zip to Save Time and Disk Space

This is an easy to use professional quality mail list able to handle large or small files.

Introductory Price $49.95
$2.00 Postage & Handling

Logical Software, Inc.
P.O. Box 354
Farmington, MI 48024
(313) 477-2565

®Apple and Apple II are registered trademarks of Apple Computer Inc.

LISP for the Apple II

Pegasys Systems' new P-LISP interpreter is a full implementation of the well-known Artificial Intelligence language. Written in machine code, this powerful interpreter includes the following features:

- Over 55 functions implemented
- Extensive 45-page User Manual
- Full function trace
- Fast, efficient Garbage Collector
- Supplied with function editor and pretty-printer
- Runs in 32 or 48K Apple II or II+ with disk
- ELIZA and other sample programs included
- Special language card version provided

P-LISP is supplied on disk with User Manual for $99.95. The manual is available separately for $10.00. Please specify DOS 3.2 or 3.3.

PEGASYS SYSTEMS, INC.
4005 Chestnut Street
Philadelphia, PA 19104

PEGASYS

Orders only: 800-523-0725
PA residents and inquiries: (215) 387-1500

Pennsylvania residents add 6% sales tax
Apple is a trademark of Apple Computer, Inc.

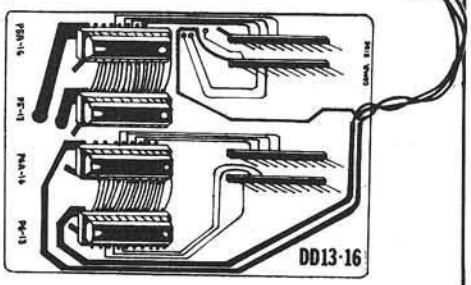
VISA

Good software is no longer a myth.

DOUBLE DOS PLUS

for Apple Computers
$39.00

NEW
DEALER INQUIRIES INVITED!!



201-839-3478

MasterCard VISA

DOUBLE DOS Plus—a piggyback board that plugs into the disk-controller card so that you can switch select between DOS 3.2 and DOS 3.3. Works with the language system eliminating the need in many cases to boot the BASICs disk. Also eliminates the chore of converting all of your 3.2 disks to 3.3
NOTE: APPLE is a registered trademark of APPLE Computer, Inc., Cupertino, CA.

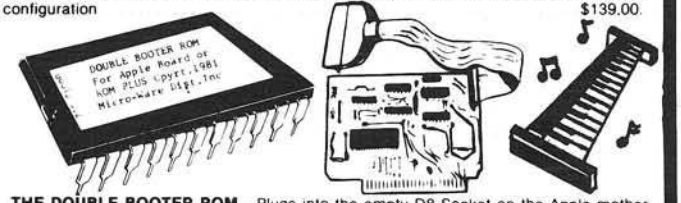
WHY IS DOUBLE DOS PLus better?

- Nothing needs to be soldered, just plug in and go.
- Since all four ROMS are used, all software will work, even early 3.1 DOS.
- Because the ROMS fit on the back of the board, it has the thinnest configuration allowing full use of slot #7
- One set of ROMS is powered up at a time, thus saving power. DOUBLE DOS Plus requires APPLE DOS ROMS
- Full 90-day warranty from TYMAC.

MICRO-WARE DISTRIBUTING INC.
P.O. BOX 113
POMPTON PLAINS, N.J. 07444

OTHER UNIQUE PRODUCTS FROM MICRO-WARE DISTRIBUTING INC.

THE APPLE CARD—Two sided 100% plastic reference card for the Apple computer. Loaded with information of interest to all Apple owners $3.98
PARALLEL PRINTER CARD—PPC-100—A Universal Centronics type parallel printer board complete with cable and connector. This unique board allows you to turn on and off the high bit so that you can access additional features in many printers. Use with EPSON, ANADEX, STARWRITER, NEC, SANDERS, OKI, and other with standard Centronics configuration $139.00.



THE DOUBLE BOOTER ROM—Plugs into the empty D8 Socket on the Apple motherboard or the Integer ROM Card to provide a 13 sector boot without using the BASICs Disk. DoubleBooter may also be used in the MOUNTAIN HARDWARE ROM PLUS board. This chip will not work in a plus machine unless it contains an Integer board or a ROM Plus board $29.00
DISK STIX—Contains 10 dozen diskette labels with either 3.3 or 3.2 designation. Room for program names and type also $3.98

★★★★★★★★★★★★**SOFTWARE**★★★★★★★★★★★★

SUPER SEA WAR—Hires battleship type simulation $13.95
ULTIMATE XFER—A telephone software transfer program, uses DC Hayes Assoc. micromodem $25.00
ROAD RALLYE—Hires driving game with 5 different full screen tracks $15.00
MISSILE CHALLENGER—Hires arcade type game where you defend your cities from falling missiles. 8 levels & writes name & high score to disk $19.95
SUPER PIX—Hires screen dump for the EPSON MX-80, inverse or normal, larger than full page graphics in 2 orientations. Needs Tymac PPC-100 Printer board or we will upgrade your EPSON board for $25 $39.95
GRAPH-FIT—A hires graphing program that produces bar charts, pie charts and line graphs. Has auto scaling feature too $25.00

STILL MORE APPLE GOODIES

APPLE KEYBOARD SYNTHESIZER—49 note (C to C) AGO Keyboard with 3 sawtooth sq wave shapers, 3 audio oscilators, 3 low pass filters, 4-64 point shape controllers, 2 envelope generators. Complete system $995
KEYBOARD ONLY with Apple Interface $649
GRAPHIC NOTEWRITER—Hires note write for synthesizer system $99
SUPER PIX OKI—Hires screen dump for OKI Microline 80, 82, 83 Printers. Same features as super pix. Needs Tymac PPC-100 Board $24.95
NIBBLES AWAY—The best disk back up program to date. Allows you to make backups of most every disk ever produced for the Apple. Over 10 user defined parameters including synchronized & unsynchronized copies as well as automatic half tracking and raw date transfer $59

Call 201-839-3478 for Dealer & Distributor inquiries.

Apple Bits, Part I

This article describes several aids to faster and more efficient low resolution graphics programming, including machine language routines.

Richard C. Vile, Jr.
3467 Yellowstone Dr.
Ann Arbor, Michigan 48105

This is the first in a series of articles dealing with the use of the Apple II low resolution graphics features. Some techniques will be described that use machine language to enhance the speed of graphics applications and reduce the amount of memory required in order to represent certain screen patterns.

The basic techniques to be described will enable display of patterns, each of which is 8 × 8 in size or smaller and consists of a single color. Larger patterns must be constructed from smaller pieces which fit these requirements. A modification of the machine language routine will allow multiple colors to be obtained by overlaying.

This article will describe the machine language display program which converts a numerically encoded picture into the low resolution display pattern. The next article will describe an Integer BASIC program which allows the user to interactively develop a series of patterns, store their corresponding numerical representations in memory, and save it all on disk or tape. Finally, each article in the series will present one or more applications of the techniques to the construction of animations in Lo-Res.

Bit-encoding a Picture

Consider the following eight hexadecimal numbers:

38,38,12,FE,90,28,44,83

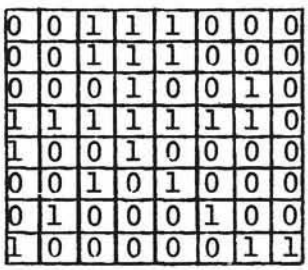
Believe it or not, they contain a picture! To see how, let's first rewrite the numbers in binary, using the following table to convert each hex digit into a 4-bit binary "nibble:"

Hex	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

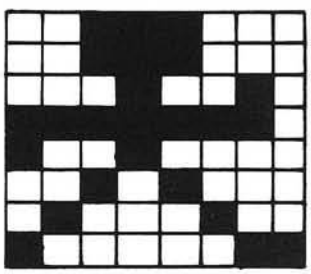
We arrive at the following numbers:

```
0 0 1 1 1 0 0 0
0 0 1 1 1 0 0 0
0 0 0 1 0 0 1 0
1 1 1 1 1 1 1 0
1 0 0 1 0 0 0 0
0 0 1 0 1 0 0 0
0 1 0 0 0 1 0 0
1 0 0 0 0 0 1 1
```

Do you see the picture yet? Just in case you don't, let's transform the pattern of 0's and 1's onto "graph paper" by superimposing a grid of squares on top of the above list, like so:



Now, erase all the 0's and completely blacken the squares containing the 1's. That gives the grid shown next:



Now, of course, you see the "picture." Erasing the grid lines should make the correspondence with the Lo-Res display pretty obvious as well. The question now becomes: "How do we turn the above process into a program?"

Shown in listing 1 is a machine language program which will carry out the process. It "assumes" that certain information has been set up for it. This information will be illustrated by listing 2 (in Integer BASIC). We discuss this further following that program.

Listing 1

```
*800ALL

080A-   A5 30       LDA   $30
080C-   8D 04 08    STA   $0804
080F-   AC 00 08    LDY   $0800
0812-   8C 03 08    STY   $0803
0815-   CE 03 08    DEC   $0803
0818-   30 31       BMI   $084B
081A-   AE 01 08    LDX   $0801
081D-   8E 02 08    STX   $0802
0820-   CA          DEX
0821-   30 F2       BMI   $0815
0823-   BD 50 08    LDA   $0850,X
0826-   AC 03 08    LDY   $0803
0829-   31 3C       AND   ($3C),Y
082B-   D0 04       BNE   $0831
082D-   A9 00       LDA   #$00
082F-   85 30       STA   $30
0831-   A5 24       LDA   $24
0833-   18          CLC
0834-   6D 03 08    ADC   $0803
0837-   A8          TAY
0838-   A5 25       LDA   $25
083A-   8E 02 08    STX   $0802
083D-   6D 02 08    ADC   $0802
0840-   20 00 F8    JSR   $F800
0843-   AD 04 08    LDA   $0804
0846-   85 30       STA   $30
0848-   4C 20 08    JMP   $0820
084B-   60          RTS
084C-   80          ???
084D-   10 10       BPL   $085F
084F-   F8          SED
0850-   01 02       ORA   ($02,X)
0852-   04          ???
0853-   08          PHP
0854-   10 20       BPL   $0876
0856-   40          RTI
0857-   80          ???
0858-   A8          TAY
0859-   B0 08       BCS   $0863
085B-   28          PLP
*
```

Machine Language Pattern Displayer

Listing 2

```
10 GR : PRINT : PRINT : PRINT
12 POKE 2048,7: POKE 2049,7
15 ROW=7+ RND (27)
20 COL=7+ RND (27)
25 COLOR= RND (15)+1
28 POKE 36,COL: POKE 37,ROW
30 FOR J=1 TO RND (10)
40 SPARK=1+ RND (20)
50 OFFSET=SPARK*7
60 POKE 60,(3072+OFFSET) MOD 256
65 POKE 61,(3072+OFFSET)/256
70 CALL 2058
72 FOR DE=1 TO RND (25): NEXT DE
75 NEXT J
80 COLOR=0: FOR J=0 TO 6: HLIN COL,
   COL+6 AT ROW+J: NEXT J
85 GOTO 15
```

Integer BASIC Fireworks Animation

The BASIC program does a series of POKEs which set up the machine language routine's information:

```
12 POKE 2048,7: POKE 2049,7
```

indicates the width and height of the patterns to be displayed.

```
28 POKE 36,COL: POKE 37,ROW
```

indicates the ROW and COLUMN of the Lo-Res screen at which the upper-left corner of the pattern to be displayed will be.

```
60 POKE 60,(3072 + OFFSET)
MOD 256

65 POKE 61,(3072 + OFFSET)
/256
```

stores the address in Apple II RAM at which the numerical codes for the pattern to be displayed begin.

The machine language program is invoked by the line:

```
70 CALL 2058
```

Running the Fireworks Animation

The numerical data which the program uses must first be entered into memory. This data resides at locations C00 to D27 (3072-3367) and has been listed on the next page. Once you have entered it (sorry about that) using the monitor, save it on tape (C00.D27W) or on disk:

```
*3DOG
>BSAVE SPARKS,A$C00,L$127
```

to avoid keying it in again later. Likewise, enter the machine language program using the monitor or the mini-assembler and save it:

```
*800.857W                    (Tape)
```

or

```
*3DOG
>BSAVE APPLE-BITS,
A$800,L$57                   (Disk)
```

(Continued)

In order to run the program, you should issue the command:

```
>LOMEM:4096
```

so that BASIC doesn't clobber the machine language program.

Assuming you are using a disk-based system, the entire sequence of commands needed to run the animation would be:

```
>BLOAD APPLE-BITS
>BLOAD SPARKS
>LOMEM:4096
>RUN FIREWORKS
```

(If you hate keying in long command sequences, cook up an EXEC file with the commands in it.)

Numerical Data for Fireworks Animation

```
*0C00.0D27

0C00- FF FF FF 15 1F 15 F5 00
0C08- 00 00 08 00 00 00 00 00
0C10- 14 00 14 00 00 00 22 00
0C18- 00 00 22 00 41 00 00 00
0C20- 00 00 41 00 00 14 08 14
0C28- 00 00 00 22 14 00 14 22
0C30- 00 41 22 00 00 00 22 41
0C38- 00 22 14 08 14 22 00 41
0C40- 22 14 00 14 22 41 41 22
0C48- 14 08 14 22 41 00 00 00
0C50- 08 00 00 00 00 00 08 14
0C58- 08 00 00 00 08 00 22 00
0C60- 08 00 08 00 00 41 00 00
0C68- 08 00 00 08 1C 08 00 00
0C70- 00 08 08 36 08 08 00 08
0C78- 08 00 63 00 08 08 00 08
0C80- 08 3E 08 08 00 08 08 08
0C88- 77 08 08 08 08 08 08 7F
0C90- 08 08 08 12 1F 10 19 15
0C98- 12 11 15 0A 06 1F 04 17
0CA0- 15 09 1F 15 1D 19 05 03
0CA8- 0A 15 0A 17 15 1F 00 0A
0CB0- 00 10 1A 00 FF FF FF 0A
0CB8- 0A 0A FF FF FF 01 15 07
0CC0- FF FF FF 1F 05 1F 1F 15
0CC8- 0A 1F 11 11 1F 11 0E 1F
0CD0- 15 11 1F 05 01 1F 11 19
0CD8- 1F 04 1F 11 1F 11 18 11
0CE0- 1F 1F 06 19 1F 10 10 1F
0CE8- 02 1F 1F 0E 1F 1F 11 1F
0CF0- 1F 05 07 1F 11 17 1F 05
0CF8- 1A 17 15 1D 01 1F 01 1F
0D00- 10 1F 0F 10 0F 1F 08 1F
0D08- 1B 04 1B 03 1C 03 19 15
0D10- 13 FF FF FF FF FF FF 00
0D18- 11 1F FF FF FF FF FF FF
0D20- FF FF FF FF FF FF FF FF
```

MICRO

Just Published!

What's Where in the Apple?

An Atlas to the Apple Computer

By William F. Luebbert
Adjunct Professor of Engineering, Dartmouth College

The **MOST DETAILED** description to date of Apple II Firmware and Hardware.

- *This Atlas and Gazetteer of* **PEEKs, POKEs,** *and* **CALLs** *lists in tabular form over 2000 memory locations.*
- *Information is presented numerically in the Atlas, and alphabetically in the Gazetteer.*
- *The names and locations of various* **Monitor, DOS, Integer BASIC,** *and* **Applesoft** *routines are listed, and information is provided on their use.*
- *The easy-to-use format includes:*
  - *The address in hexadecimal (useful for assembly programming):* $FC58
  - *The address in signed decimal (useful for BASIC programming):* . . . . . (-936)
  - *The common name of the address or routine:* . . . . . [HOME]
  - *Information on the use and type of routine:* . . . . . \SE\
  - *A description of the routine:* . . . . . CLEAR SCROLL WINDOW TO BLANKS. SET CURSOR TO TOP LEFT CORNER
- *Related register information:* . . . . . {A- Y-REGS ALTERED}

This reference tool offers information every serious Apple user needs. BASIC and assembly language users alike will find the book helpful in understanding the Apple.

128 pages, 8½ × 11 inches, cardstock cover, Wire-O binding. **$14.95***

Ask your computer store for *What's Where in the Apple* or call

800-227-1617 Ext. 564

In California call 800-772-3545 Ext. 564

24-Hour Toll-free Service **VISA and Mastercard Accepted**

34 Chelmsford Street, P.O. Box 6502, Chelmsford, MA 01824

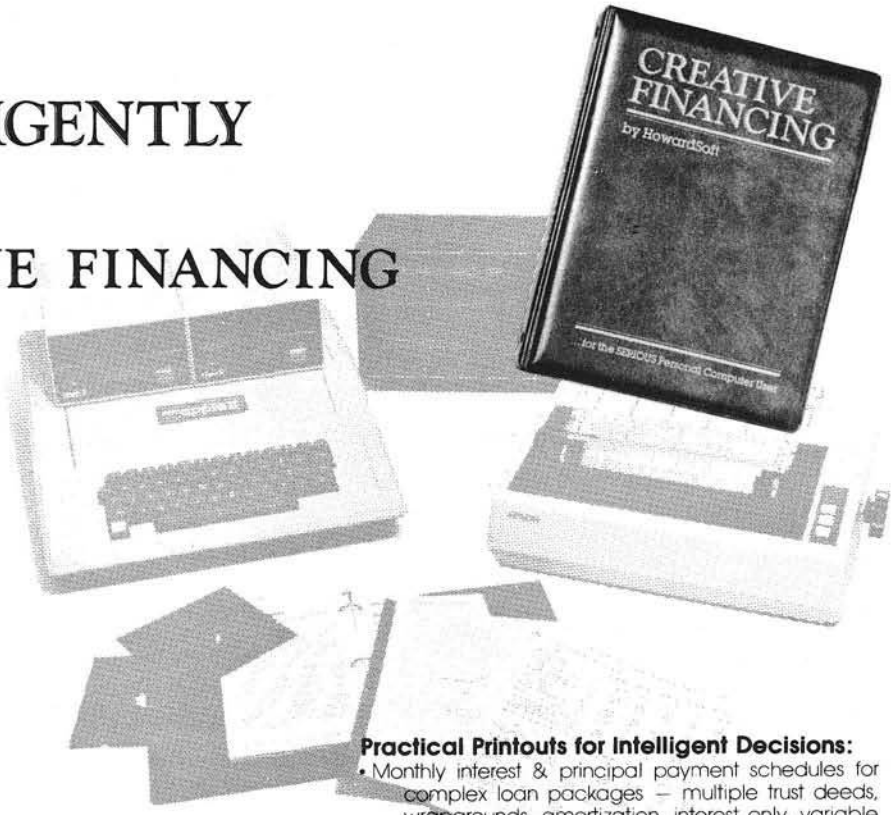
*Add $2.00 for surface shipping. Massachusetts residents add 5% sales tax.

INVEST INTELLIGENTLY

with

CREATIVE FINANCING

by HowardSoft™



Practical Printouts for Intelligent Decisions:

- Monthly interest & principal payment schedules for complex loan packages – multiple trust deeds, wraparounds, amortization, interest only, variable interest rates, delayed payments, balloons, rebates, mid-year starts, add-on interest, simple interest.
- Annual depreciation schedules for complex component depreciation – straight line, sum of year's digit, declining balance, automatic conversion to straight line, tax credits, recapture of excess depreciation.
- Tabular projections of year-end on-sale profits – internal rate of return, return on investment, return on prior year's equity, net present value, future value.
- Tabular projections of cash flows – debt service, tax credits, leases, tax payments, expenses

Get an Edge with CREATIVE FINANCING:

This sophisticated but friendly software clears the fog created by today's creative loan packages, complex tax laws, runaway inflation, and cash-draining investments. Whether you invest in bonds, stocks, income property, or trust deeds, or you just want to determine the true cost of your home mortgage or consumer loan, this package will aid you in intelligent decision-making. Realistically see the direct impact of inflation, time value of money, taxes, and cash flow timing, and play "what if" with the confidence that all important factors are properly handled. It is now possible in just a few minutes to objectively compare dissimilar loans and investments with the realism formerly achieved only by the few financial wizards who could complete analyses of this type in a few hours. This package is INDISPENSIBLE FOR ANYONE WHO INVESTS WITH BORROWED MONEY.

Analyses for Today's Complex Transactions:

- Picks up where our Real Estate Analyzer left off.
- Equivalent compound annual interest rate of loan package.
- Yield to maturity for discounted bonds.
- Present value of complex annuity schedules.
- Value of ownership vs. sale-leaseback.
- Profit comparisons for dissimilar investments.
- Risk evaluation through "what if" comparisons.

NOW AT YOUR DEALER FOR $150.

Friendly Software for Efficient Operation:

- Extensive documentation, complete with tutorials and sample analyses for the financial novice.
- Error-tolerant and trouble-free for use by the computer novice.
- Flexible, professional, paginated report generation for impressive reports on all printers.
- For all Apple Computers with 48K, Applesoft, and one 16-sector disk drive.

Also ask about our Tax Preparer and Real Estate Analyzer software.

Howard Software Services
...for the SERIOUS Personal Computer User
6713 Vista del Mar | La Jolla, CA 92037 | (714) 454-5079

Tracemark, An Apple II Debugging Aid

TRACEMARK is a debugging or study-tool utility, an extension of the Apple II Monitor TRACE command. A count of each traced instruction is kept in a separate memory block, and the trace display rate is controlled by a game paddle. Certain complex, highly convoluted problems can be better understood using this tool.

Raymond Weisling
Jalan Citropuran No. 23
Surakarta, Jawa Tengah,
Indonesia

During assembly language program development several debugging techniques are commonly employed, including single-step, trace and breakpoint. The Apple II Monitor provides the facilities for these techniques, although on a rudimentary level. Use of the TRACE function causes the program steps to scroll past at a dizzying speed, and the only way to stop the function is to hit the RESET key, which alters the stack pointer. Single-stepping allows more careful study of the program under test, but the manual requirements of typing the command 'S' for each step quickly become tedious.

TRACEMARK was developed to fill this spectrum of capability, and it also introduces an even more powerful tool for software detective work: 'footprinting' (herein called a 'mark'). How often have you wanted to know where a faulty program has gone to reach some (usually) faulty result? With single-stepping you can see the program flow, but making a record requires tedious hand-copying of the addresses. TRACEMARK not only makes a mark of the whereabouts of the processor for later evaluation, but keeps count of the number of times each instruction is executed (255 maximum). The only thing we sacrifice is knowing the order of instructions and why a certain branch occurred.

Specifically, TRACEMARK calls the Monitor single-step routine (not found in the Auto-start ROM — but if you are doing assembly language work you probably have the standard Monitor, or should). Each instruction is displayed as if in the free-running Trace mode, but the speed is controlled by the game paddle 0, from full speed (almost) to a complete stop. For the address of each instruction executed, an offset address is generated, which points to a free memory area (mark buffer) where the count of each instruction resides. This count is incremented up to a maximum of 255 ($FF). Multiple byte instructions have each byte marked identically, so any wild branches into the operand portion of an instruction will show differing mark counts.

The mark buffer resides "in parallel" with the program under test. That is, the offset of the address is only done to the high byte, greatly simplifying the mental work required when inspecting the mark buffer later. The program does not check to see if the mark buffer address is within the bounds of the actual mark buffer, so care must be exercised in selecting an offset value based on the expected range of the program under test. If extremely wide ranges are expected, it might be best to insert some boundary checking into TRACEMARK.

Program Operation

TRACEMARK, as shown in the assembly listing, resides at that popular spot, $300. It may be re-assembled to any convenient location. Similarly, page zero use may be adjusted to avoid conflicts (TPCL, TPCH, TA, TB, & TC). From the Monitor, preset values for the trace beginning point (TPCL, TPCH), the mark buffer low boundary (TA, TB) and the mark buffer high boundary high byte (TC). Then call $300. First the mark buffer is filled with zeros by CLEAR. The Monitor program counter is then loaded with the trace start ad-

Listing 1

```
LINE# LOC  CODE   LINE

0002 0000         ; TRACEMARK
0003 0000         ; REV B4 - 27 DEC 1980
0004 0000         ; R WEISLING; SURAKARTA, INDONESIA
0005 0000               *=$300
0006 0300         TPCL    = $00           ;TRACEMARK PGOGRAM COUNTER --
0007 0300         TPCH    = $01           ;-- ALSO TRACE START ADDRESS
0008 0300         TA      = $02           ;TRACEMARK BUFFER START (LO)
0009 0300         TB      = $03           ;                       (HI)
0010 0300         TC      = $04           ;BUFFER END (HI)
0011 0300         LENGTH  = $2F
0012 0300         PCL     = $3A           ;MONITOR PC
0013 0300         PCH     = $3B
0014 0300         STEP    = $FA43         ;TRACE SINGLE STEP ROUTINE
0015 0300         PREAD   = $FB1E         ;MONITOR - READ PADDLE VALUE
0016 0300         ;
0017 0300         ; TRACEMARK AND TRACE DISPLAY RATE UTILITY
0018 0300         ;
0019 0300 204E03 START  JSR CLEAR         ;ENTER HERE FROM MONITOR
0020 0303 A500          LDA TPCL        ;UPDATE MONITOR PROGRAM COUNTER
0021 0305 853A          STA PCL
0022 0307 A501          LDA TPCH
0023 0309 853B          STA PCH
0024 030B 2043FA AGAIN  JSR STEP        ;DO ONE STEP IN MONITOR
0025 030E A501          LDA TPCH        ;GET ADDRESS (PCL/H 1 STEP AHEAD)
0026 0310 38            SEC
0027 0311 E9D0          SBC #$D0        ;<-CHANGE THIS FOR OTHER OFFSETS
0028 0313 8501          STA TPCH        ;OFFSET FOR MARKING
0029 0315 A42F          LDY LENGTH      ;SIZE OF INSTRUCTION, 1-3 BYTES (0-2)
0030 0317 B100          LDA (TPCL),Y    ;GET FORMER MARK VALUE
```

(Continued)

dress, and the single-step display loop begins. After the call to STEP, the mark buffer address is calculated from TPCL, TPCH values (the Monitor PC is already set for the next instruction).

LENGTH is a value representing the size of the last instruction disassembled (less one), and acts as an index to LOOP for marking each byte of multi-byte instructions. The previous mark count is first read and checked to prevent overflow, and then the incremented count is returned to the mark buffer. The paddle is read by a call to PREAD in the Monitor, and the returned value is used to set the repeat count of two nested timing loops. In addition, the value is tested to see if it is in the range of $FC to $FF, and if so, the paddle is read again (and again) until the value drops below $FC, effectively freezing the display. The paddle read-rate is slowed down by doing this value testing, after the inner loop, to eliminate errors caused from calling the PREAD routine too rapidly. (This caveat, mentioned in the Apple literature, is due to circuit peculiarities of this crude a/d converter.

Having the speed control is handy to use: slow down or stop the display for close inspection or let it run at full speed for those loops that work well. TRACEMARK will continue as long as possible — the only means of exit are encountering a BRK or hitting 'RESET.' Before calling the program again, be sure to reset the five bytes in page zero to the correct values for initializing the mark buffer and starting the trace. If more page zero space can be spent, these initial values could be made non-destructive (but the program will be slightly longer).

Listing 1 *(Continued)*

```
0031 0319 AA            TAX
0032 031A E8            INX             ;BUMP BY ONE
0033 031B F006          BEQ DONE        ;PREVENT MARK OVERFLOW BEYOND $FF
0034 031D 8A            TXA
0035 031E 9100   LOOP   STA (TPCL),Y    ;SAVE BUMPED MARK
0036 0320 88            DEY
0037 0321 10FB          BPL LOOP        ;REPEAT IF 2-3 BYTE INSTRUCTION
0038 0323 A53A   DONE   LDA PCL         ;READY LOCAL PC FOR NEXT STEP
0039 0325 8500          STA TPCL
0040 0327 A53B          LDA PCH
0041 0329 8501          STA TPCH        ;TRACEMARK PHASE DONE
0042 032B A200   RD     LDX #0          ;DISPLAY RATE PHASE BEGIN
0043 032D 201EFB        JSR PREAD       ;READ PADDLE 0 VALUE
0044 0330 8404          STY TC          ;SAVE FOR DELAY RESTORE USE
0045 0332 C8            INY             ;ADJUST FOR DEC BEFORE TEST
0046 0333 8502          STA TA
0047 0335 8503          STA TB
0048 0337 C602   PLOP   DEC TA          ;INNER DELAY LOOP
0049 0339 D0FC          BNE PLOP
0050 033B A504          LDA TC          ;GET PADDLE VALUE AGAIN
0051 033D 8502          STA TA          ;RESTORE INNER LOOP
0052 033F A903          LDA #03         ;MAKE LOW 2 BITS HI
0053 0341 0504          ORA TC          ;OR WITH PADDLE VALUE (111111XX)
0054 0343 A8            TAY             ;-- IF RESULT $FF AND --
0055 0344 C8            INY             ;-- HERE MADE ZERO THEN --
0056 0345 F0E4          BEQ RD          ;GO BACK TO READ PADDLE (DISPLAY FREEZE)
LINE# LOC CODE   LINE

0057 0347 C603          DEC TB          ;OUTER DELAY LOOP
0058 0349 D0EC          BNE PLOP        ;DELAY TIME = PADDLE VALUE SQUARED
0059 034B 4C0B03        JMP AGAIN       ;DISPLAY RATE PHASE DONE - REPEAT
0060 034E          ;
0061 034E          ; CLEAR MARK BUFFER FROM $TBTA TO $TC00
0062 034E          ;   (THESE MUST BE PRELOADED FROM MONITOR
0063 034E          ;    EACH TIME PROGRAM IS CALLED - BEWARE)
0064 034E          ;
0065 034E A400   CLEAR  LDY $00         ;CLEAR MARK BUFFER, TA/TB = FIRST
0066 0350 98            TYA             ;                    ADDRESS OF BUFFER
0067 0351 9102   CLOP   STA (TA),Y      ;ZERO MEMORY BYTE
0068 0353 E602          INC TA          ;BUMP ADDRESS LO
0069 0355 D0FA          BNE CLOP
0070 0357 E603          INC TB          ;CARRY - BUMP ADDRESS HI
0071 0359 A504          LDA TC          ;GET LAST ADDRESS LIMIT
0072 035B C503          CMP TB          ;CHECK LIMIT AGAINST HI PART
0073 035D D0EF          BNE CLEAR       ;MORE TO CLEAR
0074 035F 60            RTS             ;DONE
0075 0360               .END
```

Example of Use

Let us see how TRACEMARK operates by running it to trace the Apple Integer BASIC program. This is a good example of a program which is highly convoluted and which cannot be debugged by conventional breakpoints, since it resides in ROM. The listing shows line 27 as being SBC #$D0, which is the offset for this example. Thus BASIC, running from $E000 to $FFFF (including display calls to within the Monitor) will be marked from $1000 to $2FFF (mark buffer). Now, preset the page zero registers as follows:

00:00 E0 00 10 30 'return'

This means to start the trace at $E000 (cold start), clearing the mark buffer from $1000 to $2FFF. Next, call TRACEMARK and watch the trace display, using the paddle for rate control. At some point hit 'reset' and then inspect the mark buffer. It will contain many zeros of course — those are places never reached by the program. But look at $2000-2022 (representing $F000-$F022) or around $2C62-2CA7 (the scroll routines in the Monitor) and you will see evidence that the program worked here for some time.

While this program seems pretty dependent on the Apple II firmware and hardware, it shows merit for study by those with other systems, since the concept of making a parallel counter or mark buffer is processor-independent. All that is necessary is a processor simulator trace program which can single-step and yield the address of each step and optionally the size of each instruction traced.

Limitations

Performing a TRACEMARK on a program with even a small amount of printing, via the Monitor display and scroll routines, will consume some considerable time while scrolling the text buffer. Worse still is the problem of a program which alters the two-byte output vector at $36-37 (CSWL), since one byte will be changed while the other remains unchanged, and everything will come to a grinding halt (when the trace program tries to output something to a half-baked jump vector). Additionally, some other Monitor routines may not be traceable, due to ambiguities relating to display status. The other limit is that of space, with no room for both the program under test and the mark buffer. Here some segmentation of the area to be marked, with mark buffer boundary checking, is called for.

TRACEMARK should enhance the set of debugging tools commonly employed for development of programs at the assembler level, and is also useful where coupled with a disassembler, for study of undocumented programs in native machine code. I hope that use of TRACEMARK will cut time from an oft-times painful and tedious chore as it has done for me.

Raymond Weisling has two degrees in music composition from California Institute of the Arts, where he also worked on software development in the Hybrid Computer Music Studio. He was a software and hardware designer for an industrial electronics manufacturer before resuming his artistic career. He uses the Apple II for design support in making acoustic and kinetic sculptures.

MICRO

PRICE BREAKTHROUGH

16K RAM BOARDS FOR APPLE JUST $129.95

HAVE YOU BEEN WAITING FOR THE COST OF EXPANSION BOARDS TO COME DOWN? YOUR WAIT IS OVER. UP UNTIL NOW **RAM** EXPANSION HAS COST AS MUCH AS $195.00. NOW **OMEGA MICROWARE** IS PROUD TO ANNOUNCE THE ARRIVAL OF A TRULY AFFORDABLE EXPANSION CARD.

NOW YOU CAN RUN PASCAL, FORTRAN, 56K CPM WITH A Z80 SOFTCARD, INTEGER BASIC, APPLESOFT AND OTHER LANGUAGES ON YOUR **APPLE**. NOW YOU CAN INCREASE USUABLE MEMORY FOR **VISICALC**. NOW YOU DON'T HAVE TO PAY A FORTUNE TO HAVE ALL THIS.

AT $129.95, **OMEGAS RAMEX 16** IS THE LOWEST PRICED CARD AVAILABLE TODAY.

WHAT DO YOU GIVE UP WHEN YOU PURCHASE THIS FIRST REALLY AFFORDABLE **RAM** EXPANSION CARD? WELL, YOU GIVE UP HAVING TO REMOVE ONE RAM CHIP FROM THE MOTHER BOARD OF YOUR **APPLE**. YOU GIVE UP HAVING TO STRAP A CABLE FROM THE CARD TO YOUR MOTHER BOARD. THAT'S IT. WHAT YOU GET IS A SIMPLE, RELIABLE, BOARD THAT JUST PLUGS IN. MEMORY REFRESH IS ACCOMPLISHED ON THE BOARD ITSELF.

THE **RAMEX 16** IS GUARANTEED NOT JUST FOR 90 DAYS. NOT EVEN 6 MONTHS. OUR WARRENTY IS FOR ONE FULL YEAR FROM DATE OF PURCHASE. WE WILL REPAIR OR REPLACE ANY BOARD THAT IS DEFECTIVE THROUGH MANUFACTURE FOR A PERIOD OF ONE YEAR AFTER PURCHASE PROVIDED THIS DAMAGE IS NOT USER INFLICTED.

ORDER YOUR **RAMEX 16** NOW BY CALLING **TOLL FREE** 1-800-835-2246. **KANSAS RESIDENTS CALL** 1-800-362-2421. MASTERCARD OR VISA ACCEPTED OR SEND $129.95. ILLINOIS RESIDENTS ADD $7.80 SALES TAX.

ANOTHER QUALITY PRODUCT FROM
OMEGA MICROWARE, INC.
FORMERLY OMEGA SOFTWARE PRODUCTS, INC.
222 SO. RIVERSIDE PLAZA
CHICAGO, IL 60606
PHONE 312-648-1944

©**OMEGA MICROWARE, INC.**

APPLE AND APPLESOFT ARE REGISTERED TRADEMARKS OF APPLE COMPUTER, INC. PASCAL IS A REGISTERED TRADEMARK OF THE REGENTS OF THE UNIV. OF CA. SAN DIEGO. VISICALC IS A REGISTERED TRADEMARK OF PERSONAL SOFTWARE. CPM IS A REGISTERED TRADEMARK OF DIGITAL RESEARCH INC. Z80 IS A REGISTERED TRADMARK OF ZILOG, INC. SOFTCARD IS A REGISTERED TRADMARK OF MICROSOFT.

PROGRAM LINE EDITOR

BY SYNERGISTIC SOFTWARE

HAVE YOU EVER WORKED ON A PROGRAM LATE INTO THE NIGHT?

Now with the **Program Line Editor (PLE),** program development and modification take only a fraction of the time previously required. PLE is the most sophisticated program development tool ever made for the Apple II* computer, letting you devote your energies to design, analysis, and testing instead of the drudgery of repetitive keying of program lines requiring slight changes.

FEATURES INCLUDE:

SOPHISTICATED PROGRAM EDITING FEATURES – More than eleven editing commands, such as INSERT, DELETE, PACK, and FIND allow you to change a program line with a minimum of keystrokes.

VERSATILE ESCAPE FUNCTIONS – User definable command sequences can be assigned to any key, then invoked whenever desired by pressing ESC and that key. Twenty-seven predefined functions come with PLE (ROM version functions cannot be modified).

LISTING CONTROL – This feature allows you to control program listings, CATALOGs, etc., with the touch of a key – stopping and starting them at will, or terminating them entirely.

LOWER CASE – Lower case characters can now be entered directly from the keyboard without any cumbersome software patches.

PROGRAMMABLE CURSOR CONTROL – Fancy screen displays are made possible with the Programmable cursor feature. Four characters control cursor movement in all four directions from within PRINT statements.

INVISIBLE AND INDESTRUCTIBLE – When booted, PLE actually makes itself a part of the Disk Operating System, and thus is unaffected by INT, FP, and MAXFILES commands.

NOW AVAILABLE ON DISK OR ROM!
ROM version will operate on Mountain Computer, Inc's ROMPLUS board or the new Andromeda, Inc. ROM board
Disk & ROM versions can operate on any configuration Apple II or Apple II PLUS with DOS 3.2 or DOS 3.3

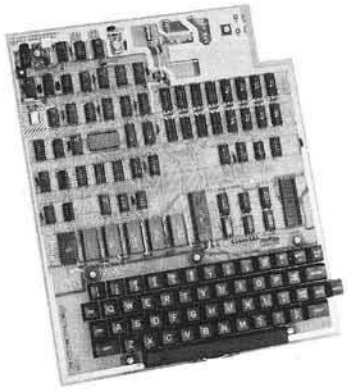
Apple II is a trademark of Apple Computer, Inc.

ORDER YOURS TODAY!
DISK, $40.00 / ROM $60.00

Available at your local dealer or send check or money order to SYNERGISTIC SOFTWARE, 5221 120th Ave. S.E., Bellevue, Washington 98006.
or call **206-226-3216.** Wash. residents add 5.4% sales tax.

DOWN-TO-EARTH PRICES ON OUT-OF-THIS-WORLD PERSONAL COMPUTERS AND COMPONENTS.

Look at this!



Ohio Scientific Superboard II
$329

- It's the first complete computer system on a board.
- Superboard II uses the ultra powerful 6502 Microprocessor
- 8K Microsoft BASIC-in-ROM
- 4K static RAM on board, expandable to 8K
- Full 53-key keyboard, with upper and lower case. Plus user expandability.
- Video interface and audio cassette interface.

The Ohio Scientific Superboard II at $329 — in today's economy — has got to be the best buy by far. It will entertain you with spectacular graphics made possible by its ultra high resolution graphics and super fast BASIC. It will help you in school or industry, as an ultra powerful scientific calculator. Advanced scientific functions and a built-in "immediate" mode allow you to solve complex problems without programming.

The Superboard II can be expanded economically, for business uses, or to remotely control your home appliances and security. Even communicate with other computers.

Read what's been written about Superboard II:

"We heartily recommend Superboard II for the beginner who wants to get into microcomputers with a minimum cost. A real computer with full expandability."

—POPULAR ELECTRONICS, MARCH 1979

"The Superboard II is an excellent choice for the personal computer enthusiast on a budget."

—BYTE, MAY 1979

Look at these easy hardware prices:

610 Board For use with Superboard II and Challenger 1P. 8K static RAM. Expandable to 24K or 32K system total. Accepts up to two mini-floppy disk drives. Requires +5V @4.5 amps.	**$ 298**
Mini-Floppy Disk Drive Includes Ohio Scientific's PICO DOS software and connector cable. Compatible with 610 expander board. Requires +12V @1.5 amps and +5V @ 0.7 amps. [Power supply & cabinet not included.]	**299**
630 Board Contact us for important details.	**229**
AC-3P 12" combination black and white TV/video monitor.	**159**
4KP 4K RAM chip set.	**79**
PS-005 5V 4.5 amp power supply for Superboard II.	**45**
PS-003 12V power supply for mini-floppies.	**45**
RF Modulator Battery powered UHF Unit.	**35**
AC-12P Wireless remote control system. Includes control console, two lamp modules and two appliance modules, for use with 630 board.	**175**
AC-17P Home security system. Includes console, fire detector, window protection devices and door unit for use with 630 board.	**249**
C1P Sams C1P Service manual	**8**
C4P Sams C4P Service manual	**16**
C3 Sams Challenger III manual	**40**

Ohio Scientific and independent suppliers offer hundreds of programs for the Superboard II, in cassette and mini-floppy form.

Freight Policies All orders of $100 or more are shipped freight prepaid. Orders of less than $100 please add $4.00 to cover shipping costs. Ohio residents add 5.5% Sales Tax.

VISA master charge

Hours: Call Monday thru Friday. 8:00 AM to 5:00 PM E.D.T.
TOLL FREE: 1-800-321-5805

Guaranteed Shipment
Cleveland Consumer Computers & Components guarantees shipment of computer systems within 48 hours upon receipt of your order. **Our failure to ship within 48 hours entitles you to $35 of software, FREE.**

To Order: Or to get our free catalog **CALL 1-800-321-5805 TOLL FREE.** Charge your order to your **VISA** or **MASTER CHARGE** account. Ohio residents call: (216) 464-8047. Or write, including your check or money order, to the address listed below.

CLEVELAND CONSUMER COMPUTERS & COMPONENTS
P.O. Box 46627
Cleveland, Ohio 44146

Order Form:

CLEVELAND CONSUMER COMPUTERS & COMPONENTS
P.O. Box 46627
Cleveland, Ohio 44146

☐ Superboard II $329.
☐ 610 Board $298.
☐ Mini-Floppy Disk Drive $299.
☐ RF Modulator $35.
☐ AC-3P 12" B/W Monitor $159.
☐ C1P Sams Manual $8.

[Attach separate sheet for other items.]

NAME _____

ADDRESS: _____

CITY: _____ STATE: _____ ZIP: _____

PHONE: _____

Payment by: VISA _____ MASTER CHARGE _____ MONEY ORDER _____

Credit Card Account #_____

Expires _____ Interbank #[Master Charge] _____

TOTAL CHARGED OR ENCLOSED $_____ [Ohio Residents add 6.5% Sales Tax]

Orders of less than $100, please add $4.00 to cover shipping costs. Orders will be accepted from U.S. and Canada only. All prices quoted are U.S., date of publication, standard UPS shipping FOB the factory.

PRICES SUBJECT TO CHANGE WITHOUT NOTICE

MICROCRUNCH: An Ultra-fast Arithmetic Computing System

Part 2

This article describes software support for the fast mathematics hardware outlined in Part I (39:07). A detailed discussion of machine code routines necessary for communication between the arithmetic processing chip and BASIC is given, along with an overview of a BASIC home-brew compiler.

John E. Hart
Department of Astrogeophysics
University of Colorado
Boulder, Colorado 80309

Part I described a hardware floating point board and demonstrated that truly high speed computing is only possible with a microcomputer if the floating point chip is used in conjunction with a compiler. This is true where the overall program is written in direct machine code. In this case, the source code higher level language statements do not need to be interpreted or pseudo-interpreted (as in Pascal).

This article describes a compiler that is useful for fast arithmetic processing, but does not translate statement types that are rarely if ever used in mathematical problems. The fundamental idea is to use the normal Microsoft BASIC interpreter to do most of the non-mathematical work and to form the overall program structure. When a mathematical loop containing floating point operations needs to be done, a jump is made to a machine language subroutine via USR that executes the equations. It is only this machine language subroutine that is generated by the compiler.

Our system compiles as machine code subroutines, all the time consuming mathematical operations. The source code for these subroutines includes a limited subset of BASIC statements. Then the full BASIC language is used to input variables, set initial conditions, print results of calculations, and perform calculations that, because they are not iterated often, are not time consuming.

There are several problems that need to be discussed.

1. How to communicate between variables used by the mathematical subroutines and variables used in the BASIC main program. Since the C8231 floating point chip uses a non-standard floating point format (at least it is different from that used by Microsoft) it is necessary for floating point subroutines to have their own variable space. The alternative of converting all BASIC variables to APU (arithmetic processing unit) format upon entry to a subroutine, and then reconverting on exit, is extremely time consuming and wasteful since only a few of the variables used are actually input or output variables. In addition, any time a change is made in BASIC the variable table shifts its position, and BASIC array storage is cumbersome and inefficient.

2. How to get in and out of a large number of compiler-generated machine code mathematical subroutines. Clearly you would like the option of writing several different subroutines and calling them from different points in the main program. Thus some kind of directory management is necessary.

3. What BASIC statements and variable allocations do we allow in the source code for mathematical subroutines?

The Limited BASIC Source Statements

Variable allocation:

Somewhat like a Tiny BASIC, all mathematical subroutine variables are described by a single alphabetic name A-Z. Unlike Tiny BASIC, any variable (except I, J, K, L, M, N that are integer variables for use in FOR loops and indexing) can be either a single number, a vector (e.g. A(I)) or a two dimensional array (e.g. U(I,J)). The vector dimension and the second array dimension must be less than 65 and the first array dimension can be anything consistent with the memory map. Thus there are two types of arithmetic that can be done in a machine language mathematical subroutine: integer and floating point. The integer arithmetic, used mostly for array indexing (e.g. U(I-2+K, J+31)) is done by the 6502 and can only be subtraction or addition.

Statement List:

SUB#, where #=1 to 9 indicating one of 9 possible subroutines.

RETURN, return from subroutine.

GOSUB#, where #=1 to 9. GO to SUB# given.

GOTO#, where #=1 to 9. GO to LABEL# within current subroutine.

FOR I=1TOJ / NEXTI Same as BASIC except no expressions allowed in index setting part of statement.

IF A=0THENGOTO# Same as BASIC, except label referred to 1 to 9, Variable reference (e.g. A) must be simple variable, not vector, etc. Also less than 0 is OK. Only comparisons w.r.t. zero can be made.

LABEL# where #=1 to 9. Jump point for GOTO and IF...THEN.

I=J+K−25, etc. General integer arithmetic involving only integer variables and numbers less than 256. Only addition and subtraction since these operations are done with the 6502. Mostly used in vector and array indices.

END Denotes termination of a particular subroutine.

In addition to these statements, general mathematical expressions can be written exactly as in BASIC. Example:

```
X = 1.234*U(I-2,J + 1) + B(J)
*SIN(3.141592*Y)
```

This is a marked improvement over such primitive compilers as FLOPTRAN IV and BASEX that do not allow chained calculations or indexing.

It can be seen that this subset of statements is sufficient to implement almost any conceivable iterative and/or conditional calculation. The advantages of the restricted variable set and limited statement types are a shorter and faster compiler. You should note that the compiler must trap all possible source code errors during the compilation, or the machine code subroutines will crash (or give back garbage) and debugging will be extremely difficult. This error trapping is the most difficult part of language translation, and it is made easier by using the restricted language outlined above.

Source statements such as those required to do a long mathematical iteration or calculation, are entered into memory under control of an editor, and then are translated into machine code and placed in the upper end of memory. The compiler and editor are written in BASIC, but being essentially word processors and language translators, execute rapidly. The memory maps for the compilation and run modes are shown in figure 1. The APU variable space depends on the precise allocation of variables, dimensions of arrays, etc. The object code is tied to an initial object starting location OI that is set before compilation.

Variable Format and Exchange

Both Microsoft BASIC and the C8231 represent floating point numbers with four bytes. The first byte contains the exponent, and the next three contain the mantissa, with the most significant bit first. Of course here we are talking about a binary representation where a number is written as

$$\left(\frac{a}{2}+\frac{b}{4}+\frac{c}{8}+\ldots.\right)\times 2^{E}$$

Figure 1: **MEMORY MAP (typical). Addresses are decimal**

Compilation	Run
0	0
Microsoft Overhead	Microsoft Overhead (loc 0-127 swapped out for math. subroutine)
700	700
BASIC Compiler	BASIC: line 0-6 Overhead line 6-700 Main Program line 730-790 Overhead Routines
16000	4000 (typ)
Compiler Variables	BASIC variables
18500 (typ)	6000 (typ., depends on variable allocation)
Source Code	APU Variables
20480 (typ)	19768 (OI-200)
Object Code	Fixed Routines and Swap Storage
32768	20480 (OI+512)
	Object Code
	32768

Figure 2: **Floating Point Formats**

			BASIC	APU
Byte 1	7		Exponent Sign	Mantissa Sign
	6		Exponent MSB	Exponent Sign
	5			Exponent MSB
	4			
	3			
	2			
	1			
	0		Exponent LSB	Exponent LSB
Byte 2	7	a	Mantissa Sign (a = 1	Mantissa a = 1 unless 0
	6	b	inferred unless 0)	
	5	c	'	
	4	d	'	
	3	e	MANTISSA (most significant bit = bit 6 byte 2)	
	2	.	'	
	1	.	'	
	0	.		
		.		
Byte 3	7	.		
	6	.		
	5	.		
	4	.		
	3	.		
	2	.		
	1	.		
	0	.		
Byte 4	7	.		
	6	.		
	5	.		
	4	.		
	3	.		
	2	u		
	1	v		
	0	w	Mantissa Least Significant Bit	

Here E is typically the exponent and a, b, c, and so forth, are the successive bits of the three byte mantissa, and are either 0 or 1. Figure 2 shows the representations for the two systems. In BASIC a 1 in bit 7 of byte 2 indicates a negative number. For the APU, a negative number is indicated by a 1 in bit 7 of byte 1! Also, bit 7 of byte 2 in APU space is always a 1, except if the number is identically zero. That is, a = 1 unless the number is zero. Note that since the mantissa sign occurs in byte 1 for the APU variable, the exponent range is less by a factor of 2 than for the BASIC variable. Indeed the BASIC exponent range is +127 to −128, e.g. the exponent is biased by bit 7, or biased negative 128. However, the APU exponent is only biased negative 64 since the mantissa sign bit occupies bit 7. Thus bit 6 gives the exponent sign.

Machine code routines have been written to convert back and forth between these two formats. Whenever you want to input a variable to APU space, or print out such a number, one of these routines is called by USR from a set of BASIC statements that precede the overall program as shown in figure 1. This is discussed in more detail below. First we list a number of machine code routines that are useful in communicating between BASIC and the APU, and between the compiled code and the APU. These routines must be entered along with each object code, but unlike the object they do not change if either the BASIC source code or main program is altered.

Fixed Routines

Listing 1 is a BASIC program that will load all the fixed routines needed for execution. This program should be run after entering the initial object address OI. OI must be a multiple of 256. In the example discussed below it is 78*256. The decimal entry points and functions of the routines entered by this program are as follows:

Listing 1

```
600  REM  FIXED ROUTINES
601  DATA 32,166,255,216,181,0,157,128,255,202,16,248,162
602  DATA 127,189,0,255,149,0,202,16,248,32,56,255,162,127,181,0,157,0
603  DATA 255,202,16,248,162,127,189,128,255,149,0,202,16,248
604  DATA 173,6,255,41,30,240,3,76,116,162,96
606  FOR J = OI TO OI + 55: READ Z: POKE J,Z: NEXT J
607  DATA 165,5,240,14,56,233,255,16,9,56,255,48,5,169,30,76,153,255
608  FOR J = OI + 170 TO OI + 187: READ Z: POKE J,Z: NEXT J
609  POKE OI + 176,WL: POKE OI + 180,OI / 256 - 1: POKE OI + 187,OI / 256
610  DATA 173,1,251,173,6,251,145,4,200
612  FOR J = 1 TO 9: READ H(J): NEXT J
614  FOR J = OI + 188 TO OI + 218 STEP 9: FOR N = 1 TO 9: POKE J + N - 1,H(N): NEXT N: NEXT J
618  POKE OI + 223,96
620  DATA 177,4,141,6,251,136
622  FOR J = OI + 228 TO OI + 230: POKE J,200: NEXT J
624  FOR J = 1 TO 6: READ H(J): NEXT J
626  FOR J = OI + 231 TO OI + 249 STEP 6: FOR N = 1 TO 6: POKE J + N - 1,H(N): NEXT N: NEXT J
627  POKE OI + 254,96
630  DATA 173,0,251,173,6,251,48,248,41,30,208,1,96,133,6
632  DATA 104,133,7,104,133,8,76,25,255
634  FOR J = OI + 140 TO OI + 163: READ Z: POKE J,Z: NEXT J
640  DATA 8,16,39,31,47
642  FOR J = 1 TO 5: READ Z: POKE OI + Z,OI / 256 + 1: NEXT J
644  DATA 24,163
646  FOR J = 1 TO 2: READ Z: POKE OI + Z,OI / 256: NEXT J
650  DATA 160,3,177,123,72,9,128,160,1,145,1,200,177,123,56,233,128,41
652  DATA 127,136,136,145,1,104,41,128,17,1,145,1,160,5,177,123,136,136
653  DATA 145,1,200,177,123,136,136,145,1,96
654  FOR J = OI + 56 TO OI + 101: READ Z: POKE J,Z: NEXT J
656  DATA 160,3,169,0,145,1,136,48,251,96
658  FOR J = OI + 102 TO OI + 111: READ Z: POKE J,Z: NEXT J
660  DATA 160,1,177,1,48,12,200,169,0,145,123,200,152,73,6,208,246,96
661  DATA 76,210,255
662  FOR J = OI + 117 TO OI + 137: READ Z: POKE J,Z: NEXT J
663  POKE OI + 137,OI / 256 - 1: POKE OI + 2,OI / 256 - 1
665  DATA 160,5,162,6,181,0,153,3,211,232,200,200,224,17,208,244,96
667  FOR J = OI - 120 TO OI - 104: READ Z: POKE J,Z: NEXT J
670  DATA 41,127,200,200,145,123,160,0,177,1,72,41,128,160,3,17,123,145
672  DATA 123,104,41,127,24,10,48,2,56,234,106,136,145,123,177,1,200
673  DATA 200,145,123,136,177,1,200,200,145,123,96
674  FOR J = OI - 46 TO OI - 1: READ Z: POKE J,Z: NEXT J
676  DATA 165,1,141,224,255,165,2,141,225,255,173,226,255,133,1,173,227
678  DATA 255,133,2,32,57,255,173,224,255,133,1,173,225,255,133,2,96
679  GOTO 684
684  FOR J = OI - 256 + 176 TO OI - 256 + 209: READ Z: POKE J,Z: NEXT J
686  DATA 4,9,12,17,22,25,30
688  FOR J = 1 TO 7: READ Z: POKE OI - 256 + 176 + Z,OI / 256: NEXT J
690  DATA 162,127,169,0,141,6,255,96
692  FOR J = OI - 90 TO OI - 83: READ Z: POKE J,Z: NEXT J
694  POKE OI - 84,OI / 256 + 1: STOP
```

(Continued)

OSI AARDVARK OSI
NOW MEANS BUSINESS!

WORD PROCESSING THE EASY WAY– WITH MAXI-PROS

This is a line-oriented word processor designed for the office that doesn't want to send every new girl out for training in how to type a letter.

It has automatic right and left margin justification and lets you vary the width and margins during printing. It has automatic pagination and automatic page numbering. It will print any text single, double or triple spaced and has text centering commands. It will make any number of multiple copies or chain files together to print an entire disk of data at one time.

MAXI-PROS has both global and line edit capability and the polled keyboard versions contain a corrected keyboard routine that make the OSI keyboard decode as a standard typewriter keyboard.

MAXI-PROS also has sophisticated file capabilities. It can access a file for names and addresses, stop for inputs, and print form letters. It has file merging capabilities so that it can store and combine paragraphs and pages in any order.

Best of all, it is in BASIC (OS65D 51/4" or 8" disk) so that it can be easily adapted to any printer or printing job and so that it can be sold for a measly price.

MAXI-PROS – $39.95

NEW-NEW–NEW TINY COMPILER

The easy way to speed in your programs. The tiny compiler lets you write and debug your program in Basic and then automatically compiles a Machine Code version that runs from 50-150 times faster. The tiny compiler generates relocatable, native, transportable machine code that can be run on any 6502 system.

It does have some limitations. It is memory hungry – 8K is the minimum sized system that can run the Compiler. It also handles only a limited subset of Basic – about 20 keywords including FOR, NEXT, IF THEN, GOSUB, GOTO, RETURN, END, STOP, USR(X), PEEK, POKE, -,=,*,/,↑,(,),<,>,Variable names A-Z, and Integer Numbers from 0-64K.

TINY COMPILER is written in Basic. It can be modified and augmented by the user. It comes with a 20 page manual.

TINY COMPILER – $19.95 on tape or disk

THE AARDVARK JOURNAL

FOR OSI USERS – This is a bi-monthly tutorial journal running only articles about OSI systems. Every issue contains programs customized for OSI, tutorials on how to use and modify the system, and reviews of OSI related products. In the last two years we have run articles like these!

1) A tutorial on Machine Code for BASIC programmers.
2) Complete listings of two word processors for BASIC IN ROM machines.
3) Moving the Directory off track 12.
4) Listings for 20 game programs for the OSI.
5) How to write high speed BASIC – and lots more –

Vol. 1 (1980) 6 back issues - $9.00
Vol. 2 (1981) 2 back issues and subscription for 4 additional issues - $9.00.

ACCOUNTS RECEIVABLE – This program will handle up to 420 open accounts. It will age accounts, print invoices (including payment reminders) and give account totals. It can add automatic interest charges and warnings on late accounts, and can automatically provide and calculate volume discounts.

24K and OS65D required, dual disks recommended. Specify system.
Accounts Receivable. $99.95

*** * * SPECIAL DEAL – NO LESS! * * ***

A complete business package for OSI small systems – (C1, C2, C4 or C8). Includes MAXI-PROS, GENERAL LEDGER, INVENTORY, PAYROLL AND ACCOUNTS RECEIVABLE – ALL THE PROGRAMS THE SMALL BUSINESS MAN NEEDS. $299.95

P.S. We're so confident of the quality of these programs that the documentation contains the programmer's home phone number!

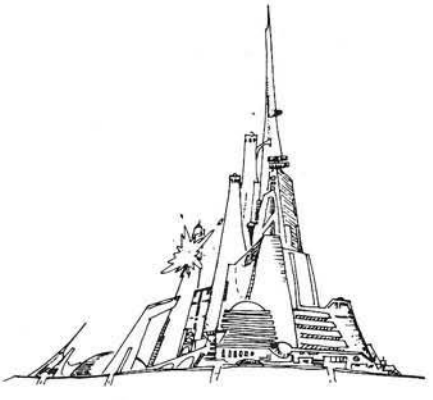
SUPERDISK II

This disk contains a new BEXEC* that boots up with a numbered directory and which allows creation, deletion and renaming of files without calling other programs. It also contains a slight modification to BASIC to allow 14 character file names.

The disk contains a disk manager that contains a disk packer, a hex/dec calculator and several other utilities.

It also has a full screen editor (in machine code on C2P/C4)) that makes corrections a snap. We'll also toss in renumbering and program search programs – and sell the whole thing for – SUPERDISK II $29.95 (5 1/4") $34.95 (8").

AND FUN, TOO!



BOOKKEEPING THE EASY WAY – WITH BUSINESS I

Our business package 1 is a set of programs designed for the small businessman who does not have and does not need a full time accountant on his payroll.

This package is built around a **GENERAL LEDGER** program which records all transactions and which provides monthly, quarterly, annual, and year-to-date PROFIT AND LOSS statements. GENERAL LEDGER also provides for cash account balancing, provides a BALANCE SHEET and has modules for DEPRECIATION and LOAN ACCOUNT computation.
GENERAL LEDGER (and MODULES) $129.95.

PAYROLL is designed to interface with the GENERAL LEDGER. It will handle annual records on 30 employees with as many as 6 deductions per employee.
PAYROLL - $49.95.

INVENTORY is also designed to interface with the general ledger. This one will provide instant information on suppliers, initial cost and current value of your inventory. It also keeps track of the order points and date of last shipment.
INVENTORY - $59.95.

GAMES FOR ALL SYSTEMS

GALAXIAN - 4K - One of the fastest and finest arcade games ever written for the OSI, this one features rows of hard-hitting evasive dogfighting aliens thirsty for your blood. For those who loved (and tired of) Alien Invaders. Specify system – A bargain at $9.95

NEW – NEW – NEW

LABYRINTH - 8K - This has a display background similar to MINOS as the action takes place in a realistic maze seen from ground level. This is, however, a real time monster hunt as you track down and shoot mobile monsters on foot. Checking out and testing this one was the most fun I've had in years! – $13.95.

NIGHT RIDER - You've seen similar games in the arcades. You see a winding twisting road ahead as you try to make time and stay on the road. NIGHT RIDER uses machine code to generate excellent high speed graphics - by the same author as MINOS.
NIGHT RIDER – $12.95 cassette only

THIEF - Another machine code goody for the C1P cassette only. You must use mobile cannon to protect the valuable jewels in the middle of the screen from increasingly nasty and trigger happy thiefs. Fast action and fun for one or two players. THIEF $13.95 on C1 cassette only!

SUPPORT ROMS FOR BASIC IN ROM MACHINES – C1S/C2S. This ROM adds line edit functions, software selectable scroll windows, bell support, choice of OSI or standard keyboard routines, two callable screen clears, and software support for 32-64 characters per line video. Has one character command to switch model 2 C1P from 24 to 48 character line. When installed in C2 or C4 (C2S) requires installation of additional chip. C1P requires only a jumper change. – $39.95
C1E/C2E similar to above but with extended machine code monitor. – $59.95

Please specify system on all orders

This is only a partial listing of what we have to offer. We now offer over 100 programs, data sheets, ROMS, and boards for OSI systems. Our $1.00 catalog lists it all and contains free program listings and programming hints to boot.

VISA

AARDVARK TECHNICAL SERVICES, LTD.
2352 S. Commerce, Walled Lake, MI 48088
(313) 669-3110

Master Charge

OSI OSI

OI-80 Protect zero page address 1 and 2 for APU BASIC conversions, and jump to proper conversion routine.

OI-46 Convert APU variable whose start address is set in location 1 and 2 and place result in BASIC variable pointed to by location 123-124 (the BASIC variable X since this is the first variable called by the main program as given in LIST 2 below).

OI Entry to object code. Swap lower half of page zero to upper memory, jump to routine called from main program, swap back page zero, check for address range error, and return (warm start if error set).

OI+56 Convert BASIC variable pointed to by 123-124 to APU variable and set in four locations starting with that pointed to by 1-2.

OI+117 APU to BASIC conversion entry. Check if APU = 0, if so set X = 0, otherwise jump to OI-46.

OI+140 Check APU for error and busy status. If there has been an error (see part I), pull program counter off stack and exit.

OI+170 Read APU floating point number on top of APU stack to memory starting with location pointed to by 4-5.

OI+228 Write memory floating point number starting at location pointed to by 4-5 to top of APU stack.

BASIC Fixed Routines

When a machine code mathematical subroutine is run a few BASIC statements must be included in the main program. These are given in list 2. The first line makes sure X is at the head of the variable table by setting it equal to zero. It also sets OI. Subroutines 730 and 735 set the variable address bases for the variable A,B,C,D,E,F,G,H,T,X,Y,Z. That is, NF contains the relative address on page zero (after swapping) for these variables. For example, A starts at location 20, B at 24, etc., X at 56, Y at 60, and Z at 64. These subroutines are called before the main program in lines 8-700.

The main program written out in list 2 is used to run the mathematical test loop described below. Line 8 identifies the APU variable X, sets BASIC X = 1, and calls subroutine 770 which executes a USR jump to the fixed routine that converts between these variables. Similarly, line 9 causes APU variable A to be set equal to the constant 1.00013. Line 10 identifies a call to the subroutine J whose starting address is set in line 2 (the first subroutine always starts at location OI+512), then executes this jump. Finally, line 11 identifies a variable X that is converted, and then printed. In the conversion calls, first set Z$ equal to the desired variable name, then CALL 770 to go from BASIC to APU, or 780 for the inverse.

The fixed routines outlined above, and these BASIC overhead instructions, are sufficient to manage a large number of mathematical subroutines and APU variables. If there is a warm start after a mathematical subroutine call, a GOSUB750 will print out the error code and an object address of a place near where the error occurred.

An Example

Consider the multiplication test program discussed in part I. This called for consecutive multiplication of X by a specified constant A for 40,000 times. One program to do this would set A and X, and call the following mathematical subroutine.

```
SUB1
FORI = 1TO200
FOR J = 1TO200
X = X*A
NEXTJ
NEXTI
RETURN
END
```

Note that two nested FOR loops are needed to get 40,000 because integer variables are limited to a range of 0 to 255 each.

List 3 gives detailed description of the object code generated by the compiler when the above statements were entered as a source code. Note OI = 19968 for this example.

By inspecting this program you can see that the 6502 is used for loop control. The variable table is the same as was set in statement 733-734 of list 2. A is at loc 20, X at 56. Note that some 6502 statements are executed concurrently while the C8231 is multiplying (20541-20549). Writing short mathematical expressions like X = X*A does not allow much co-processing because you are primarily reading and

Listing 2

```
1 X = 0: DIM S(20):OI = 78 * 256
2 S(1) = 20480
5  GOSUB 730: GOSUB 735
6  REM  END HEADER. MAIN PROGRAM, LINES 8-700.
8 Z$ = "X":X = 1: GOSUB 770: REM  BEGIN MAIN PROGRAM, SET SUBR. VARIABLE X=1.
9 Z$ = "A":X = 1.00013: GOSUB 770: REM   SET MATH SUBROUTINE CONSTANT A=1.00013
10 J = 1: PRINT "START": GOSUB 760: REM  ENTER MATH SUBROUTINE
11 Z$ = "X": GOSUB 780: PRINT X: REM   PRINT FINAL VALUE OF X AFTER 40,000 MULTS
20  STOP
730  REM   VARIABLE ADDRESS BASES--SINGLE VARIABLES ONLY
731  DIM NF(26)
733  FOR J = 1 TO 8:NF(J) = 16 + 4 * J: NEXT J:NF(20 = 52:NF(24) = 56:NF(25) = 60
734 NF(26) = 64: RETURN
735  REM  SET CONSTANTS FOR OVERHEAD ROUTINES
738 S(10) = OI + 226:S(11) = OI + 227:S(15) = 256
739 S(14) = OI / 256 - 1:S(16) = OI + 23:S(17) = OI + 24:S(18) = OI - 59: RETURN
750  REM  ERROR CHECK
751  PRINT "ERROR CODE="; PEEK (OI + 262) AND 30
752  PRINT "ADDRESS="; PEEK (OI + 263) +  PEEK (OI + 264) * 256: PRINT : RETURN
759  REM  760 IS SUB CALL ENTRY J=SUB#
760  IF J > 9 OR J < 1 OR S(J) = 0 THEN  PRINT "ILL SUB CALL TO #";J: STOP
761 X = S(J):XS =  INT (X / 256): POKE S(17),XS: POKE S(16),X - 256 * XS
763  POKE 11,0: POKE 12,OI / 256:X =  USR (0): RETURN
770  REM  BASIC TO APU CONV, Z$=CHAR, J=INDEX, I=PAGE INDEX
771 XS =  ASC (Z$) - 64: IF XS <  > 10 THEN  IF NF(XS) <  > 0 THEN I = 1:J = 1
772  POKE 11,176: POKE 12,S(14): POKE S(10),NF(XS)
773  POKE S(11),79: IF X = 0 THEN  POKE S(18),102
774  IF X <  > 0 THEN  POKE S(18),56
775 XS =  USR (0): RETURN
780  REM  APU TO BASIC
781 XS =  ASC (Z$) - 64: IF XS <  > 10 THEN  IF NF(XS) <  > 0 THEN I = 1:J = 1
782  POKE 11,176: POKE 12,S(14): POKE S(10),NF(XS)
783  POKE S(11),79: POKE S(18),117
784 XS =  USR (0): RETURN
```

***List 3:* A sample object code (all addresses decimal).**

20480	162,1	LDX-IMM	1	
20482	202	DEX		
20483	134,10	STX-Z	10	initialize integer I (I at loc 10 page zero)
20485	166,10	LDX-Z	10	load I
20487	224,200	CPX-IM	200	I equal to 200?
20489	208,3	BNE	3	
20491	76,81,80	JMP	20561	If true jump out of For loop.
20494	232	INX		If I less than 200 increment.
20495	134,10	STX-Z	10	restore I
20497	162,1	LDX-IMM	1	
20499	202	DEX		
20500	134,11	STX-Z	11	initialize integer J (J at loc 11 page zero)
20502	166,11	LDX-Z	11	load J
20504	224,200	CPX-IM	200	J equal to 200?
20506	208,3	BNE	3	
20508	76,78,80	JMP	20558	If true jump out to next I
20511	232	INX		If J less than 200 increment J
20512	134,11	STX-Z	11	restore J
20514	169,56	LDA-IM	56	load address base for variable X (lo)
20516	133,4	STA-Z	4	put into zero page loc 4 (variable pointer)
20518	160,0	LDY-IM	0	
20520	132,5	STY-Z	5	put address base (hi) into loc 5
20522	32,228,78	JSR	20196	goto fixed routine to write X to top of APU stack
20525	169,20	LDA-IMM	20	load address base for variable A (loc 20, page 0)
20527	133,4	STY-Z	4	set address pointer
20529	160,0	LDY-IM	0	
20531	132,5	STY-Z	5	
20533	32,228,78	JSR	20196	write variable A to APU stack (to OI + 228)
20536	169,18	LDA-IM	18	load op code for multiply
20538	141,7,251	STA-AB	64263	command APU to multiply top of stack by next on stack, result to top of stack
20541	169,56	LDA-IM	56	set address base for variable X (loc 56 page 0)
20543	133,4	STA-Z	4	
20545	160,0	LDA-IM	0	
20547	132,5	STY-Z	5	
20549	32,140,78	JSR	20108	APU busy-error check (to OI + 140)
20552	32,170,78	JSR	20138	Read APU to memory (to OI + 170)
20555	76,22,80	JMP	20502	J loop return
20558	76,5,80	JMP	20485	I loop return
20561	96	RTS		return from subroutine 1.

writing from the APU. However, in longer calculations involving arrays and complicated indexing, time saved by co-processing can amount to a factor of 2 or more.

The above listings, along with this example, should give the reader enough information to write machine code subroutines by hand. The 6502 just implements, writes, and reads to and from the APU, sends it commands and checks its status. Standard 6502 operations can be used for loop control, jumps between subroutines, etc. It should be possible, without undo effort, to write out such object codes for fairly straightforward calculations. If you want to try this particular program the DATA list in listing 4 should be helpful.

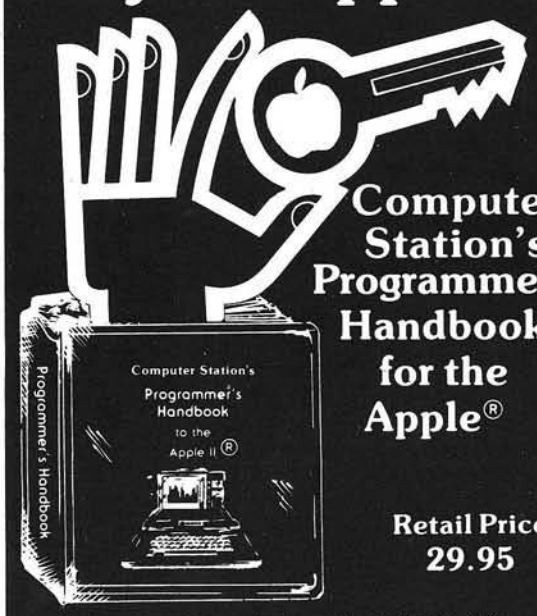
```
DATA 162,1,202,134,10,166,10,224,200,208
DATA 3,76,81,80,232,134,10,162,1,202
DATA 134,11,166,11,224,200,208,3,76,78
DATA 80,232,134,11,169,56,133,4,160,0
DATA  132,5,32,228,78,169,20,133,4,160
DATA 0,132,5,32,228,78,169,18,141,7
DATA 251,169,56,133,4,160,0,132,5,32
DATA 140,78,32,170,78,76,22,80,76,5
DATA 80,96
```

Of course, the ultimate situation is to have the compiler write out the object code as illustrated above. Clearly it takes each BASIC source statement and branches out to routines that parse through the line according to the fundamental operation (e.g. FOR, NEXT, a mathematical expression, etc.). The most complicated aspects of a compiler involve rewriting general mathematical expressions into a stack-processing type form suitable for the C8231, and in the process trapping any errors in the source code. The compiler is much too long to list here (16K of BASIC statements), or to describe in detail. However, I hope these two articles have illustrated how fast mathematical processing can be carried out on a simple micro at minimal cost. Enough material has been presented to write and execute simple mathematical subroutines. For further information (a complete manual and cassette tape) on the compiler please write the author.

Two years ago John Hart became interested in using a microcomputer to control laboratory experiments, and to do theoretical calculations involved with his research in meteorology and physical oceanography. The system described above has been used to solve a variety of problems concerned with flow over or around mountains and simple climate models.

MICRO

Here's the KEY to your Apple®!



Computer Station's Programmers Handbook for the Apple®

Retail Price 29.95

Indexed Looseleaf notebook (7½" x 9") containing all the reference material found in our popular Programmers Guide to the Apple II.®

Plus . . .

- Applesoft® & Integer
- CP/M® Digital Research, Inc.
- Basic-80 Microsoft
- Pascal
- 6502 Assembly Language
- DOS 3.3
- DOS Tool Kit
- Monitor

Including Command References for

- Applewriter®
- Visicalc® Personal Software
- Macro-Sced

Hardware Configurations & Software Commands for

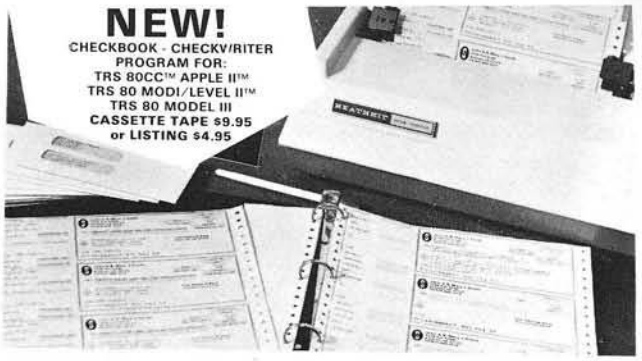
- Spinwriter
- PaperTiger
- Silentype
- Special ROMs

Two diskette pockets in front & back. Notebook format allows user to add personal comments. A must for every Apple® owner. Available from your local Apple® Dealer or from:

Computer Station
11610 Page Service Dr.
St. Louis, MO. 63141
(314) 432-7019

Direct order will incur a $2.00 shipping/handling charge plus sales tax where applicable

Apple, Apple II and Applesoft are the registered trademarks of Apple Computer, Inc.



NEW!
CHECKBOOK - CHECKWRITER PROGRAM FOR:
TRS 80CC™ APPLE II™
TRS 80 MODI/LEVEL II™
TRS 80 MODEL III
CASSETTE TAPE $9.95
or LISTING $4.95

NOW. . .Continuous Checks

That Can be Used With or Without Your Computer!!
The Best in A Home Checking System

That's right. Continuous Checks fan-folded in a 3-to-a-page desk set design. And they can be computer printed, handwritten or typewritten — whichever suits the quantity or situation.

SPECIAL DESIGN

Our checks are not a high-volume business form adaptation. They're specially designed **Computer/Manual Checks** for the home user. And they're **easy to use,** too. No need to change printer form width when loading. Our checks are the same **9½" width** as standard tractor feed printer paper. Check tear down size is the same as the classic **personal-sized check** issued by all banks.

COMPLETE HOME SYSTEM

With this system you can print the bulk of your monthly checks on your computer using our **program.** Your **checks** and stubs can then be stored in our attractive Data Ring Binder **Checkbook.** Later, if you have a few checks to write, there's no need to load them into a printer - just write a check at your desk as shown above. And you can mail your checks in our **dual windowed envelopes** to eliminate addressing chores.

UNIQUE

You won't find **continuous** checks like these anywhere. And, our special small quantity printing process will give you **excellent quality** and appearance. Standard **color-coordinated** imprinting and encoding is as shown above (logo can be omitted) on **blue, grey, green,** or **buff** checks.

PRICED RIGHT

Two Hundred checks are just **$29.95** (envelopes **$14.95**). Five Hundred checks are **$49.95** (envelopes **$27.95**). Data Ring Checkbooks are only **$4.95.**

Special "ORDER NOW" Offer

So that you can start using your computer immediately to pay your monthly bills, we'll make you a special package offer. If you order directly from this ad, we can send you:

200 Checks, 100 Envelopes, A Binder, and Program for $49.95
— OR —
500 Checks, 300 Envelopes, A Binder, and Program for $74.95

You'll save as much as **$9.85** over the separate purchase price. And with the special package you can begin your monthly checkbook balancing and bill paying as soon as you receive your checks. Just enclose a voided check (for encoding information) with your payment (VISA - MasterCard orders must show signature, expiration date, and account number). Or, send today for samples.

E-MAIL ADDRESS: SOURCE TCH 411 microNET-CIS 70270,236

SYNERGETIC SOLUTIONS
PHONE: (813) 646-6557

4715 SHEPHERD RD.
DEPARTMENT M91
MULBERRY, FL 33860

TRS 80™ Tandy, Inc. Apple II™ Apple Computer, Inc.

ATTENTION: SOFTWARE AUTHORS

OHIO SCIENTIFIC

(A M/A—COM COMPANY) IS SEEKING A

6502 based, full screen edit, cursor orientated word processor to run on its Challenger series microcomputers. Programs submitted should either currently run on O.S.I. products or should include parameters on adaptability. Ability to operate on both memory mapped video and serial CRT, as well as data file linkage, preferred.

Legal Authors should contact the official agents: —Department W.P., Anaheim Computer Distributing, 1150 Knollwood Circle, Anaheim, California 92801, U.S.A.

Any copyright information or preconditions should be stated at the time of submission of program. Both Ohio Scientific and Anaheim Computer Distributing will treat all communications in the strictest confidence.

For any discussion call Keith Beverton at 1-714/995-3041 between 11:30 a.m. and 12:30 p.m. (19.30 and 20.30 GMT) -Monday to Friday.

OSI COMPATIBLE HARDWARE

IO-CA10X SERIAL PORT $125
ACIA based RS-232 serial printer port. DIP SWITCH selectable baud rates of 300-9600. Handshaking (CTS) input line is provided to signal the computer when the printer buffer is full. Compatible with OS-65U V1.2 and OS-65D.

IO-CA9 PARALLEL PORT $175
Centronics Standard Parallel printer interface for OSI computers. The card comes complete with 10 ft. of flat ribbon cable. Compatible with OS-65D and OS-65U software.

IO-CA9D DIABLO PARALLEL PORT $175
DIABLO 12 BIT WORD Parallel port for use with word processor type printers. Complete with 10 ft. cable. Compatible with OS-65U software.

IO-LEVEL 3 MULTI-USER EXPANSION $450
Provides 3 printer interfaces currently supported by OSI-Serial, Centronics Parallel, Diablo Parallel. 4K of memory at D000 for Multi-user executive. 4 Port serial cluster. The LEVEL 3 card allows expansion of an OSI C3 machine up to 4 users with appropriate additional memory partitions.

24MEM-CM9...$380 **16MEM-CM9...$300** **8 MEM-CM9...$210**
24K memory card is available at 3 different populated levels. All cards are fully socketed for 24K of memory. The card uses 2114-300ns chips. DIP SWITCH addressing is provided in the form of one 16K block and one 8K block. Also supports DIP SWITCH memory partition addressing for use in multi-user systems.

FL470 FLOPPY DISK CONTROLLER $180
OSI-Type floppy disk controller and real time clock. Will Support 5¼" or 8", Single or double-sided drives. Requires drives with separated data and clock outputs.

BIO-1600 BARE IO CARD $50
Super 1/O Card. Supports 8K of 2114 memory in two DIP SWITCH addressable 4K blocks, 2 16 Bit Parallel Ports may be used as printer interfaces, 5 RS-232 Serial Ports with CTS & RTS handshaking. With manual and Molex connectors.

BMEM-CM9 BARE MEMORY CARD $50
Bare 24K memory card, also supports OSI-type real time clock and floppy disk controller. With manual and Molex connectors.

#96 PROTOTYPE CARD $35
Prototype board holds 96 14 or 16 pin IC's. Will also accommodate 18, 24, or 40 pin IC's. Row and column zone markings, easy layout. 1/16" epoxy glass P.C. board.

C1P-EXP EXPANSION INTERFACE $65
Expansion for C1P 600 or 610 boards to the OSI 48 Pin Buss. Uses expansion socket and interface circuitry to expand to 48 Pin Backplane. Requires one slot in backplane.

BP-580 BACKPLANE $47
Assembled 8-slot backplane with male Molex connectors and termination resistors.

DSK-SW DISK SWITCH $29
A circuit when added to OSI Minifloppy systems extends the life of drives and media. Accomplish this by shutting off Minifloppy Spindle motor when system is not accessing the drive. Complete KIT and manual.

PW-5-6 POWER SUPPLY $29
Power One brand supply 5V – 6 amps with overvoltage protection. Reg. $49.95.

D&N MICRO PRODUCTS, INC.

3684 N. Wells Street Ft. Wayne, Indiana 46808
219/485-6414

TERMS: Check or money order Add $2 Shipping, Outside U.S. add 10%.

Engineering

PROJECT ENGINEER ELECTRONICS DEVELOPMENT

If you appreciate the challenge of a fast-paced, professional environment, this opportunity within our Hyland Diagnostics Division may be of interest. As a leading international medical products manufacturer with facilities in operation throughout the world, we seek degreed, "hands-on" electronic technologists or engineers to modify and develop microprocessor controlled manufacturing equipment.

Qualified candidates will have shown ability to:

- **Successfully meet project objectives**
- **Design and prototype microprocessor controlled mechanisms**
- **Effectively interface with corporate support and manufacturing personnel**
- **Program 6502 Synertek Microchips**

Strong communications and interpersonal skills enabling you to work closely with all levels of our staff are essential.

This position is located at Travenol's Round Lake facility, offering ready access to Chicago, Milwaukee, and Wisconsin resort areas. We provide a competitive salary commensurate with experience. A generous benefits package includes profit sharing and employee stock purchase plan. If you seek a professional environment, send your resume with salary history and requirements in confidence to: Vincent T. Luber.

TRAVENOL

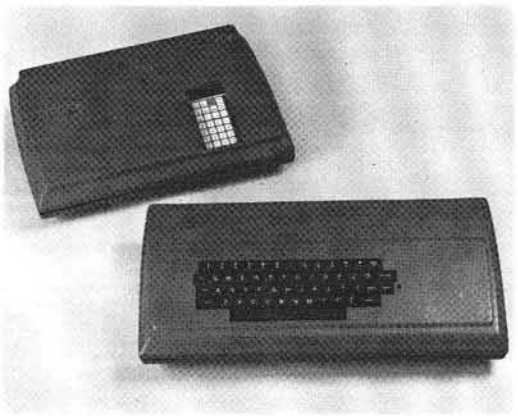
Equal Opportunity Employer

Travenol Laboratories, Inc.
Administration Bldg.
P.O. Box 490
Round Lake, Ill. 60073
(312) 546-6311

DYNAMIC DUO

Designed and Engineered Specifically for the SYNERTEK SYM-1 and KTM-2

- VITAL COMPONENTS PROTECTED
- ALL FASTENERS PROVIDED
- NO ALTERATION REQUIRED
- EASILY ASSEMBLED



ATTRACTIVE FUNCTIONAL PACKAGING:

- **High Quality Thermoformed Plastic***
- **Molded In Data Blue Color**
- **Available From Stock**

*Rohm & Hass - KYDEX 100

enclosures group

786 bush street
san francisco, california 94108

TO ORDER: 1. Fill in this coupon (Print or Type Please)
2. Attach Check or Money Order.

NAME _____

STREET _____

CITY _____ STATE _____ ZIP _____

FOR SYM-1: Please Ship Prepaid _____ **SSE 1-1**(s) @ $39.50 each
California Residents Please Pay $42.07 (Includes Sales Tax)

FOR KTM-2: Please Ship Prepaid _____ **SKB 1-3**(s) @ $69.50 each
California Residents Please Pay $74.02 (Includes Sales Tax)

TOTAL ENCLOSED: $_____

Dealer Inquires Invited. — No C.O.D.'s Please. — Allow 2-3 Weeks for Processing and Delivery.

MICRO Classified

SYM-1 Expansions

Bare W7AAY 4K RAM board: $8.00 plus SASE. Assembled W7AAY ROM board: $16.00. RAE symbolic disassembler source on cassette: $15.00. Instruction packet to add floppy disks to SYM -$15.00. RAE/KMMM software interface on 5¼" diskette: $15.00. SASE for more information.

John M. Blalock
Blalock & Associates
P.O. Box 39356
Phoenix, AZ 85069

Extended SYM-BASIC

Adds 30 commands, requires 16K, $85 US/$95 Can., object on cassette, manual, and source listing. SYM-FORTH 1.0: fig-FORTH for 16K SYM-1. Editor, assembler, cassette interface $135 US/$155 Can., object on cassette, manual and source listing.

Saturn Software Limited
8246 116A St.
Delta, B.C., V4C 5Y9
Canada

C1P Extended Monitor

2K EPROM has 14 cursor control/editing functions, improved keyboard decoding. Machine language save, load, display, modify, move, breakpoint processing and much more. For 24, 32, 64 char/line. $39.95 plus $1.00 shipping. $1.00 for complete information.

Bustek
P.O. Box A
St. Charles, MO 63301

PET/CBM Owners

Real world software at low cost. 2114 RAM adapter and 4K Memory Expansion for "old" 8K PETs. Write for free catalog!

Optimized Data Systems
Dept. M, Box 595
Placentia, CA 92670

PET Arcade Software

Astroidz and *Munchman* games for your 8K old-new ROMS. *Astroidz* are invading the galaxy. Four levels of play. *Munchman* is based on arcade game Pac-Man. ZIP and ZAP are out to get you. Fantastic graphics. $9.95 each cassette.

ComputerMat
Box 1664M
Lake Havasu, AZ 86403

Spanish Hangman

2,000 SPANISH words and sentences taught in a fun way on the Apple. Send for your school's free 30-day evaluation diskette, from:

George Earl
1302 South General McMullen
San Antonio, TX 78237

Ohio Scientific Tee-Shirts

Men's S, M, L in black with blue to white Logo, $7.50. We also have an XREF program (list variables and line #'s), disk $24/tape $6.98, and a true backspace, screen stall and clear for C1P at $6.98. Please add $.75 postage/handling on all orders.

Computer Business Service
P.O. Box 20384
San Jose, CA 95160

Business Software by ADS

For the Apple II and Atari/800. Why pay more for a bunch of unrelated programs? Business Plus will handle invoices, statements, credit memos and more, much more! Just $299 complete or $25 for demo disk (credited towards purchase). VISA, Mastercharge accepted.

Advanced Data Systems
7468 Maple Avenue
St. Louis, MO 63143
314/781-9388

Super-Maze

Attention OSI owners: Get *Super-Maze*, similar to the arcade game. Runs on 32 x 64 cassette systems; or *Super-Maze II* — a faster and more aggressive version. Send $1.50 for more information, or $14.95 for *Super-Maze*, $17.95 for *Super-Maze II*, $21.95 for both, $2.00 for hard copy only.

Vega Enterprises
1564 Locust Ave., Suite 101A
Long Beach, CA 90813

Free PET/CBM COMAL Interpreter

The battle between BASIC and Pascal may soon be over. COMAL is winning. COMAL is powerful and structured like Pascal, easy to learn and use like BASIC. COMAL interpreter is FREE with each User's Group diskette. Write for details.

COMAL User's Group
5501 Groveland Ter.
Madison, WI 53716

SeaFORTH for KIM/SYM/AIM/SEA Systems

Threaded subroutine, compiled (not interpreted) FORTH is much faster than fig-FORTH. Editor, assembler, compiler is 8K, ROMable. Floating point, disk extensions, OEM Target Compiler available. For the advanced FORTH programmer. Single-user license — $100 (cassette) or $100 plus ROMs.

Seawell Microsystems
P.O. Box 20367
Seattle, WA 98102
(206) 322-3123

OSI Superboard — Cabinet and Accessories

Pre-cut kit with hardware to build a handsome pine cabinet to house your superboard. Room inside for the power supply and all your extras, $20. RS-232 interface kit, $10. Send for our catalog of software, hardware, kits and accessories.

Dee Products
150 Birchwood Road
Carpentersville, IL 60110

MICRO™

CBM/PET? SEE SKYLES ... CBM/PET?

"You mean this one little Disk-O-Pro ROM will give my PET twenty-five new commands?

And for just $75.00? Why, that's only $3.00 a command!"

The Disk-O-Pro in any PET with Version III (BASIC 2.0) ROMs **(### COMMODORE BASIC ###)** will give 19 software compatible disk instructions*: 15 identical with the new BASIC 4.0 (or with 8032 ROMs) compatible with both old and new DOS. Plus 4 additional disk commands...including appending **(MERGE)**, overlaying **(MERGE #_____)** and **PRINT USING**, allowing formatting output of strings and numbers on the PET screen or on any printer.

**NOTE: Old DOS doesn't recognize three of the commands.*

Those are just 3 of the important commands—and there are 7 more beauties—on your Disk-O-Pro that have never been available previously to PET/CBM users. (Skyles does it again!)...Beauties like the softtouch key **(SET)** which allows you to define a key to equal a sequence of up to 80 keystrokes; like **SCROLL** whereby all keys repeat as well as slow scrolling and extra editing features; like **BEEP** which allows you to play music on your PET.

The Disk-O-Pro is completely compatible with the BASIC programmer's Toolkit. The chip resides in the socket at hexadecimal address $9000, the rightmost empty socket in most PETS. And for the owners of "classic" (or old) PETS, we do have interface boards.

(For those owning a BASIC 4.0 or 8032, even though the Disk-O-Pro may not be suitable, the Command-O is. Just write to Skyles for additional information. Remember, we have never abandoned a PET owner.)

Complete with 84-page manual written by Greg Yob...who was having so much fun that he got carried away. We had expected 32 pages.

Skyles guarantees your satisfaction: if you are not absolutely happy with your new Disk-O-Pro ROM chip, return it to us within ten days for an immediate full refund.

Disk-O-Pro from Skyles Electric Works **$75.00**

Complete with interface board (for "classic" PETS) **95.00**

Shipping and Handling ..... *(USA/Canada)* **$2.50** *(Europe/Asia)* **$10.00**

California residents must add 6%/6½% sales tax, as required.

Skyles Electric Works
231E South Whisman Road
Mountain View, California 94041
(415) 965-1735

Visa/Mastercard orders: call tollfree **(800) 227-9998** (except California). California orders: please call **(415) 965-1735.**

SEE SKYLES ... CBM/PET? SEE SKYLES ... CBM/PET?

CBM/PET? SEE SKYLES ... CBM/PET? SEE SKYLES ... CBM/PET?

PANORAMAS!

with

GRAF-PAK

© 1981

HI-RES GRAPHICS DUMP ROUTINES

Easy to Use! **Multiple Scale Factors!**

Precise Reproduction! Normal/Inverse Inking!

Dump either page one or page two, horizontally or vertically on the paper; or, dump both as a *two page panorama* with both pages butted in perfect registration. Compatible with I/O cards from Apple, Epson, SSM, Tymac, California Computer and Mountain Computer.

SCALE FACTORS PROVIDED

PRINTERS SUPPORTED	PANORAMIC or VERT 1X	2X	3X	4X	5X	HORIZ 1X	2X	3X	POST-PAID PRICE
DP9000	•	•				•			29.95
DP9001	•	•	•			•	•		39.95
DP9500	•	•	•	•		•	•		44.95
DP9501	•	•	•	•	•	•	•	•	49.95
MX70	•	•				•			29.95
MX80	•	•				•			29.95
MX100	•	•	•	•		•	•		44.95
IDS440	•	•				•			29.95
IDS445	•	•				•			29.95
IDS460	•	•	•			•	•		39.95
IDS560	•	•	•	•	•	•	•	•	49.95

SmartWare

2281 Cobble Stone Court Dayton, Ohio 45431

513/426-3579 *Dealer Inquiries Invited!*

POWER TO YOUR AIM

Treat Your AIM to a Quality Power Supply

1. Designed to Rockwell's specifications for the AIM 65 (5 volts at 2 amps, regulated; 24 volts, .5 amps avg., 2.5 amps peak, unregulated).
2. *Overvoltage protection* to protect the expensive circuits in your AIM (5 volt output).
3. Handsome all metal case (two-tone blue).
4. Fuse (externally accessible), switch, pilot light, line cord, cable from power supply to AIM — all included.
5. Conservative thermal design for long life.
6. Special on orders postmarked in August or September. $59.95 plus shipping (5 lbs.). (Regularly $64.95.)

CA residents add 6% sales tax.
VISA/MC, cashier's or registered check.
Personal check (allow 2 weeks to clear).

CompuTech
Box 20054
Riverside, CA 92516

GET FREE SOFTWARE FOR YOUR APPLE!!!

HOW? Just order any of the items below, and for every $100 worth of merchandise order an item from the Bonus Software Section at NO COST! C.O.D. & Personal Checks accepted for all orders.

HARDWARE BY APPLE

APPLE II PLUS, 48k	1199
DISK DRIVE+CONTROLLER (3.3)	535
DISK DRIVE only	445
Language System w. Pascal	397
Silentype Printer & Interface	549
Integer or Applesoft Firmware Card	159
Graphics Tablet	645
Parallel Printer Interface Card	149
Hi-Speed Serial Card	155

HARDWARE by Others

HAYES MICROMODEM II	300
VIDEX VIDEOTERM 80 W. GRAPHICS	320
MICROSOFT Z80 SOFTCARD	269
MICROSOFT 16k RAMCARD	159
CORVUS 10MB HARD DISK	CALL
SSM AIO SERIAL/PARALLEL A&T	189
MICRO-SCI Disk & Controller	495

VIDEO MONITORS

Leedex-Video-100 12" B&W w/Cable	139
Leedex 12" Green w/Cable	165
Leedex 13" COLOR MONITOR & cable	399

SOFTWARE by APPLE

APPLE FORTRAN	159
APPLE PILOT	125

HARDWARE by Mountain Computer

Clock/Calendar Card	239
A/D & D/A Interface	319
Expansion Chassis	555
ROMplus Card	135
Mark Sense Card Reader	995

SOFTWARE by Others

PEACHTREE BUSINESS SOFTWARE	CALL
VISICALC	120
EZ WRITER PROF. SYSTEM	229
APPLE FORTRAN by MICROSOFT	159
APPLE BASIC COMPILER by MICROSOFT	315
APPLE COBOL by MICROSOFT	599
MUSE SUPER-TEXT II	139
PROGRAMMA APPLE PIE	119

PRINTERS

EPSON MX-80	515
EPSON MX-70 W. GRAPHICS	415
CENTRONICS 737	737
NEC SPINWRITER 5510 RO	2795
VISTA V300 DAISY WHEEL 25CPS	1750
VISTA V300 DAISY WHEEL 45CPS	2025

BONUS SOFTWARE HERE!

Let us acquaint you with MESSAGE-MAKING SOFTWARE. Just place the disk in the APPLE, enter the text, and colorful, dynamic messages appear on the screens of TV sets connected to the computer. Use the software to broadcast messages on TV screens in schools, hospitals, factories, store windows, exhibit booths, etc. The following program is our latest release:

SUPER MESSAGE: Creates messages in full-page "chunks". Each message allows statements of mixed typestyles, typesizes and colors, in mixed upper and lower case. Styles range from regular APPLE characters, up to double-size, double-width characters with a heavy, bold font. Six colors may be used for each different typestyle. Vertical and horizontal centering are available, and word-wrap is automatic. Users can chain pages together to make multi-page messages. Pages can be advanced manually or automatically. Multi-page messages can be stored to disc or recalled instantly.

REQUIRES 48K & ROM APPLESOFT ..... $ 50.

APPLE PLOTS YOUR DATA & KEEPS YOUR RECORDS TOO

APPLE DATA GRAPH 2.1: Plots up to 3 superimposed curves on the Hi-res Screen both the X & Y axes dimensioned. Each curve consists of up to 120 pieces of data. Graphs can be stored to disc and recalled immediately for updating. Up to 100 graphs can be stored on the same disc. Great for Stock-market Charting, Business Management, and Classroom Instruction!

REQUIRES 48 K & ROM APPLESOFT ..... $ 40.

APPLE RECORD MANAGER: Allows complete files to be brought into memory so that record searches and manipulations are instantaneous. Records within any file can contain up to 20 fields, with user-defined headings. Information can be string or numeric. Users can browse thru files using page-forward, page-backward or random-search commands. Records can easily be searched, altered or sorted at will. Files can be stored on the same drive as the master program, or on another. If a second drive is available. Records or files can be printed, if desired.

Additional modules coming are a STATISTICS INTERFACE, CHECKBOOK, MAILING LIST & DATA-ENTRY.

REQUIRES 48K & ROM APPLESOFT ..... $ 35.

* All Software above on Disk for APPLE DOS 3.2, convertible to 3.3.

CONNECTICUT INFORMATION SYSTEMS CO.
218 Huntington Road, Bridgeport, CT 06608 (203) 579-0472

MICRO

Updates and Microbes

Charles Schwarz of Bethesda, Maryland, sent this note:

I was very annoyed to discover that the assembly language program by Steve Emmett on pages 38-39 of your July, 1981 issue (38) has been cut off. I very much enjoy reading about well-tested assembly language programs in your magazine, but errors such as this one make the reading very difficult.

We regret our mistake. See Emmett's listing below:

Emmett Listing

```
08D1 8D1008            STA BUFHI
08D4 EE1208            INC RWS          ;RWS TO WRITE
08D7 AD0D08            LDA CTRK
08DA 8D1708            STA IBTRK
08DD AD0E08            LDA CSCT
08E0 8D1808            STA IBSECT
08E3 AD1008            LDA BUFHI
08E6 8D1C08            STA IBBUFH
08E9 AD1208            LDA RWS
08EC 8D1F08            STA IBCMD
08EF 60          RTN   RTS
08F0             ;
08F0 A901        END   LDA #$01          ;RESET TEMPORARY
08F2 8D0F08            STA CDIO         ;STORAGE AND
08F5 8D1108            STA NTRK         ;IOB TO
08F8 8D1208            STA RWS          ;INITIAL
08FB 8D1F08            STA IBCMD        ;CONDITIONS
08FE A903              LDA #$03
0900 8D0D08            STA CTRK
0903 8D1708            STA IBTRK
0906 AD0C08            LDA BUFAB
0909 8D1008            STA BUFHI
090C 8D1C08            STA IBBUFH
090F A900              LDA #$00
0911 8D0E08            STA CSCT
0914 8D1808            STA IBSECT
0917 8D2008            STA IBSTAT
091A A90F              LDA #$0F         ;SET END FLAG
091C 8D2808            STA FLAG
091F 60                RTS
                       END
```

John Martin of Cleveland Heights, Ohio, called to tell us of an omission we made from listing 1 of Monobyte Checksum Dumper for C1P *by Peter Broers in MICRO (38:68). The rest of the listing follows:*

Broers Listing

```
                ;
1EDF 20E71E     MONOUT JSR HEXOUT       ;SUBROUTINE TO DUMP A BYTE AS
1EE2 A90D              LDA #$0D         ;2 HEX DIGITS + CR, I.E.
1EE4 4CB1FC            JMP SAVBYT       ;"MONITOR LOADABLE FORMAT"
1EE7            ;
1EE7 48         HEXOUT PHA              ;SUBROUTINE TO PRINT (AND SAVE) BYTE
1EE8 4A                LSR              ;AS TWO HEX DIGITS
1EE9 4A                LSR
1EEA 4A                LSR
1EEB 4A                LSR
1EEC 20F01E            JSR DIGOUT
1EEF 68                PLA
1EF0            ;
1EF0 290F       DIGOUT AND #$0F         ;SUBROUTINE TO PRINT (AND SAVE)
1EF2 0930              ORA #$30         ;A HEX DIGIT
1EF4 C93A              CMP #$3A
1EF6 9002              BCC *+4
1EF8 6906              ADC #$06
1EFA 4CEEFF            JMP BYTOUT
                       END
```

Warren Ward of Alberta, Canada, sent another update to the Superboard article.

For C1P and Superboard owners who want to use Edward H. Carlson's mini-assembler (MICRO, March 1981), here are a few line changes that suit his program to the smaller screen format:

```
1 FOR X=0 to 25:PRINT:NEXT X:GOTO 1990
20 FOR Z=2 TO LEN(C$) : POKE N+Z, ASC(MID$(C$,Z,1)): NEXT:RETURN
99 C$="No":N=Q+1:GOSUB 20:
100 PRINT:PRINT AD;:INPUT C$: PRINT">":L$=LEFT$(C$,3): L=LEN(C$)
106 IF L$="ASC"THEN M=ASC (C$):Z=1:GOSUB 2: GOTO 100
1995 Q=54084
4050 N=N+M*L:L=L/16:NEXT:C$= STR$(N):N=Q:GOSUB 20: GOTO 100
```

Line 1 is cosmetic — it scrolls the screen for a clean start. Line 1995 relocates assembler comments into the C1P video memory, and changes the other lines to reposition the comments so they'll all fit on the screen.

It's impossible to squeeze a full comment sequence legibly into the same 24-character line as the command input, so the third PRINT statement in 100 starts a new line, preceded by a "greater-than" sign, beneath the address line. The first PRINT in 100 puts a space between each pair of lines for greater readability: to fit more information on the screen, leave it out.

The display resulting from these changes is almost as easy to read as the original. Users will still have to keep a notebook handy, though, if they want to save the assembler's translation before it disappears off the top of the screen.

Cliff Harris of Anaheim, California, wrote about his update:

I was intrigued by Edward Carlson's *6502 Assembler in BASIC* in the March, 1981 issue of MICRO. I thought if he could adopt the program from PET BASIC to OSI, then I could transform it into something my Apple could understand. See listing 1.

Listing 1

```
  1 HOME: GOTO 1990
 99 C$="NO": N=Q+18: GOSUB 20
100 PRINT AD;: INPUT C$: L$ =
    LEFT$ (C$,3): L = LEN(C$): Q =
    Q + 128: IF Q > 2000 THEN Q
    = Q - 984: IF PEEK (37) > 20
    THEN Q = 1888
106 IF L$ = "ASC" then M = ASC
    (C$): Z = 18: GOSUB 2: GOTO
    100
221 II = OP + 8 * (CA = 1)
340 N = N - AD - 2: IF N <
    -128 OR N > 127 THEN PRINT
    "CAN'T BRANCH "; N; " -- TOO
    FAR": Q = Q + 128
1992 DEF FN H(D) = D + 48 - 57 *
    (D > 9)
1995 Q = 912
2029 Delete this line
2030 AD = 768: GOTO 100
4050 N = N + M * L: L = L/16:
    NEXT: C$ = STR$ (N): N = Q
    + 18: GOSUB 20: GOTO 100
```

Line 1 cleans up all the garbage on the screen and sets up the screen format. This is necessary since the hex addresses and commands are POKEd directly into the screen memory.

Lines 99, 106, and 4050 move the output from the assembler to result in a format that will fit on the Apple screen.

In line 100, the changes are required because of the way the screen locations are arranged in the Apple. Adding 128 to a screen location moves it down one line, unless you're on the 8th or 16th line. Then you must subtract 984 to move down one line. The Q = 1888 sets the screen into a scrolling mode once you reach the bottom of the screen.

Line 221 required a change in sign. The minus was changed to a plus. In line 34 I added "--TOO FAR" here to make the message more meaningful. The Q = Q + 128 moves the line position down one so that your next program line won't be printed in the middle of the "CAN'T BRANCH..." message.

In line 1992 I changed the 7 to 57 to get this line to work with the Apple. A side effect of this is that the hex portions of the program will be printed in the inverse mode. If you want to take the time to massage this function, you can get a display in the normal mode, or even flashing, if you're so inclined. I left it this way to minimize the number of changes in the program.

In line 1995 the number sets up the screen location where the output from the assembler will be printed on the screen. When 128 is added by line 100, you will be on the top line of the screen 16 spaces from the left (912 + 128 = 1040, which equals 1024 + 16).

Line 2029 can be deleted, as it seems to be left over from a decimal-to-hex conversion routine which is no longer part of the program.

In line 2030, AD = 768 sets the starting address of your program to $0300. You have only 255 bytes available before you run into the screen memory at $0400. If your programs are going to be longer than 255 bytes, change AD to 8192 ($2000), or whatever address suits your needs, to get into an area of memory with no conflicts.

If you want to add a "user's manual" to the top three lines of the screen, change line 1995, and add lines 1996 through 1999 (listing 2).

Listing 2

Change the following lines to add a "menu:"

```
1995 Q = 1296: POKE 34,0: HOME
1996 INVERSE: PRINT "ADD";:
     NORMAL: PRINT "NEW
     ADDRESS";: INVERSE: PRINT
     "CON";: NORMAL: PRINT
     "STORE CONSTANT"
1997 INVERSE: PRINT "DIS";:
     NORMAL: PRINT "DISPLAY
     MEMORY";: INVERSE: PRINT
     "ASC";: NORMAL: PRINT "ASCII
     EQUIVALENT"
1998 INVERSE: PRINT "HEX";:
     NORMAL: PRINT "CONVERT
     FOUR DIGIT HEX TO DECIMAL"
1999 POKE 34,3
```

The POKEs set the text window so that you can clear the screen (line 1995) and so that you won't lose your mnemonics when you get to the bottom of the screen (line 1999).

Eric R. Bean of South Bend, Indiana, pointed out this omission:

My letter to the editor in July, 1981 on page 19, mentioned a figure 1, which was not printed. Here is another copy of the clock oscillator fix for the early KIM-1 uP board (see figure 1).

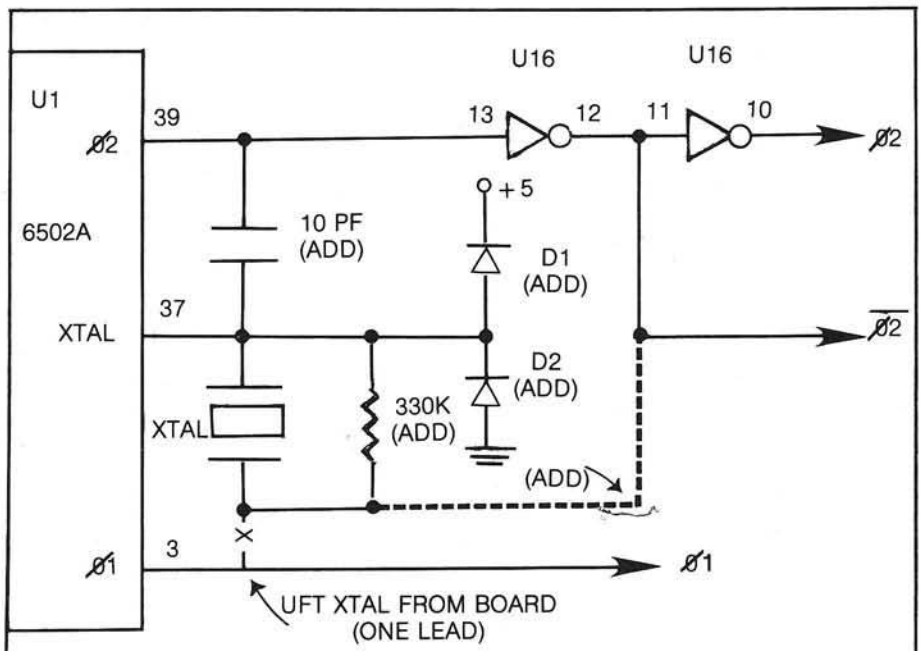


Figure 1: **Conversion of early KIM-1 XTAL oscillator to later production circuit to insure self starting.**

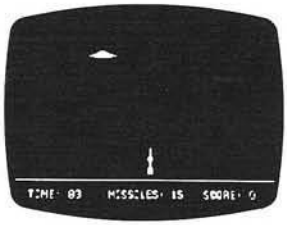
Apple II

sensational software

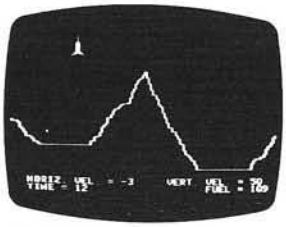
creative computing software

Space Games-I

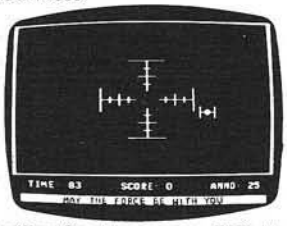
Cassette CS-4001 $11.95 4 programs Requires 16K Apple II or Apple II Plus



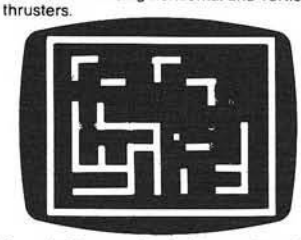
Saucer Invasion. Fire missles to destroy the invaders who fly at different speeds and altitudes.



Rocket Pilot. Maneuver your spaceship over the mountain using horizontal and vertical thrusters.



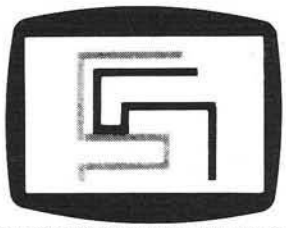
Star Wars. Shoot down as many TIE fighters as possible in 90 seconds.



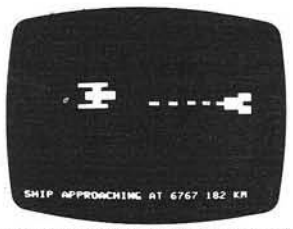
Dynamic Bouncer. A colorful, ever-changing graphics demonstration.

Strategy Games

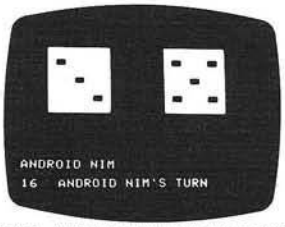
Cassette CS-4003 $11.95 4 Programs Requires 16K Apple II or Apple II Plus



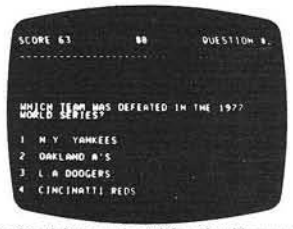
Blockade. Build a wall to trap your opponent, but don't hit anything.



UFO. Use lasers, warheads or guns to destroy an enemy spacecraft.



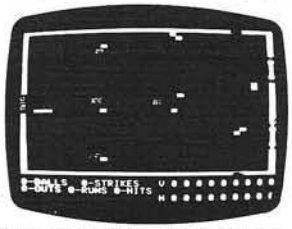
Skunk. A 2-player strategy game played with dice, skill and luck.



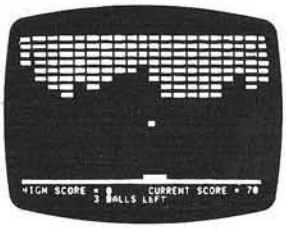
Genius. A fast-moving trivia quiz with scores of questions.

Sports Games-I

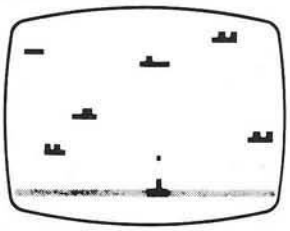
Cassette CS-4002 $11.95 4 programs Requires 16K Apple II or Apple II Plus



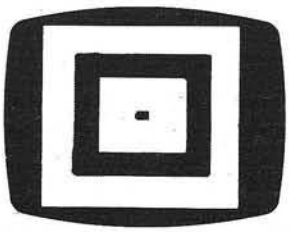
Baseball; A 2-player game with pitching, batting, fielding, stealing and double plays.



Breakout. Four skill levels and improved scoring make this the best breakout ever.



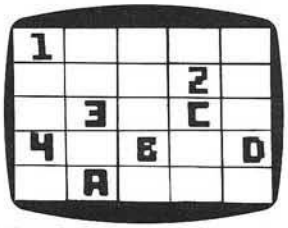
Torpedo Alley. Sink as many warships as possible in 2 minutes.



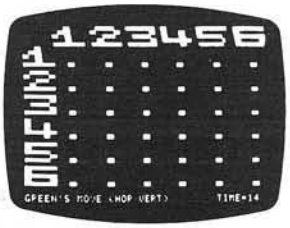
Darts. Use game paddles to control the throw of 6 darts.

Brain Games

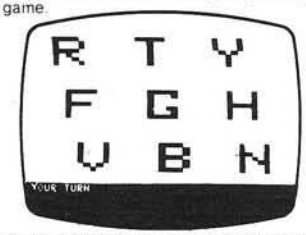
Cassette CS-4004 $11.95 7 programs Requires 16K Apple II or Apple II Plus



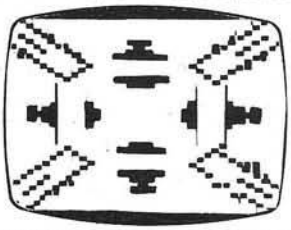
Dodgem. Be the first to move all your pieces across the board in this intriguing strategy game.



Nuclear Reaction. A game of skill, fast decisions and quick reversals of position.



Parrot. A Simon-type game with letters and tones. **Dueling digits** is a version with numbers.



Midpoints and Lines. Two colorful graphics demonstrations. **Tones** lets you make music and sound effects.

Space & Sports Games

Disk CS-4501, $24.95
Requires 32K Apple II or Apple II Plus

This disk contains all eight games from cassettes CS-4001 and CS-4002.

Strategy & Brain Games

Disk CS-4502, $24.95
Requires 32K Apple II or Apple II Plus

This disk contains all 12 games and programs from cassettes CS-4003 and CS-4004.

Apple is the registered trademark of Apple Computer, Inc.

Order Today

To order any of these software packages, send payment plus $2.00 postage and handling per order to Creative Computing, Morris Plains, NJ 07950. Attn: Clarice
Visa, MasterCard and American Express orders may be called in toll-free.

Order today at no risk. If you are not completely satisfied, your money will be promptly and courteously refunded.

Creative Computing Software
Morris Plains, NJ 07950
Toll-free **800-631-8112**
In NJ, 201-540-0445

creative computing software

PET & APPLE II USERS

TINY PASCAL

Plus+

GRAPHICS

The TINY Pascal System turns your APPLE II micro into a 16-bit P-machine. You too can learn the language that is slated to become the successor to BASIC. TINY Pascal offers the following:

- **LINE EDITOR to create, modify and maintain source**
- **COMPILER to produce P-code, the assembly language of the P-machine**
- **INTERPRETER to execute the compiled P-code (has TRACE)**
- **Structured programmed constructs: CASE-OF-ELSE, WHILE-DO, IF-THEN-ELSE, REPEAT-UNTIL, FOR-TO/DOWNTO-DO, BEGIN-END, MEM, CONST, VAR ARRAY**

Our new TINY Pascal PLUS+ provides graphics and other builtin functions: GRAPHICS, PLOT, POINT, TEXT, INKEY, ABS AND SQR. The PET version supports double density plotting on 40 column screen giving 80 x 50 plot positions. The APPLE II version supports LORES and for ROM APPLESOFT owners the HIRES graphics plus other features with: COLOR, HGRAPHICS, HCOLOR, HPLOT, PDL and TONE. For those who do not require graphics capabilities, you may still order our original Tiny Pascal package.

TINY Pascal PLUS+ GRAPHICS VERSION-

PET 32K NEW Roms cassette	$55
PET 32K NEW Roms diskette	$50
APPLE II 32K/48K w/DOS 3.2 or 3.3	$50

TINY Pascal NON-GRAPHICS VERSIONS-

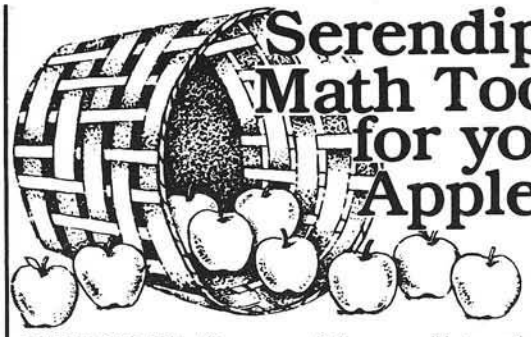
PET 16K/32K NEW Roms cassette	$40
PET 16K/32K NEW Roms diskette	$35
APPLE II w/ROM Applesoft 32K w/DOS	$35
APPLE II w/RAM Applesoft 48K w/DOS	$35
USER's Manual (refundable with software order)	$10
6502 Assembly Listing of INTERPRETER-graphics	$25
6502 Assembly Listing of INTERPRETER-non graphics	$20

master charge

VISA

FREE postage in U.S. and CANADA. Orders may be prepaid of by bankcard (include card number and expiration date). Michigan residents include 4% state sales tax. Orders accepted via THE SOURCE - CLO852.

ABACUS SOFTWARE
P.O. Box 7211
Grand Rapids, Michigan 49510
(616) 241-5510



Serendipity Math Tools for your Apple II™

INTER-STAT™ offers you a full range of interactive statistical analysis techniques, from averages and medians to binomial and poisson distributions, correlation coefficients and one- and two-way analysis of variance. $169.

ADVANCED MATH ROUTINES is the mathematical tool kit for common, yet complex numerical problems. Routines include: linear regression, matrix operations, numerical calculus, differential equations and data set recall for iterative calculations. $169.

Thoroughly tested, well documented and easy to master, each package includes a 30+ page self-teaching manual.
Serendipity's complete line of software solutions for business, education and professional applications are available at your local Computerland or Apple dealer.
For a free brochure, or to order direct contact Serendipity Systems, 225 Elmira Road, Ithaca, NY 14850.
Phone 607-277-4889. Visa and MC accepted.

™Apple Computer

SERENDIPITY SYSTEMS

THE ultimate in SPEED and language POWER for the APPLE II:

THE INTEGER BASIC COMPILER

- SPEED IMPROVEMENT BY A FACTOR OF 10 TO 20 OVER APPLE'S BASIC INTERPRETER - 15 TO 30 OVER APPLESOFT!
- NO LANGUAGE CARD NECESSARY !
- OPTIMIZE CODE FOR YOUR SPEED/SPACE REQUIREMENTS.
- OBJECT CODE AND RUN-TIME SYSTEM ARE COMPLETELY RELOCATABLE - USE MEMORY THE WAY YOU WANT TO !
- MANY POWERFUL BASIC LANGUAGE EXTENSIONS:
  - FULL STRING LENGTH OF 32767 - NO 255 LIMIT !
  - CHR$, GET AND KEY FUNCTIONS.
  - DIRECT HI-RES GRAPHICS SUPPORT.
  - HOME, INVERT, NORMAL, FLASH, AND MORE !
- MANY APPLICATIONS - EXISTING INTEGER PROGRAMS CAN BE EASILY CONVERTED TO RUN ON ANY APPLE II !
- COMPILER REQUIRES: APPLE II (OR II PLUS WITH INTEGER OR LANGUAGE CARD), 48 K AND DOS 3.3.
- SUPPLIED ON 2 DISKS WITH COMPLETE DOCUMENTATION.

PRICE: $ 149.50

Dealer inquiries invited.
(Calif. add 6.5% sales tax, Foreign add $5.00 air mail)

GALFO SYSTEMS 6252 CAMINO VERDE
SAN JOSE, CA 95119

* Apple, Applesoft - Trademarks of Apple Computer Co.

6800/6809 SOFTWARE CATALOG

PROGRAM	LANGUAGE	OBJECT	W/SOURCE ON DISK
X-FORTH	6809/6800		***$149.95
Dataman	TSC XBASIC		$149.95
Datarand	TSC XBASIC		49.95
*Bill Payer	TSC XBASIC		89.95
*Purchase Order	TSC XBASIC		49.95
*Income/Expense	TSC XBASIC		49.95
All Three	TSC XBASIC		169.95
Basic Prog. Toolkit	6809 ASMB	$49.95	69.95
Password Protection	6809 ASMB	69.95	89.95
Extended Utilities	6809 ASMB	49.95	69.95
Job Control Prog.	6800/6809 ASMB	49.95	89.95
Esther	6800/6809 ASMB	39.95	59.95
Readtest	6800/6809 ASMB	54.95	74.95
Help	6800/6809 ASMB	29.95	49.95
Dynasoft Pascal	6809	59.95	** 89.95
Plot	TSC XBASIC		44.95
Read TRS80 Tapes	6809 ASMB		54.95
Super Sleuth	6800/6809		99.00
Z80 Super Sleuth	6800/6809		99.00
Cross Assemblers	MACROS FOR TSC 6809 ASMB		EA. 49.95
	6800/1, 6805, 6502, Z-80, 8080/5	3 for	99.95
Mailing List	TSC XBASIC/6809		99.95
Forms Display	TSC XBASIC/6809		49.95
Tabula Rasa	TSC XBASIC		100.00

** SOURCE AND REPRODUCTION LICENSE-RUNTIME ONLY.
***Includes everything but the core.

U.S.A. add $2.50 for Standard UPS Shipping & Handling
Foreign orders add 10% Surface, 20% Airmail.

Specify 5" or 8" size disk and if for 6800 or 6809 system.

OUR SOFTWARE IS CMX COMPATIBLE.

OS-9 VERSIONS TO COME.

master charge VISA

FRANK HOGG LABORATORY, INC. 130 MIDTOWN PLAZA • SYRACUSE, N.Y. 13210 (315) 474-7856

MICRO

Mike Rowe
34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

Hardware Catalog

Name: **Hayes Stack Smartmodem**
System: Machine independent—RS-232 compatible
Hardware: Low Speed Modem
Language: Program controlled in any language.
Description: RS-232 compatible, 300 baud data communications system for small computers. Features program control in any language switch selectable options, full or half duplex and LED status indicators.
Price: $279.00 (suggested retail)
Available: Contact address below for nearest retail dealer.
Hayes Microcomputer Products, Inc.
5835 A Peachtree Corners East
Norcross, GA 30092
(404) 449-8791

Name: **Model 2100**
Memory: Standard 2K buffer memory; 4K option
Description: The Model 2100 is the European version of the Model 2101 having a built-in CCITT interface. Like the Model 2101 it also features: 5x9 dot matrix characters with true upper/lower case and true underscore/overscore; standard 80/132 selectability and bidirectional 120 cps printing.
Price: $1385
Available: Computer Devices Inc.
(early 4th Quarter; 60 days ARO)
25 North Avenue
Burlington, MA 01803
(Call: 1-800-225-1380)

Name: **W7AAY 4K RAM Board**
System: Synertek SYM-1
Description: Double sided reflow solder plated printed circuit board mounts on SYM-1 over Synertek name and logo. Allows memory expansion up to 8K using 2114 RAMs. Full instructions included.
Price: $8.00 ea. plus SASE
Available: John M. Blalock
Blalock & Associates
P.O. Box 39356
Phoenix, AZ 85069

Name: **Atari I/O Package**
Description: The four ports on the front of the Atari computer connect directly to a PIA for use as output as well as input. Atari owners can build custom program controllers, interface to home control circuits, etc. The I/O package comes with 4 nine-pin connectors, 4 twelve-inch lengths of nine conductor ribbon cable, and documentation. The documentation includes examples of home-built program controllers, how to access the ports through BASIC commands, shadow registers, or directly, and how to set-up and address the ports for output.
Price: $18.00 order #H309
Available: Mosaic Electronics
P.O. Box 748
Oregon City, OR 97045

Name: **Universal Analog Interface Card with A/D, Clock and Memory Expansion**
System: AIM-65, also applicable to PET, SYM, KIM and other 6502 and 6800 systems
Memory: 4K to 16K
Language: BASIC or Assembly
Hardware: AIM-65 or PET, SYM, KIM and other 6502 and 6800 systems plus Columbus Instruments IB-902-AB Card
Available: Columbus Instruments International Corporation
950 N. Hague Avenue
Columbus, OH 43204

Name: **Flexi Plus**
System: Stand Alone or Apple, AIM, SYM, KIM
Memory: Up to 56K RAM, ROM and EPROM
Description: Multi-function board includes floppy disk controller for 8" and 5¼" drives with IBM formats; IEEE-488 bus controller; RS-232 communications interface; 20 mA current loop interface; parallel and serial I/O ports; cassette interface; up to 56K bytes ROM, RAM and EPROM; and a 6809 microprocessor. May be used to expand existing 6502/6809 systems or as a complete single-board microcomputer.
Price: $695 with all options and 4K memory
Available: The COMPUTERIST, Inc.
34 Chelmsford St.
Chelmsford, MA 01824
(617) 256-3649

Name: **Covox Model 1 Voice Controller**
Language: Human Voice Input
Description: Tolerant of noise and distortion, a revolutionary self-contained speech recognition processor accurately identifies voicing existence, voice fundamental pitch, voicing duration, and vowel type in the manner of a human listener. In the stand-alone mode, this device will recognize 16 separate commands. When interfaced to a processor, such as a 6502, the system becomes highly flexible and can be adapted for continuous speech recognition, speech bandwidth compression, speech synthesis, and aids for the handicapped.
Price: $389.00
Available: Covox Company
P.O. Box 2342
Santa Maria, CA 93455
(805) 937-9545 or 928-4818

Name: **UDS-100 Series Memory I/O expansion boards for AIM 65**
Description: Two independent, baud rate selectable, asynchronous, RS-232-C channels and 20 independently programmable parallel I/O lines. Memory includes 4K bytes of 18-pin NMOS/CMOS RAM and 6 24-pin sockets accepting 1, 2, 4, or 8K × 8 RAM, ROM, PROM or EPROM devices. Full on-board bus signal buffering is included. Memory IC's and battery backup optional
Price: $259.00 basic assembly; $296.00 with battery back-up
Available: Unique Data Systems, Inc.
15041 Moran Street
Westminster, CA 92683
(714) 895-3455

Name: **Microlab**
Memory: 2K bytes of user RAM
Language: Assembly
Description: Complete educational package that includes hardware, software, and course materials for introducing microcomputers and performing laboratory experiments. Analog-to-digital conversion, interface for oscilloscope graphics, eight applications programs. Games, counter/timer, function generator, transient recorder, cooling curve and other applications
Price: $650 - $850
Available: Cambridge Development Laboratory
36 Pleasant Street
Watertown, MA 02172
(617) 926-0869

Name: **Microcomputer Control System (MCS)**
System: Rockwell International AIM 65
Description: The MCS is based upon the AIM 65 and is a complete microcomputer control system. It features three additional interface boards and firmware for real-time recording and controlling of external devices such as switches, solenoids, lights and alarms. Each MCS contains 16 input and 16 output channels that are rated at 28 VDC (at 3 Amps) and are completely optically isolated and noise suppressed. Efficient recording and controlling of external devices is accomplished using BASIC and interrupt-driven firmware package. This firmware adds 36 real-time commands to BASIC and allows "foreground" and "background" programming. While the MCS was specifically designed for the behavioral research psychologist it can also be used in other applications such as industrial control, alarm and environmental systems.
Available: Micro Interfaces, Inc.
P.O. Box 14520
Minneapolis, MN 55414

Name: **CHIEFTAIN™ 98W10**
Memory: 32K RAM (expandable)
Language: BASIC 09; Random File BASIC; Pascal Compiler; Cobol
Description: Smoke Signal, manufacturers of computer systems based on the 6800/6809 processors, has introduced the latest addition to the CHIEFTAIN™ Series of computer systems. Designated the CHIEFTAIN™ 98W10, this newest addition to the higher end of Smoke Signal Broadcasting's business computer line houses a 10 megabyte 8-inch Winchester Disk Drive. The new system is configured around the state-of-the art 6809 microprocessor allowing programs to run at twice the speed of any other similar system. The wide range of programs available for the CHIEFTAIN™ 98W10 include OS-9 Level I and Level II multi-user, multi-tasking operating system. A standard CHIEFTAIN™ 98W10 incorporates 32K of RAM — expandable up to 1 megabyte for specific requirements such as OS-9 Level II. The CHIEFTAIN™ 98W10 supports an 8-inch floppy disk drive for 1 megabyte of back-up storage. A 20 megabyte tape streamer option is also available. Dealer inquiries invited, discounts available.
Price: $8695.00 base
Available: Smoke Signal Broadcasting
31336 Via Colinas
Westlake Village, CA
91362

Name: **Terrapin-Apple Smart Interface**
Description: Terrapin, Inc. announces a smart Terrapin-Apple Interface for its robot, the Turtle. Now any Apple owner can be one of the first persons to own a robot. The interface enables the user to conveniently control the Turtle from a high level language (BASIC, Pascal, Logo, etc.) via simple I/O statements. The interface includes a parallel port with software in ROM and a power supply.
Price: $199.95
Available: Terrapin, Inc.
678 Massachusetts Ave.
Cambridge, MA 02139

The Hardware Catalog announcements are run free of charge, but are limited to only one per company, each month.

If you have hardware you'd like to announce to MICRO's readers, send for an application form.

Hardware Catalog
34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

MICRO

DEPT. E-9 P.O. BOX 30160 EUGENE, OR 97403 (503) 345-3043/NOON-7 PM

AVANT GARDE CREATIONS

SUPER DRAW & WRITE Fonts, drawing, and useful utilities. **19.95**

SUPER SHAPE DRAW & ANIMATE The best system yet, it works. . . create and/or animate shape tables like a dream. **39.95**

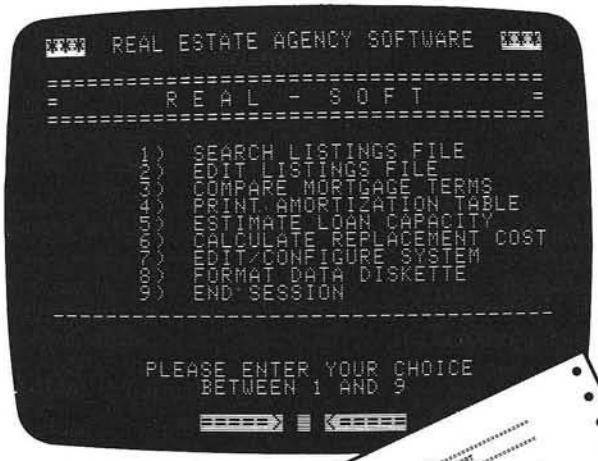
THE CREATIVITY TOOL BOX Draw, write poetry, music. Includes Action Sounds, Hi-Res Scrolling, routines, shape tables and shape view program, utilities, animation demo, and fonts. 3 diskfuls, 88 page manual. **44.95**

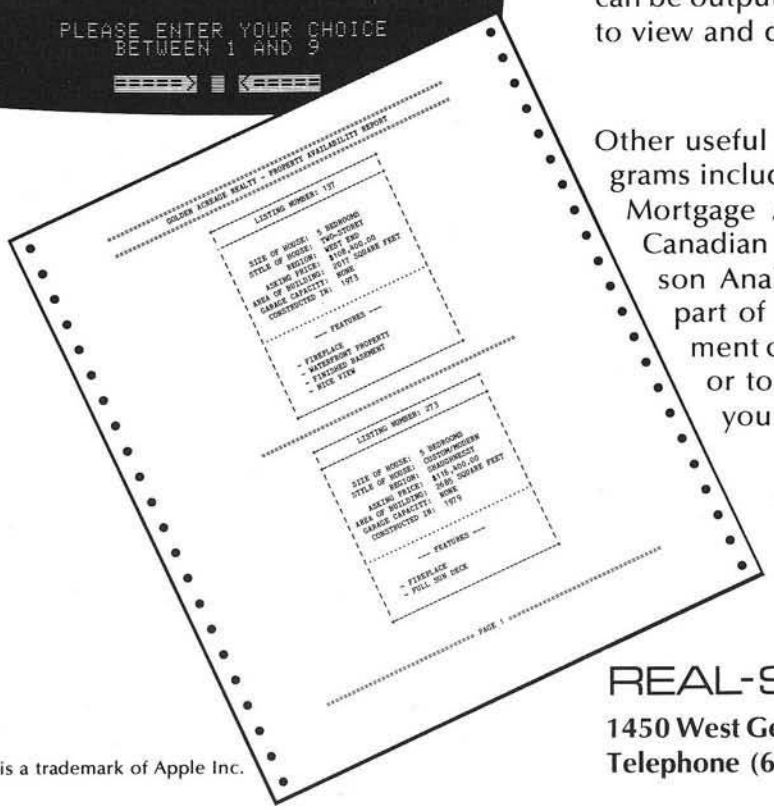
BLOCK SHAPES FOR APPLESOFT OR ASSEMBLY There is no package available today that gives computer customers what they want in the area of graphics. The crying need here, according to our customers, is for a learning package that quits ignoring the one subject that everyone seems to be trying to keep a deep dark secret; assembly & machine language graphics! The BLOCK SHAPES FOR APPLESOFT OR ASSEMBLY package is chock full of programs to create and animate all types of shapes, such as vector shapes, block-shapes, HPLOT-shapes, text file shapes, data array shapes, etc. Included in BLOCK SHAPES FOR APPLESOFT OR ASSEMBLY are shape examining, shape editing, shape drawing, music tone routines, violin sounds, noise creation, assembly language sounds, SUPERFONT and font using, and a Y TABLE for either page of hi-res that allows extra speed in machine language programs since you avoid the HPOSN subroutine. Plus colorful routines. You may never need to buy another graphics package again...because you'll finally *have a handle on what it's all about!!!!* BLOCK SHAPES FOR APPLESOFT OR ASSEMBLY available this fall (1981). 4 disks with over 200 pages of documentation. **Tentative price: 125.00**

Apple II 48K Applesoft ROM*
*Apple is a trademark of Apple Computer, Inc.

Three minutes to find that "just right" property.

Search through up to 3600 property listings with Real-Soft programs for Real Estate Agencies.





No need to flip through endless pages of listings to find that "just right" property. Simply have your client specify the price range, number of bedrooms, preferred geographical location, acceptable house styles and up to 12 additional listing features. REAL-SOFT programs will find the right properties quickly and impressively giving you more time to view the selected properties and close the sale!

REAL-SOFT is an integrated set of computer programs designed specifically for real estate agency use. All routines are fully customizable to meet the requirements of virtually any agencies. Up to 10 different house styles, 12 different listing features and 16 different areas. Very easy to learn — included is a complete set of practice data to experiment with. Requires 48K Apple II, 2 disk drives and monitor. Printer is optional.

If printer is used a printout of selected properties can be output in only a few minutes for your client to view and discuss in detail.

Other useful routines in this powerful set of programs include a Property Availability Report, full Mortgage Amortization Tables (American OR Canadian method), and a Mortgage Comparison Analysis Report. Then use still another part of REAL-SOFT to estimate the replacement cost of any listing (using local factors), or to estimate the mortgage amount that your client can qualify for!

Available at your local Apple dealer.

Apple II is a trademark of Apple Inc.

REAL-SOFT **Real estate software**

1450 West Georgia Street, Vancouver, B.C. V6G 2T8
Telephone (604) 669-2262

SOFTWARE FOR OHIO SCIENTIFIC

VIDEO EDITOR

Video Editor is a powerful full screen editor for disk-based OSI systems with the polled keyboard (except C1P). Allows full cursor-control with insertion, deletion and duplication of source for BASIC or OSI's Assembler/Editor. Unlike versions written in BASIC, this machine-code editor is co-resident with BASIC (or the Assembler), autoloading into the highest three pages of RAM upon boot. Video Editor also provides single-keystroke control of sound, screen format, color and background color. Eight-inch or mini disk $14.95. Specify amount of RAM.

SOFT FRONT PANEL

Soft Front Panel is a software single-stepper, slow-stepper and debugger-emulator that permits easy development of 6502 machine code. SFP is a fantastic monitor, simultaneously displaying all registers, flags, the stack and more. Address traps, opcode traps, traps on memory content and on port and stack activity are all supported. This is for disk systems with polled keyboard and color (b&w monitor ok). Uses sound and color capabilities of OSI C2/C4/C8 systems (not for C1P). Eight-inch or mini disk $24.95. Specify amount of RAM. Manual only, $4.95 (May be later credited toward software purchase). Six page brochure available free upon request.

TERMINAL CONTROL PROGRAM

OSI-TCP is a sophisticated Terminal Control Program for editing OS-65D3 files, and for uploading and downloading these files to other computers through the CPU board's serial port on OSI C2, C4 and C8 disk-based systems with polled keyboards. Thirteen editor commands allow full editing of files, including commands for sending any text out the terminal port and saving whatever text comes back. INDUTL utility included for converting between BASIC source and TCP file text. Eight-inch or mini disk $39.95. Manual only, $2.95.

OSI-FORTH 2.0 / FIG-FORTH 1.1

OSI-FORTH 2.0 is a full implementation of the FORTH Interest Group FORTH, for disk-based OSI systems (C1,C2,C3,C4,C8). Running under OS65D3, it includes a resident text editor and 6502 assembler. Over one hundred pages of documentation and a handy reference card are provided. Requires 24K (20K C1P). Eight-inch or mini disk $79.95. Manual only, $9.95. "OSI-FORTH Letters" software support newsletter $4.00/year.

All prices postpaid. Florida residents add 4% tax. Dealer inquiries are invited. Allow 30 days for delivery.

WRITE FOR FREE CATALOG OF SOFTWARE AND HARDWARE FOR OHIO SCIENTIFIC !!

Technical Products Company
P.O. Box 12983 Univ. Station
Gainesville, Florida 32604

Flat Rate

DISK DRIVE OVERHAUL

One Week Turnaround Typical

Complete Service on Floppy Disk Drives.

FLAT RATES

8" Double Sided Drive	$170.00*
8" Single Sided Drive	$150.00*
5¼" M.P.I. Drive	$100.00*

**Broken, Bent, or Damaged Parts Extra.*

You'll Be Notified of

1. The date we received your drive.
2. Any delays and approximate time of completion.
3. Date Drive was shipped from our plant.
4. Repairs performed on your Drive.
5. Parts used (# and description).
6. Any helpful hints for more reliable performance.

90 Day Warranty.
Ship Your Drive Today.

Write or call for further details.

PHONE (417) 485-2501

FESSENDEN COMPUTER SERVICE

116 N. 3RD STREET OZARK, MO 65721

PET & APPLE II USERS

TINY PASCAL

Plus+

GRAPHICS

The TINY Pascal System turns your APPLE II micro into a 16-bit P-machine. You too can learn the language that is slated to become the successor to BASIC. TINY Pascal offers the following:

- **LINE EDITOR to create, modify and maintain source**
- **COMPILER to produce P-code, the assembly langauge of the P-machine**
- **INTERPRETER to execute the compiled P-code (has TRACE)**
- **Structured programmed constructs: CASE-OF-ELSE, WHILE-DO, IF-THEN-ELSE, REPEAT-UNTIL, FOR-TO/DOWNTO-DO, BEGIN-END, MEM, CONST, VAR ARRAY**

Our new TINY Pascal PLUS+ provides graphics and other builtin functions: GRAPHICS, PLOT, POINT, TEXT, INKEY, ABS AND SQR. The PET version supports double density plotting on 40 column screen giving 80 x 50 plot positions. The APPLE II version supports LORES and for ROM APPLESOFT owners the HIRES graphics plus other features with: COLOR, HGRAPHICS, HCOLOR, HPLOT, PDL and TONE. For those who do not require graphics capabilities, you may still order our original Tiny Pascal package.

TINY Pascal PLUS+ GRAPHICS VERSION-

PET 32K NEW Roms cassette	$55
PET 32K NEW Roms diskette	$50
APPLE II 32K/48K w/DOS 3.2 or 3.3	$50

TINY Pascal NON-GRAPHICS VERSIONS-

PET 16K/32K NEW Roms cassette	$40
PET 16K/32K NEW Roms diskette	$35
APPLE II w/ROM Applesoft 32K w/DOS	$35
APPLE II w/RAM Applesoft 48K w/DOS	$35
USER's Manual (refundable with software order)	$10
6502 Assembly Listing of INTERPRETER-graphics	$25
6502 Assembly Listing of INTERPRETER-non graphics	$20

master charge

VISA

FREE postage in U.S. and CANADA. Orders may be prepaid of by bankcard (include card number and expiration date). Michigan residents include 4% state sales tax. Orders accepted via THE SOURCE - CLO852.

ABACUS SOFTWARE
P.O. Box 7211
Grand Rapids, Michigan 49510
(616) 241-5510

Decision Systems

Decision Systems
P.O. Box 13006
Denton, TX 76203

SOFTWARE FOR THE APPLE II*

ISAM-DS is an integrated set of Applesoft routines that gives indexed file capabilities to your **BASIC** programs. Retrieve by key, partial key or sequentially. Space from deleted records is automatically reused. Capabilities and performance that match products costing twice as much.
$50 Disk, Applesoft.

PBASIC-DS is a sophisticated preprocessor for structured **BASIC**. Use advanced logic constructs such as **IF...ELSE..., CASE, SELECT**, and many more. Develop programs for Integer or Applesoft. Enjoy the power of structured logic at a fraction of the cost of **PASCAL**.
$35. Disk, Applesoft (48K, ROM or Language Card).

DSA—DS is a dis-assembler for 6502 code. Now you can easily dis-assemble any machine language program for the Apple and use the dis-assembled code directly as input to your assembler. Dis-assembles instructions and data. Produces code compatible with the S-C Assembler (version 4.0), Apple's Toolkit assembler and others.
$25 Disk, Applesoft (32K, ROM or Language Card).

FORM-DS is a complete system for the definition of input and output froms. **FORM-DS** supplies the automatic checking of numeric input for acceptable range of values, automatic formatting of numeric output, and many more features.
$25 Disk, Applesoft (32K, ROM or Language Card).

UTIL-DS is a set of routines for use with Applesoft to format numeric output, selectively clear variables (Applesoft's **CLEAR** gets everything), improve error handling, and interface machine language with Applesoft programs. Includes a special load routine for placing machine language routines underneath Applesoft programs.
$25 Disk, Applesoft.

SPEED-DS is a routine to modify the statement linkage in an Applesoft program to speed its execution. Improvements of 5-20% are common. As a bonus, **SPEED-DS** includes machine language routines to speed string handling and reduce the need for garbage clean-up. Author: Lee Meador.
$15 Disk, Applesoft (32K, ROM or Language Card).

(Add $4.00 for Foreign Mail)

***Apple II is a registered trademark of the Apple Computer Co**

MICRO Software Catalog

Mike Rowe
34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

Name: **Enhanced Graphic Software for the Epson MX-80 and MX-100**
System: Apple II or Apple II+
Memory: 48K
Language: 6502 Assembly
Hardware: Disk drive, Epson MX-80 or MX-100 printer
Description: Graphic dump which allows the user to get hard copy graphics of anything that can be loaded on the high-resolution pages of the Apple with one-keystroke commands. Easy to use, versatile, well supported as are all graphic dumps from Computer Station. (The Epson MX-80/MS-100 now added to our line.)
Price: $44.95 includes software, documentation, practice pictures/plots
Author: David K. Hudson
Available: Computer Station
11610 Page Service Dr.
St. Louis, MO 63141
(314) 432-7019

Name: **Number Cruncher Disk**
System: OSI Challenger (C2 and C3 series)
Memory: 48K
Language: BASIC under OS 65D
Hardware: Disk drive, CRT, optional printer
Description: A statistical analysis package, including a data base management system with facilities for convenient handling of data series. Contains commands for producing descriptive statistics, plus exploratory data analysis graphics and regression.
Price: $195.00 for 8" disk and documentation postpaid. $20.00 for manual only. Free flier available on request.
Author: Mike Anderson
Available: Responsive Computer Technology, Inc.
P.O. Box 719
Silver Spring, Maryland 20901

Name: **The Executive Secretary™**
System: Apple II
Memory: 48K
Language: Applesoft in ROM or Language System
Hardware: Apple II, one or two disk drives, lower case adapter or 80-column video board, shift key modification, printer
Description: This is the ultimate word processor for the Apple II computer. Works with 40- or 80-column screens interchangeably, shows lower case, has real shift key. Other features: works at professional typing speeds; versatile page numbering and header printing on each page; file merge and unmerge; block operations — move, transfer, and delete; automatic insertion of full phrases for user-defined abbreviation (unlimited number); automatic envelope address; built-in card file system; interfaces with Data Factory™, On-Line Database, Information Master, and Visicalc™ files; file chaining and nesting; "if" and relational commands to allow conditional printing of information based on the contents of a database; insertion of data directly from database files (in lower case, if desired); permits keyboard input during print time; multi-level outline indentation; right and left justified tab stops; dynamic text reformatting; immediate mode configuration for display screen, number of disk drives, and printer (including Centronics 737 and IBM Selectric); interfaces with CCS clock board for time stamping of documents; embedded or external printer commands; character/word/line insert/replace/delete; selective or global search and replace; built-in interface to D.C. Hayes Micromodem II™; menu-driven operation; easel-bound, indexed manual; lesson-type instructions.
Price: $250.00
Available: Aurora Systems, Inc.
2040 E. Washington Ave.
Madison, WI 53704
(608) 249-5875

Name: **Business Plus**
System: Apple and Atari/800
Memory: 48K
Language: Applesoft & Binary (Apple)
BASIC & Binary (Atari)
Hardware: Any 80-column printer
Description: An all-in-one billing system. Handles invoices, statements, credit memos, purchase orders, payables, writes checks, account aging reports (30, 60, 90 and over 90 days), bar graphs of sales, income and expenses, mailing list with search, sales register and a whole lot more. Everything you need for daily business operations.
Price: $299.00 (or $25.00 for demo - credited towards purchase) includes 2 disks and documentation
Author: Advanced Data Systems
Available: Advanced Data Systems
7468 Maple Avenue
St. Louis, MO 63143

Name: **Micro-Telegram**
System: Apple II or Apple II Plus
Language: Integer BASIC or Applesoft
Hardware: Apple II or Apple II Plus
Description: Allows Apple owners to access Western Union Services worldwide, send and receive TWX™, Telex™ and international cables, and send mailgrams. Apple owners can also access continuously updated reports on news, stock, foreign exchange, gold, futures, sports reports and ski conditions through Infomaster®, the Western Union Data Base.
Price: $250.00 - suggested retail, FOB Boston, includes mini-floppy diskette and documentation
Author: Microcom, Inc.
Available: Microcom, Inc.
89 State St.
Boston, Massachusetts 02110

Name: **Super-Text II**
System: Apple II word processing program
Memory: 48K
Language: Assembly
Hardware: Apple II or Apple II+
Description: With Super-Text the basics of text editing are learned quickly, yet its advanced features will meet the user's expanding word processing requirements into the future. Add the Form Letter Module and Address Book Mailing List for the ultimate in professional or personal use word processing.
Price: $150.00 includes documentation, an unlimited time replacement policy, and dual disk
Author: Ed Zaron
Available: MUSE Software
330 N. Charles
Baltimore, MD 21201
(301) 659-7212

Name: **Eureka™ Learning System**
System: Apple II or Apple II+
Memory: 32K Cassette, 48K Diskette
Language: Applesoft with some machine language
Hardware: Cassette or diskette (DOS 3.3)
Description: An interactive, menu driven program that helps teachers create courseware. No programming is necessary. Any subject may be taught using symbols and line drawings (Hi-Res shapes). Courses are presented to students in three modes, with optional sound effects. May be used with any level of student, pre-school through adult. The material and its style is up to the teacher, not the computer.
Price: $995.00 for software license
Author: Eiconics, Inc.
Available: Eiconics, Inc.
200 Cruz Alta
Taos, New Mexico 87571

Name: **Sentence Diagramming**
System: Apple II
Memory: 48K
Language: Applesoft
Hardware: Apple II, Disk II (one or two drives)
Description: Teaches sentence diagramming, parts of speech, and usage, for individual student sessions. It is also an excellent tool for teachers to use for instruction in one, two, or all of these areas. Students may use the teacher-formatted disk for individual practice at 3 levels of difficulty. Each level has 20 separate sentences. Options include creating student record files, monitoring progress, reading records, omitting diagramming sections, etc. This is the best grammar disk available today. Grades 6-12.
Copies: Many
Price: $19.95 includes disk, manual, demo sheet
Available: Avant-Garde Creations
P.O. Box 30161 MCC
Eugene, OR 97403

Name: **Extended SYM-BASIC**
System: SYM-1
Memory: 16K
Language: 5½K machine language program
Hardware: Serial terminal and Synertek BASIC ROMs
Description: Extended SYM-BASIC adds over 30 new commands/functions to standard SYM-BASIC. Features include: a unique input line editor; pagination of program listings; hex arithmetic and arguments; built-in printer control; auto line number prompting; realtime clock; powerful trace/debug command; trigonometric patch; ultra renumber; powerful execute command; range delete command; and many others. List of commands follows: $, @HH, @MM, @SS, APPEND, AUTO, CA, CALL, CR, CHAIN, DEL, DR, EDIT, EXEC, GET, GOTO, IN=, LOADP, LOAD NUM, OUT=, PAGE, PRINTOFF, PRINTON, PRINTUSING, SAVEP, SAVEV, SAVEB, STIME, TRACE, VERIFY.
Copies: 50 copies (Note: over 200 copies of 8K version sold.)
Price: $85.00 U.S., $95.00 Canada, includes object on cassette and 90-page instruction manual complete with source listing
Author: John W. Brown
Available: Saturn Software Limited
8146 116A St.
Delta, B.C., V4C 5Y9,
Canada

Name: **A.3. Frequency Analysis**
System: PET
Memory: 8K
Language: BASIC
Hardware: PET/CBM
Description: Using harmonic analysis techniques, a frequency scan is made of a time series, such as stock prices, which discloses frequencies of significant amplitudes. A harmonic analysis is then made at chosen frequencies. Included is a logical file input and modification to update and delete old data.
Price: $15.00 for cassette and documentation
Author: Claud E. Cleeton
Available: 122-109th Ave., S.E.
Bellevue, WA 98004

Name: **HSD Anova**
System: Apple II or Apple II Plus, DOS 3.2
Memory: 48K
Language: Applesoft
Hardware: Optional printer with serial or parallel interface
Description: HSD Anova is a powerful, flexible analysis of variance program suitable for scientific research and business analyses. This single program analyzes balanced designs of from one to eight independent variables. It can handle designs composed of between-subjects and/or within-subjects factors. Design specification and data entry are simple. Data entry is from keyboard or disk, with data editing. Output is an Anova table on CRT or printer.
Price: $74.95 includes disk, complete documentation, binder.
Author: Stephen Madigan, Ph.D.
Virginia Lawrence, Ph.D.
Available: Human Systems Dynamics
9249 Reseda Boulevard
Suite 107C
Northridge, California 91324

Name: **FORTH-79 Standard**
System: Apple II, Apple II+
Memory: 48K
Language: Machine Language and FORTH-79
Hardware: 1-14 disk drives (13 or 16 sector-compatible)
Description: FORTH-79 is a structured language suited for systems and applications programming with advantages where execution speed is important (i.e., data acquisition, process control, animation, and video games). Programs run faster than BASIC and are compact. The 32-bit integer arithmetic vocabulary is ideal for business applications and is also extensible. Package includes screen editor, macro-assembler and vocabularies for strings, double precision integers and Lo-Res graphics.
Price: $89.95 includes software and manual (including FORTH-79 and Fig-FORTH)
Author: Martin Tracy and Philip Wasson
Available: MicroMotion
12077 Wilshire Blvd. #506
Los Angeles, CA 90025
(213) 821-4340

Name: **S-FORTH**
System: OSI disk systems
Memory: 20-96K
Hardware: No extra hardware required
Description: S-FORTH is a full implementation of fig-FORTH including editor, a virtual memory disk subsystem, and compatibility with OS65D. All OS65D commands are still usable. You can exit from S-FORTH to OS65D and then return to S-FORTH. It is over 10 times faster than BASIC and is as fast as Pascal. The FORTH compiler uses less memory than Pascal and allows any user with at least 20K to have an excellent FORTH system.
Price: $34.95 for 5¼" or 8" disk (disk and source listing together are $49.95)
Author: Digital Systems
Available: Aurora Software Associates,
P.O. Box 99553
Cleveland, Ohio 44199

Name: **Dental Insurance Form Writer**
System: Apple II with firmware card or Apple II Plus
Memory: 48K RAM
Language: Applesoft, DOS 3.2, 3.3
Hardware: Disk drive, 80-column printer
Description: You can prepare Universal American Dental Association Insurance Claim forms on your Apple. Each form can be prepared, saved to disk, reloaded, edited and printed as many times as you desire. Dental Insurance Form Writer allows rapid billing and claim submittal with a minimum of effort. A master form can be created for each family/patient and saved for later use. This master can be loaded, treatments entered, printed and sent as a pre-authorization or actual statement. Over 100 families per diskette.
Price: $100 includes manual
Author: J. McFarland
Available: Andent Inc.
1000 North Ave.
Waukegan, Illinois 60085

Name: **Pulsar II**
System: Apple II or Apple II Plus
Memory: 48K
Language: Machine
Hardware: One disk drive, 13 or 16 sector controller card
Description: Two games — Pulsar II and Wormwall in a unique combination. Each game has eight levels of play and score can be transferred between the two. The object of Pulsar II is to destroy the spinning shields around the Pulsar and destroy it. Wormwall places you in one of the strangest mazes ever created. The walls do not connect and openings occur only temporarily as colored lines cross. Little creatures chase you in each part of the maze.
Price: $29.95 includes disk and documentation.
Author: NASIR — Presented by Sirius Software, Inc.
Available: Your local Apple dealer or software store.

Name: **The Dragon's Eye**
System: PET or Apple
Memory: 32K (PET) and 48K (Apple)
Language: PET BASIC, Applesoft BASIC
Hardware: PET, Apple II
Description: An overland fantasy game, where the player has 21 game days (approximately a half an hour playing time), to find the Dragon's Eye, a magical jewel hidden by an evil magician. The player chooses one of 16 characters, and gains a set of magical abilities. He also chooses among 13 commands. When he combats the vicious monsters, animated graphics display the action between player and beast.
Price: $24.95
Author: Automated Simulations, Inc.
Available: Automated Simulations, Inc.
P.O. Box 4247
Mountain View,
California 94040

Name: **Sneakers**
System: Apple II or Apple II+
Memory: 48K
Language: Machine
Hardware: Disk Drive
Description: Sneakers are little guys who appear to be friendly but will quickly stomp you out if you do not get them first. After the Sneakers come wave after wave of Cyclops, Saucers, Fangs, H-Wings, Meteors, Scrambles and Scrubs. The variety is incredible and the challenge unending. *Sneakers* is playable with keyboard or paddles.
Price: $29.95 includes disk, documentation and a T-shirt transfer
Author: Mark Turmell
Available: Your local computer software store

Name: **Olympic Decathlon**
System: CP/M or TRS-80
Memory: 48K for Apple II or II+; 32K for disk drive TRS-80 Model 1; 16K for cassette TRS-80 Model 1.
Language: Machine
Hardware: Apple II or II+; TRS-80 Model 1, disk or cassette
Description: Enjoy the excitement and skill of Olympic competition with this game which takes you through all 10 events of the real Decathlon. Each event is presented with animated graphics that put you into the action. Eight and six players can compete respectively with the TRS-80 and Apple versions. Repeat feature lets you practice any event as many times as you wish prior to beginning the actual competition. Your best times can be compared to actual recorded Olympic Decathlon times.
Price: $29.95 includes instruction manual and 5¼" disk (cassette for TRS-80 cassette version)
Author: Tim Smith
Available: Microsoft Consumer Products
400-108th Ave. NE
Suite 200
Bellevue, WA 98004

Name: **Cavern of the Dwarves**
System: SYM with BAS-1 or KIM 8K BASIC at 2000 H
Memory: 16K
Language: BASIC
Hardware: Terminal using standard serial I/O ports on SYM or KIM
Description: An adventure game in which you wander a large cavern seeking treasure, fighting monsters, and trying to avoid getting killed by the many dwarves who inhabit the cavern. You communicate with the computer using one- and two-word commands.
Price: $10.00 on cassette tape, ppd. in U.S. only
Author: Lee Chapel
Available: Lee Associates
2349 Wiggins Ave.
Springfield, IL 62704

Hot pursuit through space and the vortices of time!

RAINWARE PRESENTS. . .

Time Lord

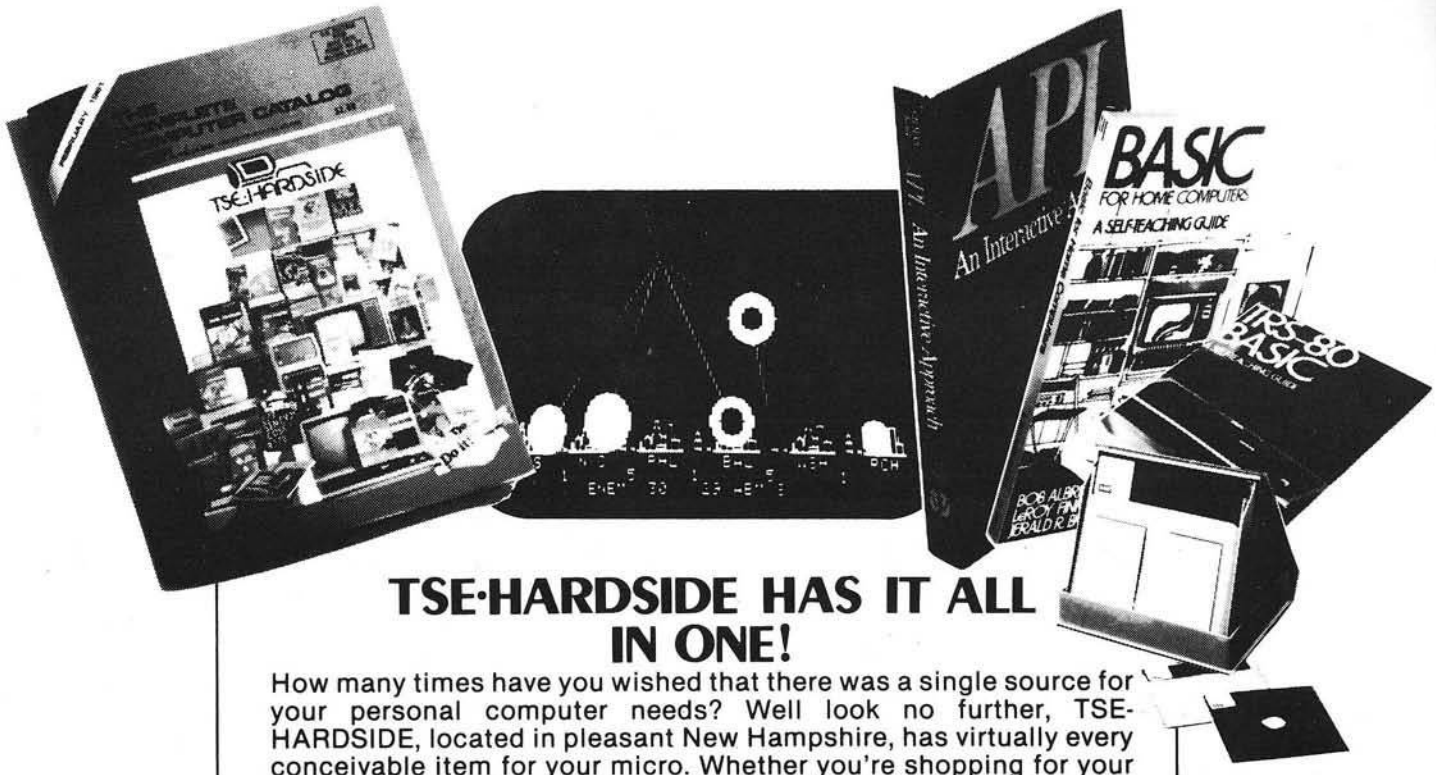
The fallen Time Lord, who presumptuously calls himself The Master, is at large. The elders of Waldrom have supplied you with the hyperspace-worthy vessel Tardus, and commissioned you to eliminate the evil "Master". Your resources include clones who will fight for you, the formidable CRASER weapons of the Tardus, and magic weapons such as Fusion Grenades and Borelian Matrix Crystals.

Traveling through hyperspace in search of the evil one, you will encounter Time Eaters, Neutron Storms, and other alien creatures and phenomena. Entering real space to search planets, you will encounter still other dangers. You will enter native settlements to buy food and supplies — or to fight for survival.

And once you find The Master can you destroy him?

Based on Dr. Who of PBS fame.
Apple Integer Basic,
Disk, 48K . . . $29.95

TSE-I-HARDSIDE
6 South St., Milford, NH 03055 (603) 673-5144
TOLL FREE OUT-OF-STATE 1-800-258-1790



TSE·HARDSIDE HAS IT ALL IN ONE!

How many times have you wished that there was a single source for your personal computer needs? Well look no further, TSE-HARDSIDE, located in pleasant New Hampshire, has virtually every conceivable item for your micro. Whether you're shopping for your Apple, Pet, TRS-80™ or Atari, TSE-HARDSIDE has it all. We stock hardware, software, books, magazines and specialty items for all of the popular machines. So the next time you're out shopping for your system don't be surprised, be satisfied. Remember the name TSE-HARDSIDE as your choice for quality, service and reliability.

TSE·HARDSIDE
6 South St., Milford, NH 03055 (603) 673-5144
TOLL FREE OUT-OF-STATE 1-800-258-1790

Save $2.

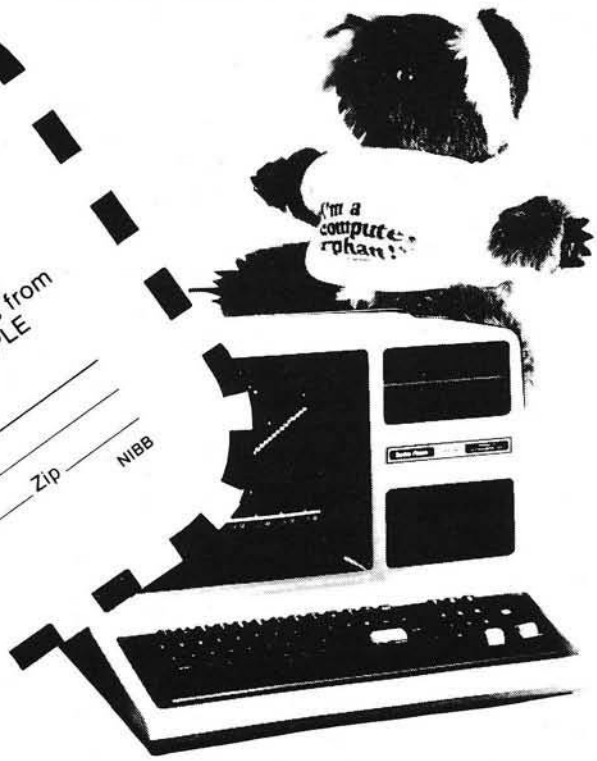
If you or a friend would like to receive our TSE/HARDSIDE Complete Computer Catalog, simply send $1.00 and receive a $2.00 certificate good toward your first purchase of software, hardware, books or specialty items from TSE/HARDSIDE. I have a (check one) ☐ TRS-80, ☐ APPLE ☐ ATARI ☐ PET. A photocopy of this coupon will be accepted

Name _____

Address _____

City _____ State _____ Zip _____ NIBB

MAIL TO: TSE/HARDSIDE
6 South Street
Milford, NH 03055



MICRO

Dr. William R. Dial
438 Roslyn Avenue
Akron, Ohio 44320

6502 Bibliography: Part XXXVI

1045. Interface Age 6, Issue 2 (February, 1981)

Lane, Eric T., "Microcomputing—At the Speed of Light," pg. 75-76, 136.
Graphics show what an object looks like at the speed of light, for the Apple.

Stotts, Gary A., "Amortization Schedule," pg. 90-91.
The Apple program to show you where your money goes when you are paying off a loan.

Schlarb, Keith N., "Information Source for Home and School," pg. 94-95, 138-140.
A random access file program for the Apple.

1046. Apple Cookbook 1, Issues 5/6 (February, 1981)

Maly, Frank, "Proper Program Design," pg. 1-2.
A tutorial on writing proper programs with the Apple.

Golding, Val J., "Applesoft from Bottom to Top," pg. 3-6.
A roadmap to Applesoft to help you understand programming.

Reynolds, Lee, "EXEC Files on the Apple II," pg. 6-7.
A short instructional article on the EXEC command.

Rivers, Jerry, "Sorting," pg. 7-10.
A discussion, with examples, of the various types of sorting routines.

Simpson, Rick, "Introduction to Assembly Language," pg. 10-13.
A tutorial on assembly and machine language for the Apple.

Tyro, A., "Pascal: Beginners Notes," pg. 13-14.
A Tab Demo program in Pascal.

Anon., "Apple Doodle," pg. 15.
An assortment of short routines and procedures for the Apple.

1047. 73 Magazine No. 245 (February, 1981)

Erdei, Steven G., "Under Software Control," pg. 94-98.
A repeater control system with minimal hardware, using a KIM-1.

1048. Byte 6, No. 2 (February, 1981)

Zimmermann, Mark, "A Beginner's Guide to Spectral Analysis," pg. 68-90.
An instructional article including listings for the PET.

Woteki, Thomas H., "A Pascal Library Unit for the Micromodem II," pg. 106-136.
Programs for the Micromodem on an Apple/Pascal system.

1049. L.A.U.G.H.S. 3, No. 1 (January, 1981)

Connelly, Pat, "A Disassembler for S-C Assembler," pg. 3-8.
A disassembler which dissembles into source code format is a definite asset to augment your S-C Assembler, for Apple.

1050. Southeastern Software Newsletter Issue 24 (February, 1981)

George McClelland, "Software Reviews," pg. 1-6.
A review of new games for the Apple.

1051. Dr. Dobb's Journal 6, No. 52, Issue 2 (February, 1981)

Caulkins, Dave and Harris, David C., "PAN — One Activity of the PCNET Project," pg. 17, 37.
Discussion and updates for PAN, a communication net implemented on the PET.

Gordon, H.T., "Byte-Count Routine," pg. 37.
Modifications to CNTBYT and BYTNUM routines for 6502 micros.

1052. The Transactor 2, No. 12 (ca. February, 1981)

Anon., "Exclusive OR on Your PET," pg. 2.
Add this useful function to the PET.

Anon., "Bits and Pieces," pg. 2-5.
Miscellaneous PET notes including discussion of logical operators; differences among BASIC 1.0, 2.0, and 4.0; screen loading; NEC Spinwriter; Card Print Utility, etc.

Hooks, Dave, "Card Print Utility," pg. 6-7.
Listing and cross references for the PET.

Hoogstraat, J., "PET BASIC Label Support Interface," pg. 8-13.
An interesting PET routine residing in the second cassette buffer allowing the use of labels in BASIC. For BASIC 2.0.

Anon, "BASIC 4.0, DOS 2.0 and the Relative Record System," pg. 14-21.
The new PET operating systems, discussion and tutorial with examples.

Higginbottom, Paul, "BASIC 2.0 to BASIC 4.0 Conversions (40 Column)," pg. 22-31.
All about converting several types of PET BASIC programs. With memory map, entry points, 6502 op codes, status variables, etc.

Troup, Henry, "The PET NMI Vector," pg. 32-33.
An instructional article on the non-maskable interrupt.

Butterfield, Jim, "A Few Entry Points, 1.0/2.0/4.0 ROM," pg. 34-35.
A useful tabulation for PET users.

Troup, Henry, "Fun with WAIT Statements," pg. 36-37.
All about PET Wait command.

Anon., "8032 Control Characters," pg. 38-40.
Discussion and tabular summary of control functions.

1053. T.A.R.T. 2, Issue 1 (February, 1981)

Smith, Eric, "String Art," pg. 3-5.
A graphics program for the Apple.

Sander-Cederlof, Bob and Koerin, Sid, "Hi-Res Crest Design," pg. 6-8.
A graphics program for the Apple.

Shipley, Jim, "Revision 7 and 7a Motherboards," pg. 12-13.
A description of the latest Apple motherboards and instructions for hardware modifications.

1054. MICRO No. 33 (February, 1981)

Guest, Ronald A., "A Simple Securities Manager for the Apple," pg. 7-13.
An Apple program to manage your stocks.

Elm, Robert L. ,"Why WAIT?", pg. 15-16.
Interesting applications of the WAIT function on the OSI and PET systems.

Colsher, William L., "An Atari Assembler," pg. 17-19.
A simple one-pass assembler for the Atari.

Cheng, Thomas, "Turning USR(X) Routines into BASIC DATA Statements," pg. 21-22.
A program for the OSI C1P to save machine language routines as BASIC DATA statements.

Wells, George, "Improved Dual Tape Drive for SYM BASIC," pg. 23-28.
Utility routines for SYM to enhance the use of two cassettes, including a tape duplication feature.

Bongers, C., "In the Heart of Applesoft," pg. 31-47.
A tutorial on how to work with Applesoft.

Kollar, Larry, "One Dimensional Life on the AIM 65," pg. 50-52.
A Life game taking advantage of the AIM's 20-character display.

Tenny, Ralph, "Increase KIM-1 Versatility at Low Cost," pg. 57-59.
A hardware article for the KIM involving moving the primary address decoder off-board, making it possible to add other I/O devices.

Strasma, James, "PET String Flip," pg. 65-66.
A solution to the problem of upper and lower case inversion using CBM 2022 and 2023 printers with Old ROM PETs.

Wright, Loren, "PET Vet," pg. 68.
A modified routine to allow you to recover from a crash without losing memory; how to avoid accidental INPUT exit, etc.

Ell, David A., "A C1P Sound Idea," pg. 71-72.
A hardware addition creating a belltone for the C1P or Superboard II.

Sebra, Randy, "Does Anybody Really Know What Time It Is?", pg. 75-79.
Hardware and software for using the OKI Semiconductor MSM5832 CMOS clock chip on your 6502 system, with a BASIC listing for the SYM.

1055. Atari Computer Enthusiasts 2, Issue 2 (February, 1981)

Smith, Wynn, "More on BASIC."
How to get more speed from your BASIC programs on the Atari.

Crawford, Chris, "Missile-Graphics Demo."
A BASIC listing for an Atari graphics routine.

Anon., "POKE Text Into Graphics 8!", pg. 7.
How to POKE alphanumeric graphics characters into the Atari graphics 8 mode.

1056. The Apple Peel 3, No. 2 (February, 1981)

Jenkins, Jerry, "Apple Program Conventions," pg. 6.
A routine to add credits, etc. to program listings for Apple software donated to a club library.

Brown, Thomas A., "Telephone Dialer," pg. 9-10.
Hardware and software listing for an Apple telephone dialer.

1057. Creative Computing 7, No. 2 (February, 1981)

Piele, Donald T., "How to Solve It — With the Computer," pg. 82-92.
A group of programs including Applesoft conversions.

Chapel, Lee, "Monster Combat," pg. 106-116.
A game written in BASIC for the KIM.

Stith, John E., "Lower-Case Display for Apple Writer," pg. 124-129.

(Continued on next page)

Send for FREE Control Page
Also Available soon on Atari

EDIT 6502 T.M. LJK

Two Pass Assembler, Disassembler, and Editor Single Load Program

DOS 3.3., 40/80 Columns, for Apple II or Apple II Plus*

A MUST FOR THE MACHINE LANGUAGE PROGRAMMER. Edit 6502* is a two pass Assembler, Disassembler and text editor for the Apple computer. It is a single load program that only occupies 7K of memory. You can move freely between assembling and disassembling. Editing is both character and line orientated, the two pass disassemblies create editable source files. The program is so written so as to encompass combined disassemblies of 6502 Code, ASCII text, hex data and Sweet 16 code. Edit 6502 makes the user feel he has never left the environment of basic. It encompasses a large number of pseudo opcodes, allows linked assemblies, software stacking (single and multiple page) and complete control of printer (paganation and tab setting). User is free to move source, object and symbol table anywhere in memory. Requirements: 48K of RAM, and ONE DISK DRIVE. Optional use of 80 column M&R board, or lower case available with Paymar Lower Case Generator.

TAKE A LOOK AT JUST SOME OF THE EDITING COMMAND FEATURES. Insert at line # n Delete a character Insert a character Delete a line # nl, n2 to line # n3 Change line # nl to n2 "string1" Search line # nl to n2 "string1".

LOOK AT THESE KEY BOARD FUNCTIONS: Copy to the end of line and exit: Go to the beginning of the line: abort operation: delete a character at cursor location: go to end of line: find character after cursor location: non destructive backspace: insert a character at cursor location: shift lock: shift release: forward copy: delete line number: prefix special print characters. Complete cursor control: home and clear, right, left down up. Scroll a line at a time. **Never type a line number again.**

All this and much much more — Send for FREE Information.

Introductory Price $50.00.

LJK™ COMPUTER BASED SOFTWARE ENTERPRISES Inc

LJK Enterprises Inc. P.O. Box 10827 St. Louis, MO 63129 (314)846-6124

*Edit 6502 T.M. of LJK Ent. Inc.. — *Apple T.M. of Apple Computer Inc.

A modification for the Applewriter software.

Yob, Gregory, "Personal Electronic Transactions," pg. 154-160.
Discussion of the PET Toolkit, BASIC 4.0, and PILOT language.

Blank, George, "Outpost: Atari," pg. 168-171.
Discussion of Atari graphics, listing for Decimal Dump, the PEEK function, etc.

Carpenter, Chuck, "Apple-Cart," pg. 172-180.
Discussion of programming languages for the Apple, including Mini PILOT; Mini-PILOT Interpreter, etc.

1058. KB Microcomputing No. 50 (February, 1981)

Baker, Robert W., "Tape-To-Disk Data File Copy," pg. 15.
A basic program for the PET.

Chamberlin, Hal, "Simulation of Musical Instruments," pg. 142-148.
Part 2 of this series on music synthesis includes routines written for the KIM.

Kammer, David W., "Autoloader Program for the C1P and Superboard II," pg. 158-160.
Routines to load and save machine language programs on OSI systems.

1059. Personal Computing 5, No. 2 (February, 1981)

Miles, Kenneth, "Menu-Writer," pg. 38-42.
Let Apple write a menu for your Applesoft or Integer BASIC programs.

Karis, Robert, "Archimedes' Spiral," pg. 81-82.
A versatile graphics Hi-Res program for the Apple.

1060. CSRA Computer Club Newsletter (February, 1981)

Morse, Ken, "Permanent (?) Comment," pg. 1.
A program to convert Integer BASIC line numbers to 65535 to make them more resistant to undesired deletion.

Sikes, Randy, "DOS 3.3 Update," pg. 4.
A simple fix for the new Apple DOS 3.3.

1061. Softalk 1, No. 6 (February, 1981)

Wagner, Roger, "Assembly Lines," pg. 26-29.
Part 5 of the guide to assembly language covers branch instructions.

1062. O.S.I. Users Independent Newsletter No. 8 (February, 1981)

Curley, Charles, "Implementing the NEC Spinwriter," pg. 1-8.
Some notes for OSI owners who wish to use the Spinwriter.

1063. Poke Apple 3, No. 1 (February, 1981)

Averill, Bonnie Kaufman, "Elementary Programming: A Basic Budget," pg. 11-13.
The first of a series of tutorial articles showing how to develop a budgeting program.

Neff, Thomas M., "Apple Notes," pg. 14-15.
Discussion of the use of Apple's Editor/Assembler, found in the DOS Toolkit. Also Neil Konzen's Program Line Editor and the INIT command.

Garvey, Michael, "Storing and Reading an Array on Disk," pg. 16-17.
A utility for the Apple.

Greene, Amos, "Sex and the Single Drive (Disk Version)," pg. 18-19.
An instructional article on the storage of data on the Apple Disk.

1064. Apple-Dayton 2, No. 2 (February. 1981)

Anon., "Program List Formatter," pg. 2.
An EXEC File Operator for the Program List Formatter in the Winter Orchard magazine.

1065. The Michigan Apple-Gram 3, No. 2 (February, 1981)

Tuttleman, Roger, "Languages, Languages," pg. 1-4.
An Apple/Pascal program to plot circles, etc. on the Hi-Res screen.

Tuttleman, Roger, "Getting Started in Pascal," pg. 6-7.
An instructional article with a BASIC Booter Routine in Apple/Pascal.

Rivers, Jerry, "Principally Pascal," pg. 11-14.
PRINTIT, a program for Apple/Pascal designed to send all or part of a .TEXT file to either the console or to your printer.

Deegan, W. Curt, "Hiding Amidst the DOS with PUMA," pg. 16-18.
Protected user memory area for your Apple.

Anon., "IAC Apnote: The Apple II Cassette Interface," pg. 19-21.
A description of the structure and operation of the cassette interface.

Anon., "IAC Apnote: The Preliminary Apple Pascal Guide to Interface Foreign Hardware," pg. 26-41.
A detailed guide to interfacing in Apple Pascal systems.

Sokal, Dan, "Pascal—PEEKs and POKEs," pg. 42-43.
A program for the Apple/Pascal library.

Anon., "IAC Apnote: Text Screen Mapping and Use," pg. 43-44.
A good source of information on how to POKE characters on the Hi-Res screen.

1066. The Seed 3, No. 2 (February, 1981)

Anon., "Apple PI Conventions," pg. 4.
A program to enter standard program labels into software items donated to club libraries, for the Apple.

Breyfogle, Louis D. and Quinn, Jack D., "The 13/16 Sector Problem: A Solution," pg. 8-9.

Stadfeld, Paul, "Space Exploration," pg. 11.
An instructional article on using Applesoft's SPC function.

1067. Compute! 3, No. 2, Issue 9 (February, 1981)

Lee, Arnie, "LED — A Line-Oriented Text Editor," pg. 16-20.
A utility for the PET to maintain PASCAL source language statements.

Baker, Robert W., "The Atari 825 Printer," pg. 24-28.
Description and evaluation of a printer for the Atari systems.

Butterfield, Jim, "Simulated PRINT USING," pg. 30-32.
A program for 6502 micros.

Albrecht, Bob and Firedrake, George, "The Mysterious and Unpredictable RND," pg. 34-40.

Part 2 of several articles on the RND function, PET oriented.

Wachtel, A., "Stat Lab," pg. 42.
A statistical program for 6502 systems.

DeJong, Marvin L., "A BCD to Floating-Point Binary Routine," pg. 46-52.

A routine for the AIM 65 micro.

Lowell, J.R., "BASIC Math for Fun and Profit," pg. 54-59.
An 8K elementary arithmetic program written for the 16K PET with new ROMs.

Esbensen, Tory, "PET Spelling Lessons Your Students Can Prepare," pg. 60-62.
A program for the PET.

Falkner, Keith, "List Apple Integer BASIC Programs One Page At A Time," pg. 64-66.
A machine language utility for the Apple to assist in listing Integer programs.

Gat, Erann, "The 25¢ Apple II Real Time Clock," pg. 68-73.
An article on inexpensive hardware and software for an accurate clock for the Apple.

Martell, Eric and Murdock, Chris, "Ticker Tape Atari Messages," pg. 74.
A horizontal scrolling message routine for the Atari.

Schreibman, Arthur, "Atari Colors and Sounds with Paddles," pg. 75.
A short program for the Atari.

Veludo, Henrique, "Atari Terminal," pg. 75.
A short communications program allowing contact over the telephone with a remote computer system.

Braannon, Charles, "Character Generation on the Atari," pg. 76-78.
A tutorial on defining the character set of the Atari.

Kingston, C., "Put a Printer on the Atari Ports," pg. 82-85.
Drive a printer through the joystick ports.

Boden, Gary, "Double-Density Graphing On the OSI C1P," pg. 86-87.
A way to effectively increase the normal 24 × 24 format of the C1P to a 40 × 40 format for graphing functions.

Berger, Tom R., "A Small Operating System: OS65D — The Kernel," pg. 88-94.
Part 2 of 3 with subroutine descriptions for OSI systems.

Reid, Neal E., "Contour Plotting," pg. 97-102.
How to produce graphs of functions of two variables using the PET and a 2023 friction feed printer.

Young, R.D., "Relocate," pg. 103.
Relocating or loading programs to portions of memory other than from the normal beginning of memory.

Butterfield, Jim, "Mixing and Matching Commodore Disk Systems," pg. 104-108.
A discussion and notes on PET/CBM disk systems.

Spencer, Peter, "Memory Calendar," pg. 109-113.
A program for the PET.

Deal, Elizabeth, "Crash Prevention for the PET," pg. 114-116.
Several reasons for crashes and how to avoid them.

Butterfield, Jim, "Odds and Ends," pg. 118-119.
Notes on PET/CBM files.

Garst, John F., "Three PET Tricks," pg. 120.
On-line REMarks; flashing cursor for GET; and pretty printing.

Bruey, A.J., "Pascal on the PET," pg. 124-125.
A discussion of this additional language for the PET.

Land, Bruce, "A Terminal for 'KAOS' (KIM, AIM, OSI, SYM)," pg. 128-133.
Hardware for small single board micros to make a simple communications terminal.

MacKay, A.M., "SYMple Clock," pg. 134-137.
A clock program that is a little different — for the 4K SYM-1.

Chamberlin, Hal, "Expanding KIM-Style 6502 Single Board Computers, pg. 138-139.
How many expansion boards can the unbuffered microprocessor bus drive before becoming overloaded?

Wells, George, "Load and Save KIM BASIC Programs on Your SYM," pg. 140-142.
Can cousins marry and remain happy?

1068. Apple-Com-Post Issue 9 (ca. June, 1980)

Knuelle, Alfred, "Paddles, Joysticks undsoweiter," pg. 8.
How to use a 40K pot in place of a difficult-to-find 150K pot in constructing paddles or joysticks for the Apple.

Kniefel, J. and Goetze, Uwe, "Programmschutz," pg. 11-12.
How to add copyright statements to your basic pro-

(Continued on next page)

The only thing you can do with a baked Apple is eat it.

The more you stuff your Apple II™ with plug-in boards, the more of a chance it has to overheat.

And once that happens, it won't do anybody any good. Your program bombs and you start losing time and money.

The solution? Simple. Take two minutes to install the Dana Industries fan in the back of your Apple, and you'll practically never have to worry about overheating again.

So pick up the Dana Industries fan at your local computer store. And your Apple will have a long and fruitful life.

™Apple II is a trademark of Apple Computer, Inc.

grams and make them more resistant to obliteration or alteration.

1069. From The Core (February, 1981)

Whittaker, Alec, "Lazer Lower Case Plus," pg. 4.
A review of new hardware for the Apple. Also some routines for text files.

Schroyer, Jeff, "Lazer Lowercase in Depth," pg. 5.
An evaluation of this new hardware mod in some detail.

Anon., "Dr. Apple: Some Uses for a Disassembler," pg. 5-7.
Use the disassembler to assist in Pascal/Apple programs.

Lingwood, David A., "Overlaying in Applesoft," pg. 6-7.
How to get a 20K program to run in 8K.

Budge, Joe, "The Locksmith," pg. 3.
Some notes on the use, ethics, etc. of this utility in 'unlocking' protected software disks.

1070. Mini'app'les 4, No. 2 (February, 1981)

Pinotti, Terry L., "DOS 3.2 and 3.3 on Single Controller," pg. 2-3.
Hardware modification to your Apple Disk II controller card to switch back and forth between the 13/16 sectors.

Hammond, Daryl, "Pondering Pascal: Run Time Errors," pg. 11-15.
A tutorial on Apple/Pascal and an example of how to debug Pascal programs.

Pinotti, Terry L., "Game Paddle Port Modification," pg. 15.
A simple hardware mod to make installing accessories on the Apple I/O ports.

1071. Spreadsheet (Visigroup — A Visicalc User Group) 1, No. 1 (November, 1980)

Staff, "Definitions," pg. 2.
Definitions to make communications amongst Visicalc/6502 users easier.

Staff, "Template," pg. 3.
Typical layout for a Visicalc sheet, reserving the first two columns for variables.

1072. Spreadsheet 1, No. 2 (January, 1981)

Mellon, Arthur Mellon, "Merging Templates," pg. 2.
A time-saving procedure for setting up sheets.

Anon., "Visilist," pg. 5-6.
An accessory utility for printing out the contents of Visicalc template formulas, valuable in diagnostics and planning improvements.

Ender, Philip, "Visitip #5: Flashing M."
How to avoid that flashing M by using more memory in your Apple.

Staff, "Visitip #6: Template Development Aid," pg. 6.
How to find out where the "to" print position has got to in a template under development.

1073. Printout 2, No. 2 (February/March, 1981)

Batey, Duncan, "Matrix Codes," pg. 11.
Useful table for PET users.

Valentine, Mark, "Tick Tock PET," pg. 11.
A 12-hour clock machine-language routine for PET.

Nuttall, John, "Visicalc: How and Why," pg. 18-19.
All about Visicalc for the PET.

Turnbull, Tommy, "Tommy's Tips," pg. 21, 47.
Automatic deletion of DATA statements; instant algebraic input; etc.

Anon., "Turnkey ROMs: Do they Open the Door?", pg. 22, 31.
How to implement turnkey operation on the PET.

Jarrett, Dennis, "PET Communications — State of the Art Report," pg. 24-29.
A special in-depth presentation on PET communication hardware, systems, etc.

Staff, "Colour Display Arrives for PET," pg. 33-35.
A review of the Chromadaptor for the PET system.

Sanders, Gavin, "Get Back in the Black," pg. 36-37.
How to recover and make reuseable those old printer ribbons.

Yob, Gregory, "Personal Electronic Transactions," pg. 43-45.
How to get lower case on the PET; a program for POKE-ing machine language into the PET memory; and several short machine language routines for the PET.

1074. FWAUG Newsletter 2, No. 2 (December, 1980)

McVay, Ray, "Disk Status Finder," pg. 2-10.
A machine language routine that can be run from either DOS 3.2/3.3 which will tell the current DOS version, Master/slave status, free sectors and auto-boot file name.

Meador, Lee, "Disassembly of DOS 3.2 — Part 10," pg. 12-18.
This section of DOS is nearly identical between 3.2, 3.2.1, and 3.3 — devoted to the DOS command decoder.

1075. Apple Assembly Line 1, Issue 5 (February, 1981)

Sander-Cederlof, Bob, "Apple Noises and Other Sounds," pg. 2-9.
Routines for Apple bell, machine-gun noise, laser swoop, inch-worm sounds, touch-tone simulator, morse code output, etc.

Boering, Brooke W., "Faster 16×16 Multiply," pg. 11-12.
A routine for rapid multiplication of two 16-bit values to get a 32-bit value.

Sander-Cederlof, Bob, "A String Swapper for Applesoft," pg. 14-15.
Rearrange data, sort alphanumerics, all the easy way.

1076. Softside 3, No. 5 (February, 1981)

Pelczarski, Mark, "Developing Data Base: Part 6," pg. 16-17, 82-83.
This month capabilities for print formatting are added to this utility for Apples and Ataris.

Truckenbrod, Joan, "Three Dimensional Rotation — Part III," pg. 22-25.
An Apple Hi-Res graphics program.

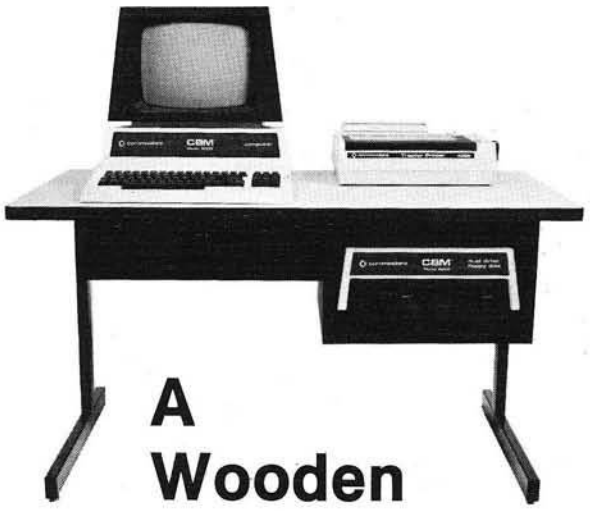
Schoenmeyer, Roger, Thompson, Robert and Mueller, Carl, "Apple One-Liners," pg. 45.
Three oneline programs for the Apple.

Case, Phillip, "Miner," pg. 48-52.
Atari and Apple versions of a program offering adventure deep in an abandoned gold mine.

Cross, Mark, "Famous Sayings Hangman," pg. 59-61.
A different type of Apple Hangman.

Ockers, Stan, "Changing Hearts," pg. 64-65.
An Atari graphics program.

Daoust, John, "Darts," pg. 84.
A paddle game for the Apple.



A Wooden Computer?

Not from Commodore!

So why should the desk look like wood? A pleasant cream and charcoal trimmed desk looks so much better with Commodore systems. One look and you'll see. Interlink desks are right. By design.

The specifications only confirm the obvious:

- Cream and charcoal color beautifully matches the Commodore hardware and blends with your decor.
- An ideal 710 mm (28") keyboard height yet no bumping knees because a clever cutout recesses the computer into the desk-top.
- High pressure laminate on both sides of a solid core for lasting beauty and strength.
- Electrostatically applied baked enamel finish on welded steel legs—no cheap lacquer job here.
- T-molding and rounded corners make a handsome finish on a durable edge that won't chip.
- Knocked down for safe, inexpensive shipment.
- Patented slip joints for quick easy assembly.
- Leveling glides for uneven floors.
- Room enough for a Commodore printer on the desk, yet fits into nearly any den or office niche—H: 660 mm (26") W: 1170 mm (46") D: 660 mm (26").
- Matching printer stand available with slot for bottom feeding.

PRICE: $299

In short, as Commodore dealers, we won't settle for anything that looks good only in the catalog! Our customers won't let us. They don't buy pictures. And neither should you. This is why **we will let you use one of our desks for a week** and then decide. If for any reason you don't like it, just return it in good condition for a cheerful refund.

If your Commodore dealer doesn't carry our desks yet, send a check for $299 and we will ship your desk freight paid!

Name_____

Address_____

City_____ **St**_____ **Zip**_____

Interlink, Inc., Box 134, Berrien Springs, MI 49103
Master Charge and Visa welcome. Call our order line:
616-473-3103

Advertisers' Index

Aardvark Technical Services....86
Abacus Software....96, 100
Advanced Operating Systems....BC
Anaheim Computer....89
Andromeda, Inc....48
Applied Analytics, Inc....42
Avant-Garde Creations....98
Beta Computer Devices....43
Broderbund Software....31
Central Point Software....7
Classified Ads....91
Cleveland Consumer Computers....82
Columbus Instruments....43
CompuTech....92
Computer Case Co....35
Computer Mail Order....34
Computer Station....89
Co-Op Software....61
Connecticut Information Systems, Co....92
Consumer Computers....26
Creative Computing....95
Decision Systems....100
Dtack Grounded....35
D&N Microproducts Inc....90
Dosware Inc....73
Eastern House Software....35
Enclosures Group....90
Exatron....18
Execom Corp....32
Fessenden Computer Service....100
Galfo Systems....96
Gimix, Inc....22
Hogg Laboratories....96
Howard Software....78
Hudson Digital Electronics....36
Huntington Computing....6
Instant Software....24-25
Interlink, Inc....111
Jini Micro Systems....44
Lazer Systems....65
LJK Enterprises....107
Logical Software, Inc....74
Maramoty & Scotto Software Corp....19
Micro Co-Op....40
Micro Distributors....109
MICRO INK, Inc....77
Micro Interfaces....12
Microsoft Consumer Products....IFC
MicroSoftware Systems....67
Micro-Ware Distributing Inc....74
Mittendorf Engineering....14
Modular Systems....13
Mountain Computer....IBC
National Computer Shows....1
Nikrom Technical Products....67
Omega Microware....81
Omega Software Products....47
Pegasys Systems....74
Perry Peripherals....13
Progressive Computing....13
Quality Software....70
Real-Soft....99
Rosen Grandon Associates....67
Sensible Software....57
Serendipity Systems, Inc....96
Sirius Software....2
Skyles Electric Works....17, 31, 46, 91
Small Business Computer Systems....67
Smartware....92
Smoke Signal Broadcasting....23
Softape....56
Stellation Two....47
Sunset Electronics....40
Synergetic Solutions....89
Synergistic Software....81
Synertek Systems....4
Technical Products....100
Terrapin, Inc....7
Travenol Laboratories Inc....90
TSE Hardside....104-105
Unique Data Systems....41
Versa Computing....58

Next Month in MICRO

In addition to the usual, balanced coverage we strive for in MICRO, since June 1981 we have added pages devoted to special topics. We call these added pages BONUS sections.

Commodore Bonus Section

This section will include the following articles:

- **VIC Light Pen-manship —** evaluates several manufacturers' light pens which can be used with the VIC and offers demonstration programs for applying the pens as a scribe or color paint brush.
- **The PET from A to D —** Analog to digital conversion on the PET.
- **Speeding Up ASCII File Retrieval —** Machine language techniques to speed file retrieval in the PET and Apple.
- **Commodore ROM Genealogy —** An up-to-date list of all the combinations of Commodore's ROMs, keyboards, and screens.
- **Character Set Substitution —** Explains how PET characters are generated and discusses several alternate character sets that can be substituted for the largely redundant second character set.

Apple Bonus Section

Apple users will continue to receive extra material in October in the Apple Bonus Section. A sampling of articles follows.

A Booby Trap in Applesoft addresses "protection" schemes for Applesoft. *Random Numbers Generator in Machine Language* offers a simple subroutine to use in a machine language program whenever random numbers are needed. And *Taming the Wild Reset* tells how a user can easily modify an Apple keyboard to ignore normal resets. *Solar Simulation* provides a program for printing information and plotting positions (using hi-res graphics) of the first six planets of the Solar System.

Coming in November

November brings a double BONUS featuring Games and still more on the Apple — just in time for the upcoming, gift-giving holiday month of December.

40% OFF

More MICRO for Less Money When You Subscribe

***Your money goes farther when you* subscribe.** During the course of a year, when you subscribe, you save 40% (in the U.S.).

Pay only $18.00 ($1.50 a copy) for 12 monthly issues of MICRO sent directly to your home or office in the U.S.

But on the newsstand — if you can locate the issue you want — you pay $30.00 a year ($2.50 a copy).

Save 40% **and** make sure you get every issue. Subscribe to MICRO today.

MICRO
34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

Please send me MICRO for __ 1 year __ 2 years
NOTE: Airmail subscriptions accepted for 1 year only.

Check enclosed $_____
Charge my _____ VISA account
_____ Mastercard account

No. _____

Expiration date _____

Name _____

Address _____

_____

City/State _____ Zip ____

Subscription Rates

1 Year*

Country	Air	Surface
United States		$18.00
Canada		21.00
Europe	$36.00	21.00
Mexico, Central America	39.00	21.00
Middle East, N. Africa	42.00	2100
S. America, Central Africa	51.00	21.00
S. Africa, Far East, Australasia	60.00	21.00

* Airmail subscriptions accepted for only 1 year. For U.S. and Canadian 2-year rates, multiply by 2.

Job Title: _____

Type of Business/Industry: _____

Mountain Computer
CPS MultiFunction Card™

The Only Interface Card You Need!

Connecting a Parallel Printer?

Epson® - Centronics® - IDS Paper Tiger® - CPS handles all these printers and others with on-board intelligence to provide paging and other features found on no other card.

Connecting a Serial Printer?

Diablo® - Qume® - NEC - TI 800 Series® - CPS handles these printers and others with standard RS-232 interface providing selection of baud rates, handshakes, paging, and more.

Connecting a Modem or Terminal?

Hayes Smartmodem® - Novation CAT® - M & R Pennywhistle® - CPS handles these and others with full/half duplex operation, baud rate selection, and even a transparent terminal mode which includes a dual mode feature that permits printing of text to parallel printers while 'on-line' eliminating the need for special terminal software—and more.

Connecting with the Time?

The on-board calendar/clock provides real time and date information including day of week, day, month, year, hours, minutes, and seconds for any application requiring a time stamp—battery backed-up for over two years!

Connecting with The Source?

Used with a modem, CPS provides the connection to information utilities, such as The Source,® Dow Jones, and others. Additionally, CPS provides the connection to big-time electronic mail with programs such as Micro Courier® and Micro Telegram,® and other data transfer programs.

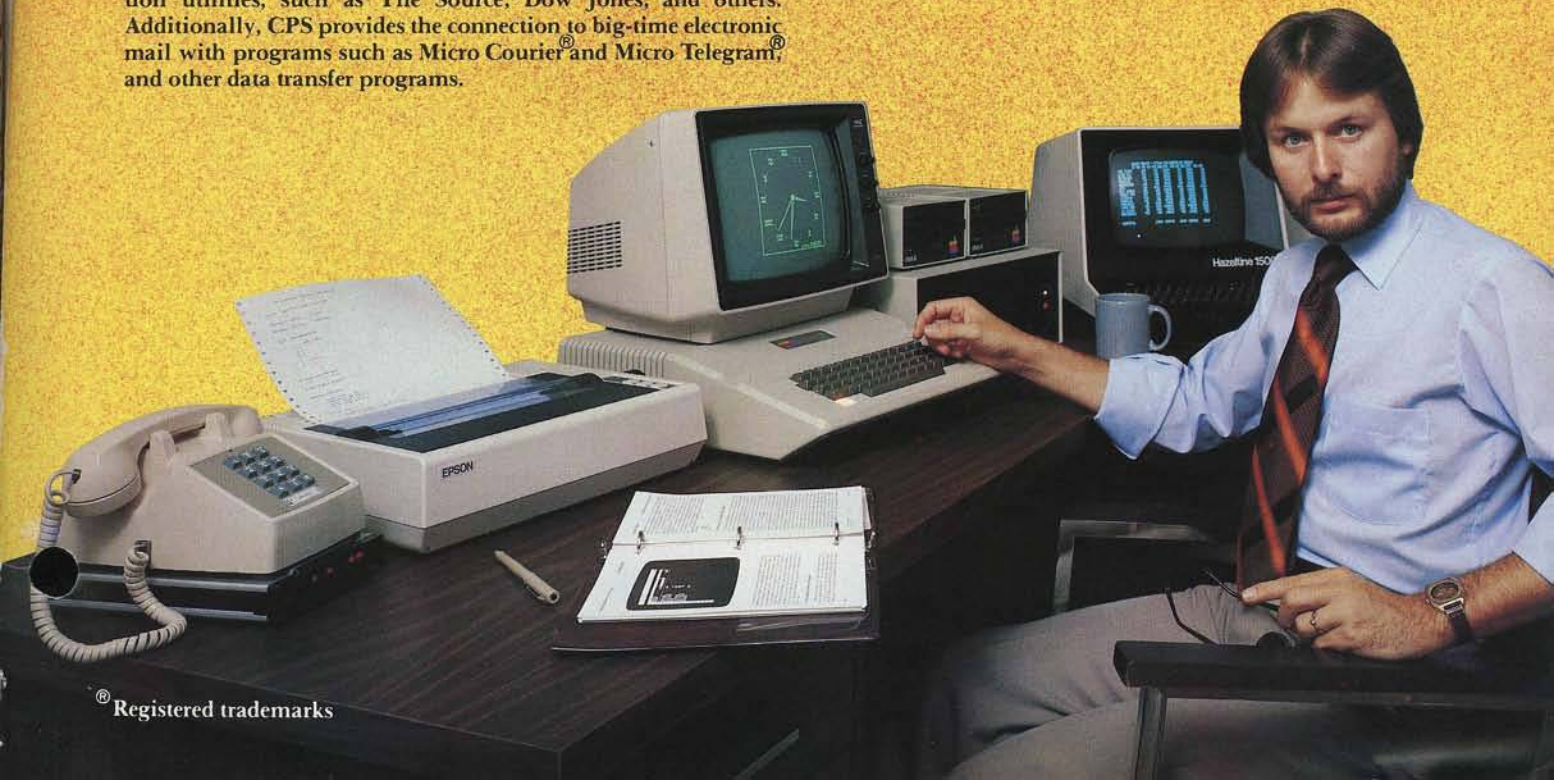
Connect with Easy Use!

CPS has no switches to set! All functions on the card are set from a user program. Menu driven screens set up your choice of all functions on CPS and store them on-board in CMOS RAM—battery backed-up (including the clock) for over two years! To change parameters, run the set-up program again—or use special commands from your keyboard. Furthermore, most existing software programs are immediately usable with CPS. *Phantom Slot Capability* permits assignment of CPS' functions to your software's pre-defined slots.

Connect with Your Apple® Dealer

Drop by your Apple dealer and see how the CPS MultiFunction Card provides the most comprehensive capabilities for RS-232C serial interface, parallel output, and real-time calendar/clock of any card available today—*all on one card*—at one low price—competitive with *any one* of the three or more single function cards that it replaces.

Mountain Computer
INCORPORATED
300 El Pueblo Road, Scotts Valley, CA 95066
(408) 438-6650 TWX: 910 598-4504



® Registered trademarks

Designing Computer Software Programs is Serious Business

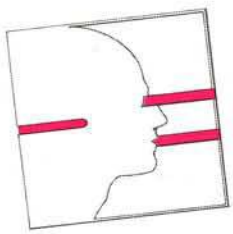
ADVANCED OPERATING SYSTEMS
A Division of Howard W. Sams & Co., Inc.

Dear Software Designer:

Microcomputers, their peripherals, and software are a part of the most volatile industry in existence today.

Companies associated with this growing area come and go at a painful rate. Presently, good software is at a premium. The average computer owner is far more sophisticated than he was a year ago. He demands an ever increasing diet of faster high resolution games that can only be met by the best programmers. The day of the small businessman, who has the time to sweat through the debugging of his general ledger while on the line, is gone.

While state of the art programmers exist, they must feel confident that their efforts will be developed and marketed by a company with the experience and strength to assure complete success for their products.



Advanced Operating Systems is a division of ITT and specifically related to Howard W. Sams & Co., Inc. (Sams Books). Our professional experience goes back to the birth of the microcomputer industry. Advanced Operating Systems benefits from the strength of ITT, one of the largest companies in the world, as well.

Advanced Operating Systems is committed to helping the micro trade grow in a businesslike and professional manner. Our 150 representatives in the field will sell your product both to computer stores and direct to the consumer. You'll be represented by one of the top advertising agencies in the United States. Your product will benefit from professional package design, effective copy and layout design. You can be assured your program will receive extensive advertising in both computer and general interest magazines, as well as catalog sales.

If you want the security of dealing with a company that will reliably send your royalty checks month after month, while providing you with professional, yet personal attention, write or call me to discuss your programming efforts.

We are easy people to deal with. We understand your problems, and we pay competitive royalties. Even more importantly, we can do more for you and your program than any existing software firm.

We are interested in state of the art games, business, professional and utility programs.

Let me hear of your work, completed or still in process. We are ready to go to work for you!

Sincerely,

Palmer T. Wolf
Software Acquisition Manager

450 St. John Road
Suite 792
Michigan City, IN 46360
(219-879-4693)