NO. 45 $2.50 FEBRUARY 1982

MICRO™

THE 6502/6809 JOURNAL

FORTH Feature

Using Atari's Countdown Timers

Utilities for the Color Computer

OSI Feature


apple II


MICROSOFT
TASC
The Applesoft Compiler

JUDGE THE REST, THEN BUY THE BEST

Only GIMIX offers you **SOFTWARE SWITCHING** between **MICROWARE's OS-9** and **TSC's FLEX**. Plus you get the power of the GMXBUG system monitor with its advanced debugging utility, and memory manipulation routines. A wide variety of languages and other software is available for these two predominant 6809 Disk Operating Systems.

You can order a system to meet your needs, or select from the 6809 Systems featured below.

JUDGE THE FEATURES AND QUALITY OF GIMIX 6809 SYSTEMS

GIMIX' CLASSY CHASSIS™ is a heavyweight aluminum mainframe cabinet with back panel cutouts to conveniently connect your terminals, printers, drives, monitors, etc. A 3 position keyswitch lets you lock out the reset switch. The power supply features a ferro-resonant constant voltage transformer that supplies 8V at 30 amps, + 15V at 5 amps, and − 15V at 5 amps to insure against problems caused by adverse power input conditions. It supplies power for all the boards in a fully loaded system plus two 5 ¼'' drives (yes! even a Winchester) that can be installed in the cabinet. The Mother board has fifteen 50 pin and eight 30 pin slots to give you the most room for expansion of any SS50 system available. 11 standard baud rates from 75 to 38.4K are provided and the I / O section has its own extended addressing to permit the maximum memory address space to be used. The 2 Mhz 6809 CPU card has both a time of day clock with battery back-up and a 6840 programmable timer. It also contains 1K RAM, 4 PROM/ROM/RAM sockets, and provides for an optional 9511A or 9512 Arithmetic Processor. The RAM boards use high speed, low power STATIC memory that is fully compatible with any DMA technique. STATIC RAM requires no refresh timing, no wait states or clock stretching, and allows fast, reliable operation. The system includes a 2 port RS232 serial interface and cables. All GIMIX boards use gold plated bus connectors and are fully socketed. GIMIX designs, manufactures, and tests in-house its complete line of products. All boards are twice tested, and burned in electrically to insure reliability and freedom from infant mortality of component parts. All systems are assembled and then retested as a system after being configured to your specific order.

56KB 2MHZ 6809 SYSTEMS WITH GMXBUX/FLEX/OS-9 SOFTWARE SELECTABLE

With #58 single density disk controller	**$2988.59**
With #68 DMA double density disk controller	**$3248.49**
to substitute Non-volatile CMOS RAM with battery back-up, add	300.00
for 50 Hz export power supply models, add	30.00

Either controller can be used with any combination of 5'' and/or 8'' drives, up to 4 drives total, have data recovery circuits (data separators), and are designed to fully meet the timing requirements of the controller I.C.s.

5 ¼'' DRIVES INSTALLED IN THE ABOVE with all necessary cables

	SINGLE DENSITY		DOUBLE DENSITY		
	Formatted	Unformatted	Formatted	Unformatted	
40 track (48TPI) single sided	199,680	250,000	341,424	500,000	2 for **$700.00**
40 track (48TPI) double sided	399,360	500,000	718,848	1,000,000	2 for **900.00**
80 track (96TPI) single	404,480	500,000	728,064	1,000,000	2 for **900.00**
80 track (96TPI) double	808,960	1,000,000	1,456,128	2,000,000	2 for **1300.00**

Chart shows total capacity in Bytes for 2 drives.

Contact GIMIX for price and availability of 8'' floppy disk drives and cabinets; and 5'' and 8'' Winchester hard disk system.

128KB 2Mhz 6809 DMA Systems for use with TSC's UNIFLEX or MICROWARES's OS-9 Level 2

(Software and drives not included)	**$3798.39**
to substitute 128KB CMOS RAM with battery back-up, add	600.00
for each additional 64KB NMOS STATIC RAM board, add	639.67
for each additional 64KB CMOS STATIC RAM board, add	988.64
for 50 Hz export power supply, add	30.00

***NOTE:** UNIFLEX can not be used with 5'' minifloppy drives.*

GIMIX has a wide variety of RAM, ROM, Serial and Parallel I/O, Video, Graphics, and other SS50 bus cards that can be added now or in the future. Phone or write for more complete information and brochure.

THE SUN NEVER SETS ON GIMIX USERS

GIMIX Systems are found on every continent, except Antarctica. (Any users there? If so, please contact GIMIX so we can change this.) A representative group of GIMIX users includes: **Government Research and Scientific Organizations** in Australia, Canada, U.K., and in the U.S.; NASA, Oak Ridge, White Plains, Fermilab, Argonne, Scripps, Sloan Kettering, Los Alamos National Labs, AURA. **Universities:** Carleton, Waterloo, Royal Military College, in Canada; Trier in Germany; and in the U.S.; Stanford, SUNY, Harvard, UCSD, Mississippi, Georgia Tech. **Industrial users** in Hong Kong, Malaysia, South Africa, Germany, Sweden, and in the U.S.; GTE, Becton Dickinson, American Hoechst, Monsanto, Allied, Honeywell, Perkin Elmer, Johnson Controls, Associated Press, Aydin, Newkirk Electric, Revere Sugar, HI-G/AMS Controls, Chevron. **Computer mainframe and peripheral manufacturers,** IBM, OKI, Computer Peripherals Inc., Qume, Floating Point Systems. **Software houses;** Microware, T.S.C., Lucidata, Norpak, Talbot, Stylo Systems, AAA, HHH, Frank Hogg Labs, Epstein Associates, Softwest, Dynasoft, Research Resources U.K., Microworks, Analog Systems, Computerized Business Systems.

TO ORDER BY MAIL

SEND CHECK OR MONEY ORDER OR USE YOUR VISA OR MASTER CHARGE. Please allow 3 weeks for personal checks to clear.

U.S. orders add $5 handling if order is under $200.00. Foreign orders add $10 handling if order is under $200.00.

Foreign orders over $200.00 will be shipped via Emery Air Freight COLLECT, and we will charge no handling. All orders must be prepaid in U.S. funds. Please note that foreign checks have been taking about 8 weeks for collection so we would advise wiring money, or checks drawn on a bank account in the U.S.. Our bank is the Continental Illinois National Bank of Chicago, account #73-32033 Visa or Master Charge also accepted.

GIMIX INC. reserves the right to change pricing and product specifications at any time without further notice.

are registered trademarks of GIMIX Inc.

FLEX AND Uniflex are trademarks of Technical Systems
Consultants Inc. OS-9 is a trademark of Microware Inc. See their ads for
other **GIMIX** compatible software

GIMIX Systems are chosen by the Pros because of quality, reliability and features.

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

YOU GET

WORD PROCESSING POWER

AND

PRINT ... USING

FOR YOUR APPLESOFT* PROGRAMS

WITH THE

FORMAT ROM

WITH THIS POWERFUL ROM IN YOUR MOUNTAIN COMPUTER'S ROMPLUS* OR ANDROMEDA'S ROMBOARD* YOU WILL MAKE APPLESOFT* BASIC ONE STEP MORE POWERFUL THAN THE MOST ADVANCED LANGUAGE AVAILABLE. AND - BECAUSE READ ONLY MEMORY NEVER FORGETS, THESE NEW COMMANDS BECOME A PERMANENT PART OF APPLESOFT'S* BASIC LANGUAGE.

PRINT STATEMENT FORMATTING - Can you imagine formatting a letter with Applesoft print statements? Monumental to say the least - you would have to agree. Well - with the print statement formatting capabilities of the FORMAT ROM all you have to worry about is what you want your print statements to say and leave the rest to the FORMAT ROM. You tell the FORMAT ROM, right in your Applesoft program, how you want your letter, manuscript, etc. to look and best of all, formatting commands can be changed at any point in your program with simple basic commands. FORMAT ROM gives you the same powerful commands as a dedicated word processor costing hundreds of dollars. Here are some of the powerful formatting instructions which you will have with the format rom: **DEFINE PRINTING MARGINS - DEFINE THE LENGTH OF A PRINTED LINE - RIGHT HAND JUSTIFICATION - NO WORD WRAP AROUND JUSTIFICATION - AUTOMATIC INDENTING OR OUTDENTING OF PARAGRAPHS - SKIPPING OF LINES AFTER EACH PRINTED LINE** (Double space/Triple space etc.) - **PAUSE AFTER PRINTING SO MANY LINES - CENTER THE NEXT LINE OF TEXT** (FORMAT ROM FIGURES IT OUT FOR YOU) - **USER DEFINED CHARACTER SUBSTITUTION** - FORMAT ROM will even put two spaces after each period it finds, even if you forget and only put one space.

PRINT...USING - Will format and tabulate the output of alpha/numerical data that has been predefined by you without having to go through the basic programming steps to get the format requirements. Five modes of PRINT...USING are available: Alpha mode - will tabulate and right justify strings of any length. Alpha/Numeric mode - will tabulate any predefined string to the left of any predefined numerical data which can be formatted in any of the following output modes; Floating Point or Integer with any number of digits, Fixed output with any number of decimal places with round off capabilities of positive or negative numbers, Fixed output with commas inserted every third digit from the left of the decimal point. All PRINT...USING routines will tabulate, right justify, line up all decimal points, pad the right and left side of a number with any predefined character and can be used within formulas or equations which will then format the mathematical result. Overflow messages or symbols, defined by you, can be printed if a number overflows the limits specified by you. PRINT...USING can be used in immediate or deferred and is compatible with all Applesoft basic commands.

FORMAT ROM will support all printers, 80 column boards, lower case adapters, and requires 48K, FP in ROM, DOS 3.2 or 3.3, M.C's ROMPLUS* or Andromeda's ROMboard*	**$49.95**

OTHER ROMS AVAILABLE: All ROMS are compatible with MC's Romplus or Andromeda's ROM Board.

- **DUAL DOS ROMS** - Switch from one DOS (3.2 or 3.3) to the other without booting	**$54.95**
- **FP RENUMBER/MERGE ROM** - Apple Computer's infamous renumber program	**$39.95**
- **BASICS ROM** - Will boot standard, special, and dedicated 13 sector disks	**$39.95**
- **FP EDITROM** - Global search, change, and remove. (Works jointly with PLE)	**$39.95**
- **COMMAND ROM** - Catalog Command Menu and Disk Map	**$39.95**
- **DISK COPY/SPACE ROM** - Duplicates 13 or 16 Sector Disks	**$39.95**
- **FP UTILITY ROM** - Auto Line Numbering, LIST CTRL, RESURRECT KILLED PROG. Alphabetize DISK CAT., Expung DOS	**$39.95**

(Check or M.O.) Visa or MasterCard Accepted

SOFT CTRL SYSTEMS, BOX 599, WEST MILFORD, NJ 07480 **201-728-8750**

*REGISTERED TRADEMARK
ALL FIRMWARE IS COPYRIGHTED

NO. 45 FEBRUARY 1982

MICRO™

THE **6502/6809** JOURNAL

STAFF

Editor/Publisher
ROBERT M. TRIPP

Associate Publisher
MARY GRACE SMITH

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

Advertising
Sales Representative
KEVIN B. RUSHALKO
603/547-2970

DEPARTMENTS

THE CHIEFTAIN™ 5¼-INCH WINCHESTER HARD DISK COMPUTER

**SO ADVANCED IN SO MANY WAYS . . .
AND SO COST-EFFECTIVE . . .
IT OBSOLETES MOST OTHER SYSTEMS
AVAILABLE TODAY AT ANY PRICE.**

● HARD DISK SYSTEM CAPACITY

The Chieftain series includes 5¼- and 8-inch Winchesters that range from 4- to 60-megabyte capacity, and higher as technology advances. All hard disk Chieftains include 64-k memory with two serial ports and DOS69D disk operating system.

● LIGHTNING ACCESS TIME

Average access time for 5¼-inch Winchesters is 70-msec, comparable to far more costly hard disk systems. That means data transfer ***ten-times faster*** than floppy disk systems.

The Chieftain Computer Systems:

Here are the Chieftain 6809-based hard disk computers that are destined to change data processing . . .

☐ **CHIEFTAIN 95W4**
4-megabyte, 5¼-inch Winchester with a 360-k floppy disk drive (pictured).

☐ **CHIEFTAIN 95XW4**
4-megabyte, 5¼-inch Winchester with a 750-k octo-density floppy disk drive.

☐ **CHIEFTAIN 98W15**
15-megabyte, 5¼-inch Winchester with a 1-megabyte 8-inch floppy disk drive.

☐ **CHIEFTAIN 9W15T20**
15-megabyte, 5¼-inch Winchester with a 20-megabyte tape streamer.

● 2-MHZ OPERATION

All Chieftains operate at 2-MHz, regardless of disk storage type or operating system used. Compare this to other hard disk systems, no matter ***how*** much they cost!

● DMA DATA TRANSFER

DMA data transfer to-and-from tape and disk is provided for optimum speed. A special design technique eliminates the necessity of halting the processor to wait for data which normally transfers at a slower speed, determined by the rotational velocity of the disk.

● RUNS UNDER DOS OR OS-9

No matter which Chieftain you select . . . 5¼- or 8-inch floppy, or 5¼- or 8-inch Winchester with tape or floppy back-up . . . they ***all*** run under DOS or OS-9 with ***no need*** to modify hardware or software.

● UNBOUNDED FLEXIBILITY

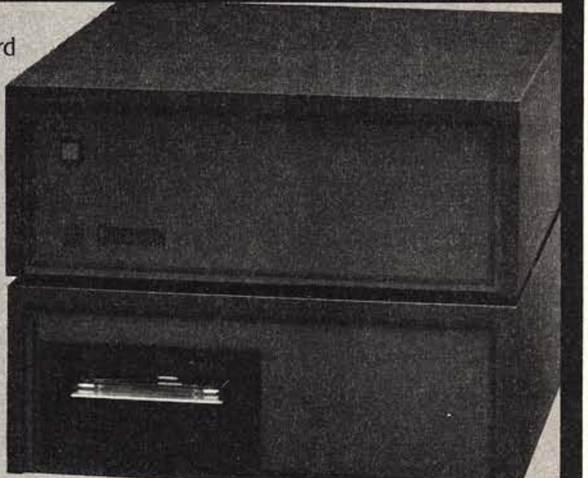
You'll probably never use it, but any Chieftain hard disk system can drive up to 20 other Winchesters, and four tape drives, with a single DMA interface board!

● SMOKE SIGNAL'S HERITAGE OF EXCELLENCE

This new-generation computer is accompanied by the same ***Endurance-Certified*** quality Dealers and end-users all over the world have come to expect from Smoke Signal. And support, software selection and extremely competitive pricing are very much a part of that enviable reputation.

20-Megabyte Tape Streamer Back-Up Option

Available with all Chieftain hard disk configurations. This cartridge tape capability provides full 20-megabyte disk back-up in less than five minutes with just one command, or copy command for individual file transfers. Transfers data tape-to-disk or disk-to-tape. Floppy back-up is also available in a variety of configurations.



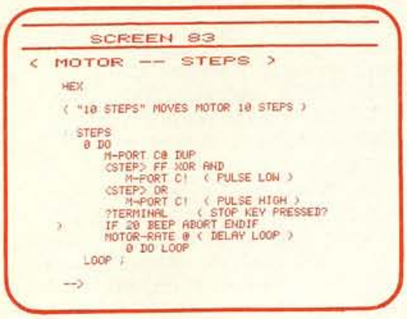
Write or call today for details (including the low prices) on the Chieftain Series. . . and on dealership opportunities

SMOKE SIGNAL BROADCASTING®

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

Name ____________________
Company ____________________
Address ____________________
City ____________ State ________ Zip ________
Telephone (   ) ____________________

About the Cover



```
      SCREEN 83
< MOTOR -- STEPS >

  HEX

  ( "10 STEPS" MOVES MOTOR 10 STEPS )

  : STEPS
     0 DO
        M-PORT C@ DUP
        <STEP> FF XOR AND
           M-PORT C!  ( PULSE LOW )
        <STEP> OR
           M-PORT C!  ( PULSE HIGH )
        ?TERMINAL     ( STOP KEY PRESSED?
)       IF 20 BEEP ABORT ENDIF
        MOTOR-RATE @ ( DELAY LOOP )
           0 DO LOOP
     LOOP ;

  -->
```

The body of water in this spectacular sunset shot is none other than the Firth of Forth, a few miles from Edinburgh, Scotland. The name of the language FORTH, covered in this issue, has quite a different derivation. See FORTHword (p. 83).

The cover scene was photographed with a Nikon F2 24 mm lens and Kodachrome 64 film.

Cover photo by Kevin Harkins
Kevin Harkins Studio
Lowell, Massachusetts

MICRO is published monthly by:
MICRO INK, Inc., Chelmsford, MA 01824
Second Class postage paid at:
Chelmsford, MA 01824 and Framingham, MA 01701
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
34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

or call
617/256-5515
Telex: 955329 TLX SRVC
800-227-1617

Subscription Rates	*Per Year*
U.S.	$24.00 2 yr. / $42.00
Foreign surface mail	$27.00
Air mail:	
Europe	$42.00
Mexico, Central America, Middle East, North Africa, Central Africa	$48.00
South America, South Africa, Far East, Australasia, New Zealand	$72.00

MICRO
Editorial

Dozens of magazines, newspapers, and newsletters report on and analyze the rapidly changing microcomputer field. Any publication intending to cover this industry effectively, has to change constantly too.

MICRO is no exception. Loyal readers have noticed not only physical changes in MICRO over the last four years, but also changes in content. The journal's first issues were devoted to single board computers, and technical "how-to" articles — material much like what you would find in a user's manual. As the Apple, PET, and other 6502 computers became popular, manufacturers began supplying better documentation. Consequently MICRO extended the base of computers covered in the magazine, as well as the type of information provided.

Now, as you know, MICRO offers its readers programming techniques and aids, enhancements, applications, and hardware pieces. And most recently MICRO expanded its coverage to include 6809-based machines. We want all of our readers to get the most out of their computers. We intend to publish articles that will help you to do just that, no matter which system you own. We expect to continue to grow and change with the industry.

To accomplish this expansion and development successfully, MICRO needs the support of its readers. We are currently developing a pool of 6809, Atari, VIC, and TRS-80 Color Computer authors. If you own, or have access, to any of these machines, and have developed applications or techniques, you may be a potential MICRO author.

We're also encouraging the submission of less system-specific articles: applications or techniques that can be applied to more than one machine, or general ideas beneficial to the users of many different computers.

To help broaden our coverage we have added two new columns: "From Here to Atari," and "The Single Life." We encourage anyone interested in writing a 6809 or TRS-80 Color Computer column to send us a proposal. In April we are planning a special feature on 6809-based systems. We would be more than happy to accept input — ideas, suggestions, questions, articles — from 6809 users.

We've also added a new department, written strictly for your entertainment. We think you'll enjoy "It's All Ones and Zeros." On the serious side, another new item is the Technical Data Sheet. In this you'll find technical information in a concise, easy-to-read form. This month we offer a reference for the 6502 programmer.

One of the interesting things about the computer field is that nobody knows everything and everybody knows something. Even with a few months of experience you may have discovered or developed something that would be useful and helpful to other users. In other words, everyone is a novice in some aspects; the most experienced user can benefit from a fresh perspective.

Submitting an article to MICRO is easy. Many computerists think they "can't write," but it's conveying the information that is important. We have a good editorial department to help put the copy into final form. All we ask is that you double-space your typewritten copy, number and put your name on each page, and use clear, simple language. If you include a listing, we encourage you to submit it on magnetic media, but in any case, make sure it is printed with clean, black ink. If you have any more questions about manuscript or listing format, write for a copy of our new Writer's Guide.

Our aim is to provide a magazine full of useful and interesting information. We hope you'll help us to do that, both by letting us know what you need and want and by sharing your discoveries through articles, letters, columns, and short subjects entries.

Marjorie J. Morse

MICRO

New Publications

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

Directories

The Apple II Blue Book, WIDL Video (5245 W. Diversey Ave., Chicago, IL 60639), 1981, 10 5/8 × 8 3/8 inches, paperback. $19.95

A directory of software, hardware, peripherals, and information for the Apple. The book lists reference manuals, publications, newsletters, users groups, clubs, time sharing systems, and more.

CONTENTS: *Software Source Index. Utility Software Programs. Data Base Management Software Programs. Word Processing Software Programs. Graphics Software Programs. Business Software Programs. Games & Entertainment Software Programs. Education Software Programs*—Administration; Basic Skills; Language Arts; Foreign Language Programs; Elementary Mathematics; Advanced Mathematics; General Science; Biology; Chemistry; Physics; Computer Science; Music; Business; Social Studies; Misc. Program Collections Covering Several Areas of Study. *Boards. Peripherals. Accessories. Music and Speech. Storage. Misc. Resources. Supplies. Power Supplies, Regulation, and Static Control. Books. Magazines and Publications. Special Apples. Networking. Time Sharing and Communications. User Groups. Authorized Apple Dealers.*

International Microcomputer Software Directory, Imprint Software (South Howes St., Ft. Collins, CO 80521), 1981, 11 × 8½ inches, paperback.
ISBN: 0-907352-030 $29.95

A reference for microcomputer software for all applications and systems. The information is drawn from a database that is continually updated from all parts of the world through offices in Britain and America. The directory contains three sections: System Classification, Subject Classification, and Software House Classification.

CONTENTS: *Acknowledgements. Publisher's Preface. How to Use the Directory. Buyer's Guide. Notes to Software Houses and Dealers. Other Serivces of Imprint Software. Subject and Category Codes. System Codes (Machines, Microprocessors, and Operating Systems), and Abbreviations. Section 1*—System Classification. *Section 2*—Subject Classification. *Section 3*—Software House Classification. *Appendix 1*—Table of Machines and Operating Systems by Microprocessors. *Appendix 2*—Glossary of Terms. *Appendix 3*—Software Houses — How to Update Your IMSD Listing. *Index of Program Names.*

For Beginners

The Computers Are Coming by Irv Brechner. Irv Brechner Publisher (P.O. Box 453, Livingston, NJ 07039), 1981, 92 pages, 6 ×9 inches, paperback. $4.95

An introduction to computers written from a non-technical angle.

CONTENTS: *The Computers Are Coming. Don't Be Afraid... It Won't Bite You. Try It... You'll Like It. Nobody's Perfect. Thanks For The Memories. Always Willing and Able... Never Says "No". Runs Circles Around A Speeding Bullet! Doesn't Do Windows. If You Can Write in English. What Goes In Is What Comes Out. Hardware... Software... Who Cares! And, On Tonight's Progam. Keeping Up With The Kids. Now It's Time For Fun and Games! Let's Go Down To The Corner Computer Store. No Mortgage Necessary! See You... On Line!*

VIC

Understanding Your VIC by David Schultz. Total Information Services, Inc. (P.O. Box 921, Los Alamos, NM 87544), 1981, 140 pages, 10 6/8 × 8 3/8 inches, paperback.

A tutorial presentation on how BASIC works on the VIC. Contains step-by-step exercises for self-instruction.

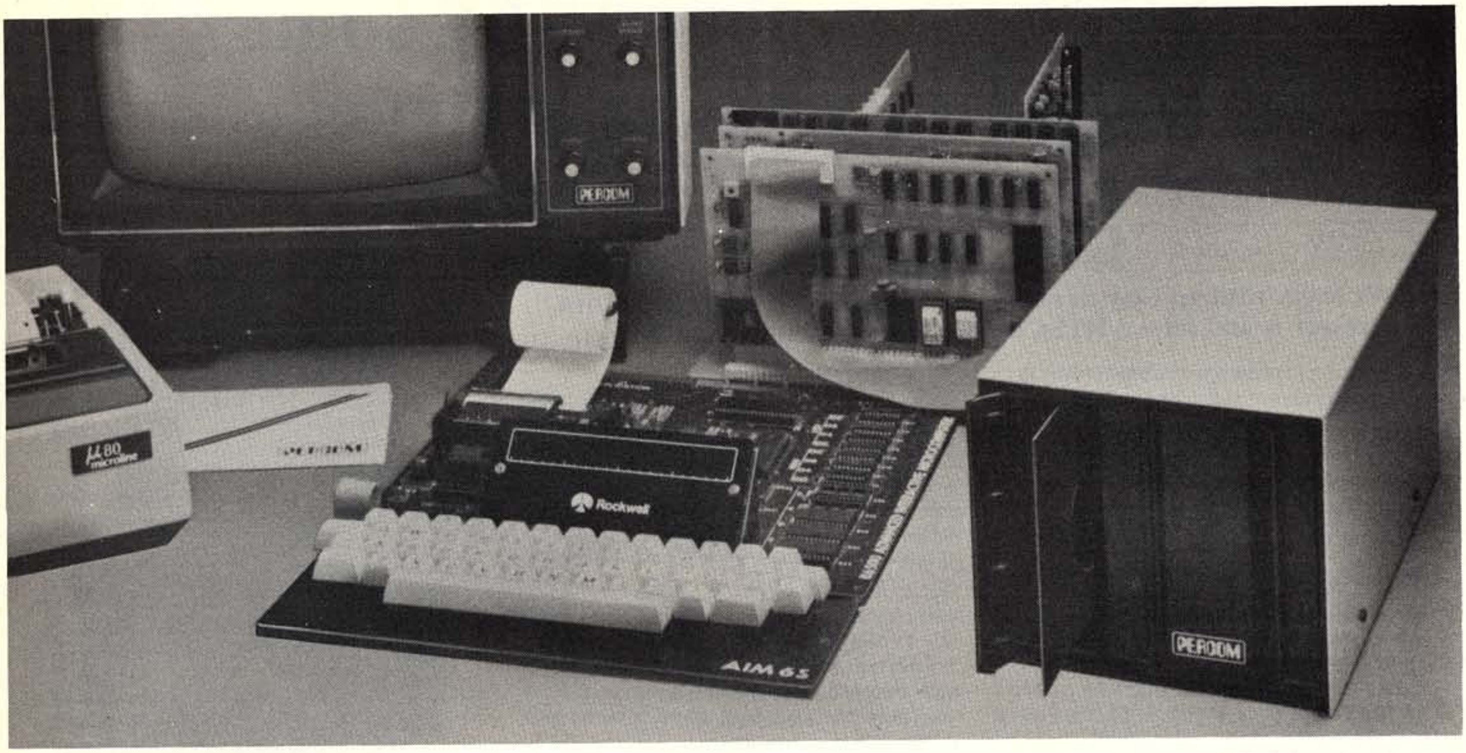
CONTENTS: *Introduction*—Assumptions Made About the User; Exercises; Programming; VIC Keyboard and TV Display; Notation; BASIC Overview. *CBM BASIC Calculator Mode*—Using Strings in CBM BASIC; Numeric and Fractional Values; Conversion of Data; Balance Your Checkbook in Calculator Mode; Reserved Words; Modes of Variables and Constants. *Inputting A Program*—Blanks; Multiple Program Statements; Typing Mistakes; CBM BASIC Commands; 'STOP' and CONT. *Getting Information Out of Your Program*—Output Formats — Numeric Data; Output Formats — Character Strings; Spacing. *Getting Information Into Your Program*—Design Goals; INPUT; READ, DATA, and RESTORE. *Data Representation*—Largest Numeric Value; Smallest Numeric Value; Integer Range; Memory Space Used; Number of Significant Digits; Rounding. *Using the Cassette for Program Storage*—SAVE a Program; VERIFY a Program; LOAD a Program. *Branching*—GOTO; ON X GOTO. *The IF Statement*—THEN Form; GOTO Form; Multiple Statement. *Subroutines*—GOSUB; RETURN; ON X GOSUB. *Strings*—Legal String Names; Subscripted String Variables; Maximum String Length. *Operations on Strings*—Statements; Functions. *Data Representation and Processing Programs*—Number System Conversion Programs; Logical Operations Program. *Subscripted Variables*—Legal Subscripts; Dimensions. *Program Design*—Psuedo Code; Data Description; Typical BASIC Implementation. *Color*—Statement of the Problem; Refinement of the Solution; Psuedo Code of Get Choice; Psuedo Code of Set VIC; Miscellaneous Psuedo Code; Data Description; Writing the Program. *Sound*—The Problem; Restrictions; Sound Program Design; Refinement of the Solution; Data Definition; Writing the Program. *Appendices*—Appendix A: Program Listings; Appendix B: Data Processing Background.

Atari

Atari BASIC, Learning By Using by Thomas E. Rowley. Ing. W. Hofacker GmbH (53 Redrock Lane, Pomona, CA 91766), 1981, 73 pages, 8 × 5½ inches, paperback.
ISBN: 3-92-1682-86-X

A supplementary resource for learning BASIC programming on the Atari. Contains short programs and learning exercises. Appropriate for beginners as well as advanced users.

CONTENTS: *Introduction. Screen Drawings*—German Flag; Design; Circle; Star; Cover Page; Symbols in Graphics 2. *Special Sounds*—Sound Effects; Musical Tune; Up and Down Sound; Audible Joystick. *Keys, Paddles and Joysticks*—Console Switches; Paddle Motion; Joystick Drawing; TAB; Key Control; Pick a Key. *Specialized Screen Routines*—Player-Missile Graphics; German Font; Mixed Mode Screen; Characters in Graphics 4, 6, and 8. *Graphics and Sound Applications*—Duel; Video Art; Guessing Game; Slot Machine; Linear X-Y Plot. *PEEKs, POKEs, and Special Stuff. Appendix 1*—Description of Memory Addresses. *Appendix 2*—Player Missile Memory Map. *Appendix 3*—Building a Display List. *Appendix 4*—Calculating Screen Position.

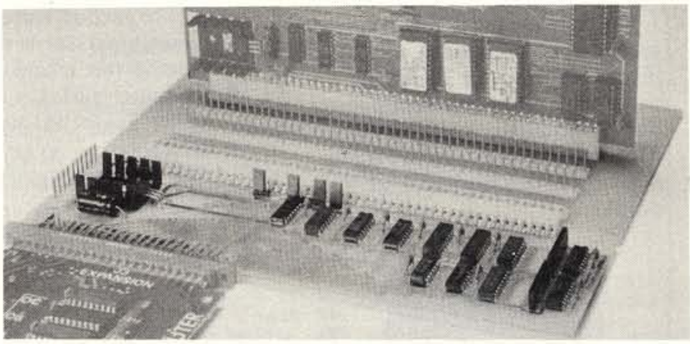


The **PERCOM 'M'** Line

Low-Cost Expansion for Your AIM, KIM or SYM Computer

Complete Disk Storage Systems from $599.95

- **The right storage capacity** – Available in 1-, 2- and 3-drive systems, with either 40- or 80-track drives.
- **Flippy storage** – Flippy drives (optional) let you flip a diskette and store data and programs on the second recording surface.
- **High Storage Capacity** – Formatted, one-side storage capacity is 102 Kbytes (40-track drive), 205 Kbytes (80-track drive).
- **Proven Controller** – The drive controller design is the same as the design used in the Percom 680X LFD mini-disk system. This system – introduced in 1977 – has given reliable service in thousands of applications. Two versions are available: the MFD-C65 for the AIM-65 expansion bus, and the MFD-C50 for the System-50 (SS-50) bus.
  - Includes an explicit data separator circuit that's reliable even at the highest bit densities.
  - Provides for on-card firmware.
  - Includes a motor inactivity time-out circuit.
  - Capable of handling up to four drives.
  - Capable of reading both hard- and soft-sectored diskettes.
- **DOS included** – The MFD disk-operating system works with the AIM monitor, editor, assembler, Basic and PL/65 programs, interface is direct, through user I/O and F1, F2 keys. Diskette includes DOS source code and library of 20 utility commands.
- **Reliability assurance** – Drives are burned-in 48 hours, under operating conditions, to flag and remove any units with latent defects.
- **Full documentation** – Comprehensive hardware and software manuals are included with each system.



Expand Your Computer with Proven System-50 Modules

The Percom M65/50 Interface Adapter connects your computer to Percom's System-50 (SS-50) motherboard, allowing you to expand your AIM, KIM or SYM with proven System-50 modules. You can add disk storage, memory modules,even a video display system. The M65/50 provides buffer-amplification of address, data and control lines. On-card decode circuitry lets you allocate address space either to the computer or to the expansion motherboard. Price: only $89.95, *including* System-50 motherboard.

System Requirements: AIM-65, KIM or SYM computer with expansion bus and four Kbytes RAM (min).

Yes . . . I'd like to know more about Percom MFD drive systems. Rush me free literature.

Send to
PERCOM DATA COMPANY, Inc., Dept. 65-M
11220 Pagemill Rd. Dallas, TX 75243

name

address

city state

zip phone number

MAIL TODAY!

PERCOM

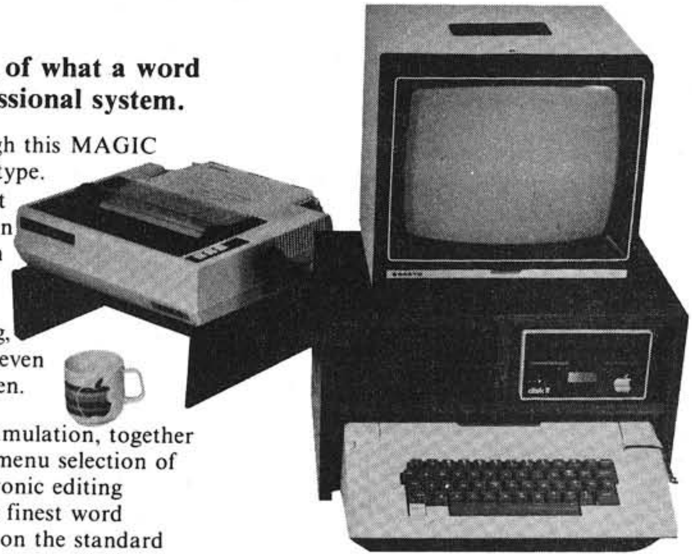
PERCOM DATA COMPANY, INC.
11220 PAGEMILL RD. DALLAS, TX 75243
(214) 340-7081

Toll-Free Order Number: **1-800-527-1222**

PRICES AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

ARTSCI explains why some word processing systems are better than others.

Let's begin with an easy to understand explanation of what a word processor is and how ARTSCI has created a professional system.



Apple is a trademark of Apple Computer Inc.

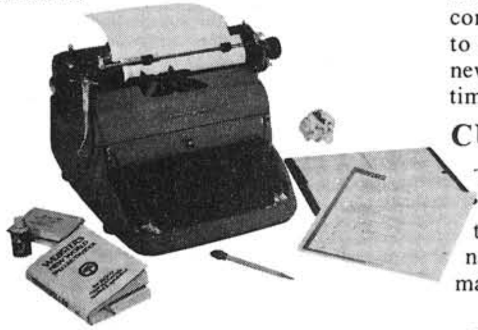
A word processing system is simply an easier, faster and less expensive way to type. With a modern word processor, documents are entered on a video screen instead of paper.

You can enter your first rough draft without concern about errors or spelling. Simply go back and insert letters, delete words and even move paragraphs with a few keystrokes. No document will ever have to be retyped.

WORD PROCESSING AND THE APPLE II

The APPLE II is the most expandable, inexpensive microcomputer available today. It can perform almost any task, including word processing.

The standard APPLE II however, uses a 40 column video display. This display causes a serious word processing problem: How do you display a full sized 80 column letter? Most word processing programs available today do not solve this problem.



THE MAGIC WINDOW

ARTSCI has developed the MAGIC WINDOW word processing system that incorporates the full power of a professional word processor and solves the APPLE'S display problem without expensive hardware.

The first feature of a professional word processing system is the ability to enter and edit data in a fast and friendly manner. The MAGIC WINDOW operates just like a standard typewriter. The electronic paper moves to the left across the video screen as you type. Almost any size document can be represented on the video screen. You can see the edges of the paper through this MAGIC WINDOW as you type.

The rule is: What you see on the screen is what you'll get in print. However, if you print using proportional spacing, the result will look even better than the screen.

This typewriter simulation, together with simple to use menu selection of functions and electronic editing abilities, creates the finest word processor available on the standard APPLE II.

MAGIC SPELL

The second feature of an advanced word processor is the ability to find and correct mistakes. The most common mistakes in most documents is the misspelled word.

ARTSCI's MAGIC SPELL program will take any document you can create and find spelling errors. Over 14,000 commonly misspelled words are known to MAGIC SPELL. You can also add new words to the vocabulary at any time.

CUSTOM LETTERS

The third feature of a professional word processing system is the ability to alter a document by replacing names and other related data from mailing lists.

Form letters, invoices, and almost any document can be individualized by replacing names, addresses or any other personal data anywhere in the document using ARTSCI'S BASIC MAILER.

By using the BASIC MAILER you can take any mailing list and sort through the list by different criteria and print personalized letters with a few simple keystrokes.

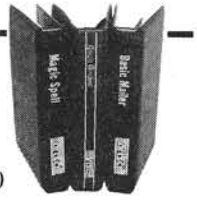
A COMPLETE SYSTEM

These three programs, THE MAGIC WINDOW, MAGIC SPELL AND THE BASIC MAILER, together form the only complete and professional word processing system available on the standard APPLE II computer system.

ARTSCI TAKES THE WORK OUT OF WORD PROCESSING

For a more thorough explanation of the ARTSCI word processing system send for our free booklet.

artsci INC



Mail to:
ARTSCI INC.
10432 Burbank Blvd.
North Hollywood,
California 91601-9990
(213) 985-2922

Name ____________________

Company ____________________

Address ____________________

City __________ State __________ Zip ____

Phone ____________________

ST PE ML CC BY CA

Utilities for the Color Computer

This versatile routine allows Color Computer users to dump or disassemble the 6809 or ASCII code in any section of memory, including the BASIC or expansion ROMs. The hex and ASCII dump output is titled and paginated. The disassembly output produced contains standard 6809 mnemonics. For learning about Color BASIC, and for writing machine language routines, the Inspector is an invaluable programming tool.

Leo E. Garrett
P.O. Box 4946
Brownsville, Texas 78520

My main interests in microprocessors are hardware and assembly language programming. For the past two years I have been using the Motorola TVBUG. It is a single board machine with a 6847 Color Video generator. One of the programs I wrote for it was a 4K BASIC with line-drawing commands.

Naturally, I was interested in the TRS-80C Color Computer. I recently managed to buy one with 16K RAM and Extended Color BASIC. The manuals, though comprehensive, contain little specific information on the interpreter or its subroutines. The machine language programmer was neglected again.

The TRS-80C Color BASIC ROMs have many subroutines that would be very useful to the machine language programmer if their entry points were only known. My first real program on my new 80C was aimed at that problem. The result was the Inspector.

The Inspector will help you dig around in the BASIC ROMs productively. Its two modes, hex dump and disassemble, produce titled listings on numbered pages. While it doesn't give you symbols and labels, it does give you access to the ROM routines.

Hex Dump

Listing 1a is a sample of the hex dump function. It has a "LONG" option suited to 80 char/line printers and a "SHORT" option which fits eight bytes on one screen line for browsing. Beneath each hex byte it prints the ASCII equivalent. If you select the "CLEAR B7" option, it will ignore bit 7 when printing the ASCII. This is nice for those cases where bit 7 is used as a flag or confusion factor.

Hex dump makes it easy to pick out messages, command tables, and other data areas. If you're running the disassembler and get screwy output, the hex dump helps you make a little more sense of it.

Disassembler

The disassembler prints the selected number of lines in standard 6809 mnemonics, with a couple of exceptions. (See listing 1b.) My printer doesn't print brackets, so parentheses were substituted.

The 6809 doesn't have bit-specific instructions (SEC, CLI, etc.) for the condition code register. Instead, the

Listing 1a

```
                        *NONSENSE PROGRAM
                        *MICROWARE SDS80C ASSEMBLER
0001 0600                      ORG $3FC0
0002 3FC0               TARGET EQU $F8D2        EXTERNAL
0003 3FC0 4E414D        DATA   FCC 'NAM'
0004 3FC3 C504                 FDB $C504        B7 SET IN 'E'
0005 3FC5 308CF8        BEGIN  LEAX DATA,PCR
0006 3FC8 1C5E                 ANDCC #$5E       CLEAR C,H,E
0007 3FCA 3670                 PSHU X,Y,S
0008 3FCC 1A53                 ORCC #83         SET F,I,V,C
0009 3FCE 328DB900             LEAS TARGET,PCR
0010 3FD2 ADF4                 JSR  ↑,S←        INDRCT INDX
0011 3FD4 3796                 PULU  D,PC,X     D=A+B
0012 3FD6 10AF9840             STY   ↑$40,X←
0013 3FDA                      END BEGIN        TRANSFER

***NONSENSE;CLR B7/LONG OPTIONS***

3FC0 4E 41 4D C5 04 30 8C F8 1C 5E 36 70 1A 53 32 8D
      N  A  M  E  .  0  .  .  .  →  6  .  .  S  2  .
3FD0 B9 00 AD F4 37 96 10 AF 98 40 41 C1 42 C2 43 C3
      9  .  -  .  7  .  .  /  .  @  A  A  B  B  C  C
```

Listing 1b

```
***NONSENSE***

3FC0    4E            ????
3FC1    41            ????
3FC2    4D            TSTA
3FC3    C5 04         BITB  #$04
3FC5    30 8C F8      LEAX  $3FC0,PCR
3FC8    1C 5E         ANDC  #$5E (CLR C,H,E)
3FCA    36 70         PSHU  X,Y,S/U
3FCC    1A 53         ORCC  #$53 (SET C,V,I,F)
3FCE    32 8D B900    LEAS  $F8D2,PCR
3FD2    AD F4         JSR   (,S)
3FD4    37 96         PULU  A,B,X,PC
3FD6 10AF 98 40       STY   ( 64,X)
3FDA    41            ????
3FDB    C1 42         CMPB  #$42
3FDD    C2 43         SBCB  #$43
3FDF    C3 2033       ADDD  #$2033
```

ORCC #nn and ANDC #nn instructions are used to set or clear flags. The disassembler specifies SET or CLR and lists the flags. It does the same for the CWAI #nn opcode.

For the indexed mode the disassembler prints the offset as a signed decimal number. This makes it a lot easier to see what is going where.

For the relative opcodes, the TARGET address is printed in hex. Again, it is much easier to see where the pointer is, or the destination of a branch. No hex arithmetic is required!

ROM Cartridges

The Color Computer uses the FIRQ line to detect the presence of a ROM cartridge. I just put a sliver of tape over that pin when using Inspector with a cartridge. To use the ROM enter EXEC &HC000. (See figure 1.)

Portability

I used Extended BASIC's hex operators extensively. If you are using a BASIC that doesn't have them, you must re-work those areas. Many methods for hex conversion have been published. In many cases a simple subsitution of decimal equivalents will be adequate. The subroutines that input and output hex will require more work.

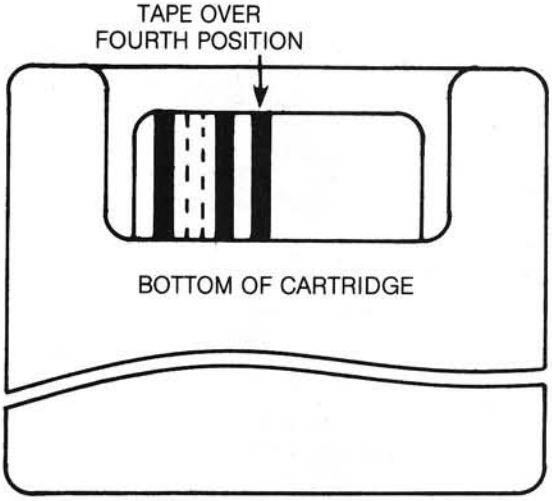


Figure 1

The bare program, stripped of all REMs, takes 13K to run. Observe the guidelines in the first lines of the listing.

Results

Table 1 shows some of the entry points and conditions I have uncovered. (There are many more.)

I had trouble with my tape recordings. The leader was too short, so the computer would try to read before the motor came up to speed. POKE &H92,1 triples the delay and all tape read operations are now good.

DP REG must be set to zero before calling.

$A30A	Write char in 'A' to screen
$A2BF	Write 'A' to printer
$A1B1	Wait for key; Char returned in 'A'
$A928	Clear screen, home cursor; 'X', 'B' changed
$A393	Get line into buffer @ $02DD; 'X' = $02DC; End byte = 0
$AC20	Move block of memory starting at top $41 = Destination top address $43 = Source top address $47 = Source bottom address $45 = Destination bottom address after move

Table 1: **TRS-80C subroutines discovered through the Inspector**

```
5 '***TYPE IN ONLY THE ACTIVE LINES.  PROGRAM WILL NOT
6 '    RUN WITH ALL REMARKS LEFT IN!!!!
7 '***ENTER 'PCLEAR 1' BEFORE ENTERING/LOADING
    PROGRAM
10 CLEAR 1500:GOSUB 70
15 '*MENU/HEADER
20 CLS:PRINT:PRINT"INSPECTOR 2.0"
30 PRINT"LEO E. GARRETT,9/4/81":PRINT:PRINT"SELECT:"
40 PRINT" 1. DISASSEMBLER":PRINT" 2. HEXDUMP"
50 INPUT"WHICH";R:ON R GOTO 800,2350
60 GOTO 20
65 '**SEARCHING STRING TABLES IS EASIER AND FASTER
    THAN
66 '   SEARCHING 'DATA' STATEMENTS
70 DIM CC$(7),IH$(20),IH(20),EX$(15),PP$(7)
80 DIM LT$(11),LT(11),BT$(15),BT(15),AT$(15),AT(15)
90 DIM PF$(5),PF(5),PB(7),RG$(3),CT$(15),CT(15)
95 '*CONDITION CODE REG BITS
100 FOR I=0 TO 7:READ CC$(I):NEXT I
```

Listing 2: **Inspector Listing**

VIGIL

VIC

PET/CBM

Exciting, new games interactive language.
- Easy to learn with 60 - powerful commands
- Double density graphics, large number display, sound!
- 9 complete programs including - Breakout, SpaceWar, Anti. U.F.O., Maze
- PET/CBM version for OLD/NEW/4.0 ROMs and 8K of memory
- VIC version has color and enhanced sound (req. 3K memory expander)
- Comprehensive 50 - page manual

	U.S. & CANADA	FOREIGN
VIGIL for PET/VIC w/9 games on cassette (diskette add $3.00)	$35	$40
VIGIL User's Manual (refundable with software order)	$10	$12
VIGIL Interpreter Listing (6502 Assembler Language)	$25	$30

TINY BASIC COMPILER - PET

A true compiler that turns your BASIC program into fast machine code
- Subset of PET BASIC compiles to 6502 machine code. Reg. only 8K memory
- Has full floating point capabilities and functions
- Compiler listing available with 16K version (included)
- Can load compiled machine code anywhere in memory
- On cassette (add $3.00 for diskette)

	U.S. & CANADA	FOREIGN
TINY Basic Compiler OLD/NEW/4.0 ROMS/8032 with 8K memory	$25	$30
TINY Basic User's Manual (refundable with software order)	$ 5	$ 7

PET & APPLE II USERS TINY PASCAL

Structured language alternative to BASIC for PET or APPLE II includes:
- LINE EDITOR - creates, modifies and maintains source language
- COMPILER - converts your source to an executable P-code format
- INTERPRETER - executes compiled P-code. Features built-in TRACE
- CASE-OF, WHILE-DO, IF-THEN-ELSE, REPEAT-UNTIL, FOR-TO/DOWNTO, PROC, FUNC
- Graphics version has more: GRAPHICS, PLOT, POINT, TEXT, INKEY, ABS, SQR APPLE II has lores & hires-COLOR, HGRAPHICS, HCOLOR, HPLOT, PDL and TONE

	U.S. & CANADA	FOREIGN
TINY Pascal PLUS - GRAPHICS PET 32K NEW/4.0 ROMS/8032/diskette	$50	$55
TINY Pascal PLUS - GRAPHICS PET 32K NEW/4.0 ROMS/8032/cassette	$55	$60
TINY Pascal PLUS - GRAPHICS APPLE II 48K and DOS 3.2/3.3	$50	$55
TINY Pascal NON-GRAPHICS PET 16K/32K NEW/4.0 ROMS diskette	$35	$40
TINY Pascal NON-GRAPHICS PET 16K/32K NEW/4.0 ROMS cassette	$40	$45
TINY Pascal User's Manual (refundable with software order)	$10	$12
TINY Pascal 6502 Interpreter Listing-GRAPHICS version	$25	$30

PIPER-THE MUSIC MACHINE

Here's the simplest way yet to compose, conduct and play music on your VIC/PET. Complete music capability — play all note values, list and edit your music, easily handles repeats and rests. Written in fast machine language. Includes sample compositions and manual. VIC version does not require additional memory. PET version requires external CB2 speaker.

	U.S. & CANADA	FOREIGN
PIPER of PET/VIC on cassette (add $3.00 for diskette)	$25	$30
PIPER User's Manual (refundable with software order)	$ 5	$ 7
PIPER 6502 Assembly Listing	$25	$30

PET* MACHINE LANGUAGE GUIDE

Now in its ninth printing. Learn the hidden talents of your OLD, NEW or 4.0 ROM PET/CBM with this easy to follow manual. Details 30 of the PET's builtin routines.

	U.S & CANADA	FOREIGN
PET MACHINE LAGUAGE GUIDE for OLD, NEW or 4.0 ROMS	$9	$11

Orders prepaid in U.S. Dollars, VISA, MC, ACCESS, etc. Prices include postage.

ABACUS SOFTWARE
P. O. Box 7211
Grand Rapids, Michigan 49510
616 / 241-5510

Listing 2 *(Continued)*

```
105 '*INHERENT OPCODE MNEMONICS AND CODES
110 FOR I=0 TO 20:READ IH$(I),TM$:IH(I)=VAL("&H"+TM$):NEXT I
115 '*EXG/TFR REG CODES
120 FOR I=0 TO 15:READ EX$(I):NEXT I
125 '*PSH/PUL REG FLAGS
130 FOR I=0 TO 7:READ PP$(I):NEXT I
135 '*NON-IMMEDIATE ACCUM/MEM CODES
140 FOR I=0 TO 11:READ LT$(I),TM$:LT(I)=VAL("&H"+TM$):NEXT I
145 '*RELATIVE CODES
150 FOR I=0 TO 15:READ BT$(I),TM$:BT(I)=VAL("&H"+TM$):NEXT I
155 '*ACC 'A'/REGS;OPCODES $80-$BF
160 FOR I=0 TO 15:READ AT$(I),TM$:AT(I)=VAL("&H"+TM$):NEXT I
165 '*ACC 'B'/REGS;OPCODES >=$C0
170 FOR I=0 TO 15:READ CT$(I),TM$:CT(I)=VAL("&H"+TM$):NEXT I
175 '*OPCODES/MNEMS FOR PREFIX $10;$11 NO TABLE
180 FOR I=0 TO 5:READ PF$(I),TM$:PF(I)=VAL("&H"+TM$):NEXT I
185 '*REG CODES FOR INDEXED POST-BYTE
190 FOR I=0 TO 3:READ RG$(I):NEXT I
200 RETURN
205 '*INPUT A HEX NMBR;NR SHOWS STATUS;DB HAS TWO-BYTE HEX
210 NR=0:INPUT KY$:IF KY$="" THEN RETURN
220 TY$=LEFT$(KY$,1)
230 IF TY$>="0"AND TY$<="9" THEN NR=1:GOTO 260
240 IF TY$>="A" AND TY$<="F" THEN NR=1:GOTO 260
250 NR=-1
260 IF NR=-1 THEN RETURN ELSE KY$="0000"+KY$
270 DB=VAL("&H"+RIGHT$(KY$,4)):RETURN
275 '*PUT TWO BYTE HEX IN LN$;SUPPLIE LDG ZEROES
280 IF DB<4096 THEN LN$=LN$+"0"
290 IF DB<256 THEN LN$=LN$+"0"
300 IF DB<16 THEN LN$=LN$+"0"
310 LN$=LN$+HEX$(DB):RETURN
315 '*ONE-BYTE HEX
320 IF SB<16 THEN LN$=LN$+"0"
330 LN$=LN$+HEX$(SB):RETURN
335 '*4 HX+SPC
340 GOSUB 280:LN$=LN$+" ":RETURN
345 '*2 HX+SPC
350 GOSUB 320:LN$=LN$+" ":RETURN
355 '*PRT DISASSEMBLED LINE;LN$=NMBRS;MN$=MNEMONIC;OP$=OPND
360 EX=0:SB=OC:GOSUB 350:IF BY=1 THEN 400
365 '*OPCD FIRST; CHK MULTIBYTE INDEXED;IX IS INDX FLAG
370 IF IX<>0 AND BY>2 THEN 500
380 IF BY=2 THEN SB=PEEK(PC+1):GOSUB 320
390 IF BY=3 THEN DB=(256*PEEK(PC+1))+PEEK(PC+2):GOSUB 280
400 PRINTLN$;TAB(19)MN$;TAB(25)OP$:NL=NL-1
405 '*CHK FOR ABORT AFTER EACH LINE
410 IF INKEY$="X" THEN EX=1
420 PC=PC+BY:IF LS=0 THEN RETURN
425 '*LS='PRINTER ON' FLAG
430 PRINT#-2,LN$;TAB(20)MN$;TAB(26)OP$:LC=LC-1
440 IF INKEY$="X" THEN EX=1
450 IF LC<>0 THEN RETURN
455 '*LC=LINE COUNTER; SPACE AND PRINT HEADER
460 PRINT#-2,CHR$(10):PRINT#-2,TAB(32)PG:FOR I=1 TO 6
470 PRINT#-2,CHR$(10):NEXT I
480 PRINT #-2,"***";HD$;"***"
490 PRINT#-2,CHR$(10):PG=PG+1:LC=56:RETURN
500 SB=PEEK(PC+1):GOSUB 350
510 IF BY=3 THEN SB=PEEK(PC+2):GOSUB 320
520 IF BY=4 THEN DB=(256*PEEK(PC+2))+PEEK(PC+3):GOSUB 280
530 GOTO 400
535 '*SET PB(x) ACCORDING TO BITS IN BT; MSB=PB(7)
540 FOR I=7 TO 0 STEP -1:IF BT>255 THEN BT=BT-256
550 PB(I)=INT(BT/128):BT=BT*2:NEXT I:RETURN
555 '*TWO BYTE OFFSET;SIGNED
560 BT=(256*PEEK(PC+BY-2))+PEEK(PC+BY-1):GOTO 580
565 '*ONE BYTE OFFSET;SIGNED
570 BT=PEEK(PC+BY-1):GOTO 610
580 OS=BT:SN=0:IF BT>&H7FFF THEN SN=1:OS=BT-&H8000
590 IF SN<>0 THEN OS=OS-&H8000
600 RETURN
610 OS=BT:SN=0:IF BT>127 THEN SN=1:OS=OS-128
620 IF SN<>0 THEN OS=OS-128
630 RETURN
635 '*5 BIT OFFSET;SIGNED
640 OS=BT:SN=0:IF BT>15 THEN SN=1:OS=OS-16
650 IF SN<>0 THEN OS=OS-16
660 RETURN
665 '*COMPUTE TARGET; MAKE IT OPERAND
670 BT=PC+BY+OS:OP$="$":GOTO 689
675 '*MAKE TWO BYTE OPERAND(EXTENDED)
680 BT=(256*PEEK(PC+BY-2))+PEEK(PC+BY-1):OP$=OP$+"$"
689 IF BT<0 THEN BT=BT+65536:'MAKE POSITIVE HEX
690 IF BT<4096 THEN OP$=OP$+"0"
700 IF BT<256 THEN OP$=OP$+"0"
```

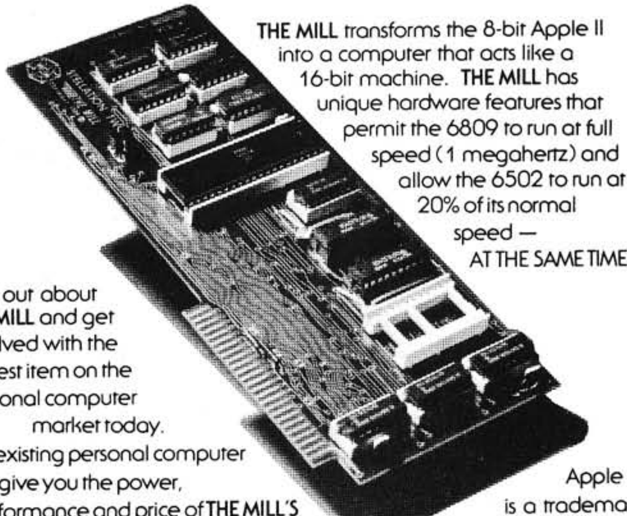
(Continued)

Listing 2 *(Continued)*

```
710 IF BT<16  THEN OP$=OP$+"0"
720 OP$=OP$+HEX$(BT):RETURN
730 OP$=OP$+"$":BT=PEEK(PC+BY-1):GOTO 710
735 '*PRTR INIT PARAMS FOR OKIDATA-80;
    SETS 64 COLMN,10 CPI
740 OPEN"O",-2,CHR$(13):PRINT#-2,CHR$(&H1B);"B"
750 LC=56:PG=0:LS=1
760 PRINT:INPUT " NAME";HD$:GOSUB 460
770 CLS:PRINT"'X' ABORTS LISTING"
780 PRINT"SHIFT '@' SUSPENDS LISTING"
   :PRINT"enter RESUMES"
790 RETURN
795 '**DISASSEMBLER MENU;'X' RETURNS TO PREVIOUS STEP
796 '  'Z' RETURNS TO START
800 CLS:PRINT:PRINT"DISASSEMBLER":EX=0
810 LS=0:INPUT "PRINTED LISTING (Y/N)";KY$
820 IF KY$="Y" THEN GOSUB 740:GOTO 850
830 IF KY$="X" THEN 800
840 IF KY$="Z" THEN 20
850 EX=0:PRINT:PRINT"BEGIN ADDRESS:";:GOSUB210
860 IF NR<>1 THEN 830
870 PC=DB
880 PRINT:PRINT"NMBR LINES";:INPUT KY$
   :IF KY$="X" THEN 800
890 IF KY$="Z" THEN 20
900 NL=VAL(KY$)
910 IF NL=0 THEN NL=1
920 GOTO 970
930 '**ERRPRT
940 MN$="????":BY=1:GOSUB 360:GOTO 970
950 '**NORMPRT
960 GOSUB 360:'**FALL THRU
965 '**ACTUAL DISASSEMBLE RTN
970 IF NL=0 THEN 880 ELSE IX=0:PF=0:RF=0:BY=0
   :RN=0:BT=0
975 '**SET VARIABLES; CHECK FOR ABORTED LISTING
980 IF EX=1 THEN 850
990 LN$="":MN$="":OP$=""
995 '**PRT PGM CNTR; GET OPCODE
1000 DB=PC:GOSUB 340:OC=PEEK(PC)
1005 '*SEE IF PREFIXED FIRST
1010 IFOC=&H10 OR OC=&H11 THEN 1350
1020 LN$=LN$+"  "
1025 '*SWI GETS SPECIAL HANDLING
1030 IF OC=&H3F THEN 1320
1035 '*LBSR/LBRA LIKEWISE; 'BSR' HAS AMBIGUOUS OPCD
1040 IF OC=&H16 OR OC=&H17 THEN 1510
1050 IF OC=&H8D THEN MN$="BSR":GOTO 1570
1055 '*ROUTE BRANCHES
1060 IF OC>&H1F ANDOC <&H30 THEN 1530
1065 '*ROUTE ACCUM/REGS(IMDT)
1070 IF OC>127 THEN 1730
1075 '*CHECK INHERENTS
1080 F=0:FOR I=0 TO 20:IF OC<>IH(I) THEN 1100
1090 F=1:MN$=IH$(I):GOTO 1110
1100 NEXT I
1105 '*NOT FND;GO CHK LOTABL; IF FOUND,
     CHK EXCEPTIONS
1110 IF F=0 THEN 1610
1115 '*WEED LEAx TO INDEX MODE
1120 IF OC>=&H30 AND OC<=&H33 THEN 1870
1125 '*WEED PSH/PUL
1130 IF OC>&H33 AND OC<&H38 THEN 1190
1135 '*WEED COND CODE OPS
1140 IF OC=&H3C OR OC=&H1A OR OC=&H1C THEN 1230
1145 '*TFR/EXG OPS
1150 IF OC=&H1E OR OC=&H1F THEN 1170
1155 '*PRT ONE-BYTE INHERENT
1160 BY=1:GOTO 960
1165 '**TFR/EXG; LFT NIBL='FROM' ; RT NIBL='TO'
     ; EX$(x)=REG
1170 BT=PEEK(PC+1):LN=INT(BT/16):RN=BT-(LN*16)
1180 OP$=EX$(LN)+","+EX$(RN):BY=2:GOTO 960
1185 '*PSH/PUL-PP$=REG; PB(x)=BIT IN POSTBYTE
1190 BT=PEEK(PC+1):GOSUB 540:FOR I=0 TO 7
1200 IF PB(I)=0 THEN 1210 ELSE OP$=OP$+PP$(I)+","
1210 NEXT I:MID$(OP$,LEN(OP$),1)=" "
1220 BY=2:GOTO 960
1225 '*SET/CLR BITS IN COND CODE REG
    ;CC$ HAS FLAG NAMES
```

```
1226 '*LINE 730 SET PB(x)
1230 BT=PEEK(PC+1):GOSUB 540:OP$="#":BY=2:GOSUB 730
1240 IF OC=&H1A THEN 1290
1245 '*ANDC,CWAI CLEAR FLAGS
1250 OP$=OP$+" (CLR ":FOR I=0 TO 7
1260 IF PB(I)=0 THEN OP$=OP$+CC$(I)+","
1270 NEXT I
1280 MID$(OP$,LEN(OP$),1)=")":GOTO 960
1285 '*ORCC SETS FLAGS
1290 OP$=OP$+" (SET ":FOR I=0 TO 7
1300 IF PB(I)=1 THEN OP$=OP$+CC$(I)+","
1310 NEXT I:GOTO 1280
1315 '*MN$ SET FOR PROPER 'SWI' BY PREFIX
1320 IF PF=0 THEN MN$="SWI":GOTO 1160
1330 IF PF=&H10 THEN MN$="SWI2":GOTO 1160
1340 IF PF=&H11 THEN MN$="SWI3":GOTO 1160
1345 '**PREFIX OPS;SET PF,GET OPCODE
1350 PF=OC:SB=OC:GOSUB 320:PC=PC+1:OC=PEEK(PC)
1355 '*CHK SWI
1360 IF OC=&H3F THEN 1320
1365 '*CHK REG/ACCUM OPCDS
1370 IF OC>127 THEN 1400
1375 '*CHK LONG REL BRANCH
1380 IF OC>&H20 AND OC<&H30 THEN 1530
1385 '*INVALID OPCD
1390 GOTO 940
1395 '*SET UP TO SEARCH PF TABLE
1396 ' USE DUMMY IMMEDIATE OPCODE
1400 SC=0:LN=INT(OC/16):RN=OC-(16*LN):SC=&H80+RN
1405 '*WEED PAGE 2 CODES
1410 IF PF=&H11 THEN 1480

1415 '*ALTER DUMMY IF NEEDED
1420 IF LN>11 THEN SC=SC+&H40
1425 '*ONLY 6 PAGE 1 CODES
1430 FOR I=0 TO 5:IF SC<>PF(I)
     THEN 1470 ELSE MN$=PF$(I)
1440 IF LN>11 THEN LN=LN-4
1445 '*WEED 'STORE IMMEDIATE'
1450 IF OC=&H9F OR OC=&H8F THEN 940
1455 '*PROCESS ADDRESS MODE
1460 GOTO1780
1470 NEXT I:GOTO 940
1475 'PAGE 2 OPCDS
1480 IF RN=3 THEN MN$="CMPU":SC=1
1490 IF RN=12 THEN MN$="CMPS":SC=1
1500 IF SC=1 THEN 1440 ELSE 940
1505 '*RELATIVE BRANCHES
1510 BY=3:IF OC=&H17 THEN MN$="LBSR  ":GOTO 1600
1520 MN$="LBRA  ":GOTO 1600
1530 F=0:FOR I=0 TO 15
1540 IF BT(I)<>OC THEN 1560 ELSE MN$=MN$+BT$(I)+" "
1550 F=1:GOTO 1570
1560 NEXT I
1565 '*INSERT 'L' IF LONG BRANCH
1570 BY=2:IF PF=&H10 THEN BY=3:GOTO 1590
1580 GOSUB 570:GOSUB 670:GOTO 960
1590 MN$="L"+MN$
1600 GOSUB 560:GOSUB 670:GOTO960
1605 '**LOTBL-OTHER OPCODES <$80;USES RT NIBL AS KEY
1606 '  LFT NIBL FOR ADDRESS MODE
1610 LN=INT(OC/16):RN=OC-(LN*16)
1620 IF OC=&H4E OR OC=&H5E THEN 940
1630 F=0:FOR I=0 TO 11:IF RN<>LT(I) THEN 1650
1640 MN$=LT$(I):F=1:GOTO 1660
1650 NEXT I:IF F=0 THEN 940
1655 '*LFT NIBL=4/5 FOR INHERENT
1660 IF LN=4 THEN MN$=MN$+"A":BY=1:GOTO 960
1670 IF LN=5 THEN MN$=MN$+"B":BY=1:GOTO 960
1680 IF LN<>0 THEN 1700
1685 '*LN=0 FOR DIRECT
1690 BY=2:GOSUB 730:GOTO 960
1700 IF LN<>7 THEN 1870
1705 '*LN=7 FOR EXTENDED; LN=6 FOR INDEXED
1710 BY=3:GOSUB 680:GOTO960
1720 GOTO 940
1725 '*ACCUM/REGS; NO PREFIX. LN FOR A/B TEST
     AND MODE
1726 ' RN USED FOR KEY
1730 LN=INT(OC/16):RN=OC-(LN*16)
1740 IF LN>11 THEN 1840
1745 '*'A' ACCUM
1750 FOR I=0 TO 15:IF RN<>AT(I) THEN 1770
1760 MN$=AT$(I):GOTO 1780
1770 NEXT I:GOTO 940
```

(Continued)



OSI TRS-80 COLOR-80 OSI

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 OSI

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. OSI

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) 4 back issues and subscription for 2 additional issues - $9.00.

ADVENTURES!!!

For OSI, TRS-80, and COLOR-80. These Adventures are written in BASIC, are full featured, fast action, full plotted adventures that take 30-50 hours to play. (Adventures are interactive fantasies. It's like reading a book except that you are the main character as you give the computer commands like "Look in the Coffin" and "Light the torch".)

Adventures require 8K on an OSI and 16K on COLOR-80 and TRS-80. They sell for $14.95 each.

ESCAPE FROM MARS (by Rodger Olsen)

This ADVENTURE takes place on the RED PLANT. You'll have to explore a Martian city and deal with possibly hostile aliens to survive this one. A good first adventure.

PYRAMID (by Rodger Olsen)

This is our most challenging ADVENTURE. It is a treasure hunt in a pyramid full of problems. Exciting and tough!

TREK ADVENTURE (by Bob Retelle)

This one takes place aboard a familiar starship. The crew has left for good reasons - but they forgot to take you, and now you are in deep trouble.

DEATH SHIP (by Rodger Olsen)

Our first and original ADVENTURE, this one takes place aboard a cruise ship - but it ain't the Love Boat.

VAMPIRE CASTLE (by Mike Bassman)

This is a contest between you and old Drac - and it's getting a little dark outside. $14.95 each.

OSI NEW-NEW-NEW TINY COMPILER OSI

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 OSI

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 (5¼") OSI

BARE BOARDS FOR OSI C1P

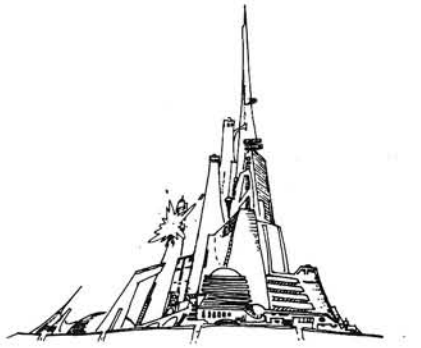
MEMORY BOARDS!!! — for the C1P — and they contain parallel ports!

Aardvarks new memory board supports 8K of 2114's and has provision for a PIA to give a parallel ports! It sells as a bare board for $29.95. When assembled, the board plugs into the expansion connector on the 600 board. Available now!

PROM BURNER FOR THE C1P — Burns single supply 2716's. Bare board — $24.95.

MOTHER BOARD — Expand your expansion connector from one to five connectors or use it to adapt our C1P boards to your C4/8P. - $14.95.

16K RAM BOARD FOR C1P — This one does not have a parallel port, but it does support 16K of 2114's. Bare Board $39.95.



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 capabibilities. 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 (0S65D 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. Specify 5¼ or 8" disk.

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 OSI

ARCADE GAMES FOR OSI, COLOR-80 AND TRS-80 (8K OSI, 16K TRS-80 AND COLOR-80)

TIMETREK - A REAL TIME, REAL GRAPHICS STARTRECK. See your torpedoes hit and watch your instruments work in real time. No more unrealistic scrolling displays! $14.95.

STARFIGHTER - This one man space war game pits you against spacecruisers, battlewagons, and one man fighters, you have the view from your cockpit window, a real time working instrument panel, and your wits. Another real time goody. $9.95

BATTLEFLEET - This grown up version of Battleship is the toughest thinking game available on OSI or 80 computers. There is no luck involved as you seek out the computers hidden fleet. A topographical toughie. $9.95

QUEST - A NEW IDEA IN ADVENTURE GAMES! Different from all the others, Quest is played on a computer generated mape of Alesia. Your job is to gather men and supplies by combbat, bargaining, exploration of ruins and temples and outright banditry. When your force is strong enough, you attack the Citadel of Moorlock in a life or death battle to the finish. Playable in 2 to 5 hours, this one is different every time.
16K COLOR-80 OR TRS-80 ONLY. $14.95

Please specify system on all orders

This is only a partial listing of what we have to offer. We offer over 120 games, ROMS, and data sheets for OSI systems and many games and utilities for COLOR-80 and TRS-80. Send $1.00 for our catalog.

AARDVARK TECHNICAL SERVICES, LTD.
2352 S. Commerce, Walled Lake, MI 48088
(313) 669-3110

OSI **COLOR-80**

Listing 2 *(Continued)*

```
1775 '*IMMEDIATE MODE. 2 OR 3 BYTES?
1780 IF LN=8 THEN OP$="#"+OP$ ELSE 1810
1790 IF RN=3 OR RN=12 OR RN=14 THEN BY=3
     :GOSUB 680:GOTO  960
1800 BY=2:GOSUB 730:GOTO 960
1805 '*DIRECT MODE
1810 IF LN=9 THEN BY=2:GOSUB 730:GOTO 960
1820 IF LN<>&HB THEN 1870
1825 '*EXTENDED MODE
1830 BY=3:GOSUB 680:GOTO 960
1835 '*'B' ACCUM
1840 FOR I=0 TO 15:IF RN<>CT(I) THEN 1860
1850 MN$=CT$(I):LN=LN-4:GOTO 1780
1860 NEXT I:GOTO 940
1865 '*INDEXED MODE; POSTBYTE TELLS ALL!
1866 '  SET PB(x) ACCDNG TO BITS IN POSTBYTE
1870 IX=1:PB=PEEK(PC+1):BT=PB:GOSUB 540
1880 LN=INT(PB/16):RN=PB-(LN*16)
1885 '*LSR LFT NIBL;ERASE INDIRECT TO MAKE REG CODE
1890 RG=INT(LN/2):IF RG>3 THEN RG=RG-4
1895 '*GET REG NAME
1900 RG$=RG$(RG)
1905 '*POSTBYTE HAS +,- 5 BIT OFFSET IF POSITIVE
1910 IF PB>127 THEN 1940
1915 '* FIVE BIT OFFSET, NO INDIRECT
1920 BT=PB-(RG*32):GOSUB 640
1930 OP$=STR$(OS)+","+RG$:BY=2:GOTO 960
1935 '*ROUTE OPS;TWO LINES USED FOR CLARITY
     (16 NMBRS ALLOWED)
1940 IF RN>7 THEN RN=RN-7:GOTO 1960
1945 '*RT NIBL HAS CODE FOR MODE/REG INDEXED
1950 ON RN+1 GOTO 2010,2020,2030,2040,2000,2050,
     2060,940
1960 ON RN GOTO 2070,2090,940,2100,2110,2130,940,1970
1965 '*INDIRECT EXTENDED
1970 BY=4:GOSUB 680
1975 '*CHK INDIRECT FLAG
1980 IF PB(4)=1 THEN OP$="("+OP$+")"
1990 GOTO 960
1995 '*ZERO OFFSET
2000 BY=2:OP$=","+RG$:GOTO 1980
2010 BY=2:OP$=","+RG$+"+":GOTO 1980
2020 BY=2:OP$=","+RG$+"++":GOTO 1980
2030 BY=2:OP$=",-"+RG$:GOTO 1980
2040 BY=2:OP$=",--"+RG$:GOTO 1980
2050 BY=2:OP$="'B',"+RG$:GOTO 1980
2060 BY=2:OP$="'A',"+RG$:GOTO 1980
2065 '*8 BIT OFFSET
2070 BY=3:GOSUB 570
2080 OP$=STR$(OS)+","+RG$:GOTO 1980
2085 '*16 BIT OFFSET
2090 BY=4:GOSUB 560:GOTO 2080
2100 BY=2:OP$="'D',"+RG$:GOTO 1980
2105 '*8 BIT OFFSET;PGM CNTR RELATIVE
2110 BY=3:GOSUB 570:GOSUB 670
2120 OP$=OP$+",PCR":GOTO 1980
2125 '*16 BIT OFFDET; PC RELATIVE
2130 BY=4:GOSUB 560:GOSUB 670:GOTO 2120
2140 DATA C,V,Z,N,I,H,F,E
2150 DATA NOP,12,MUL,3D,ABX,3A,CWAI,3C,DAA,19,EXG,
     1E,PSHS,34
2160 DATA PSHU,36,PULS,35,PULU,37,RTI,3B,RTS,
      39,SEX,1D
2170 DATA SYNC,13,TFR,1F
2180 DATA LEAX,30,LEAU,33,LEAS,32,LEAY,31,ANDC,1C,
     ORCC,1A
2190 DATA D,X,Y,U,S,PC,?,?,A,B,CC,DP,?,?,?,?
2200 DATA CC,A,B,DP,X,Y,S/U,PC
2210 DATA ASL,8,ASR,7,CLR,F,COM,3,DEC,A,INC,C,LSR,4
2220 DATA JMP,E,NEG,0,ROL,9,ROR,6,TST,D
2230 DATA BCC,24,BCS,25,BEQ,27,BGT,2E,BHI,22,BLE,2F
2240 DATA BLS,23,BLT,2D,BMI,2B,BNE,26,BPL,2A,BRA,20
2250 DATA BRN,21,BVC,28,BVS,29,BGE,2C
2260 DATA STA,7,SUBA,0,CMPA,1,SBCA,2,SUBD,3,ANDA,4
2270 DATA BITA,5,JSR,D,LDA,6,EORA,8,ADCA,9,ORA,A
2280 DATA ADDA,B,CMPX,C,LDX,E,STX,F
2290 DATA STB,7,SUBB,0,CMPB,1,SBCB,2,ADDD,3,ANDB,4
2300 DATA BITB,5,LDB,6,EORB,8,ADCB,9,ORB,A
2310 DATA ADDB,B,LDD,C,STD,D,LDU,E,STU,F
2320 DATA CMPD,83,CMPY,8C,LDY,8E,STY,8F,LDS,CE,STS,CF
2330 DATA X,Y,U,S
2340 '***HEXDUMP
2350 EX=0:LN$="":CLS:PRINT:PRINT"HEXDUMP OPTIONS"
2360 LL=8:PRINT"LINE LENGTH: S=32,L=64";:INPUT KY$
2370 IF KY$="X" THEN 2350
2380 IF KY$="Z" THEN 20
2390 IF KY$="L" THEN LL=16
2400 PRINT"ASCII: N=NORMAL;C=CLR B7";:INPUT KY$
2405 '*CF=1 IGNORES BIT 7 IN ASCII PRINT
2410 IF KY$="C" THEN CF=1 ELSE CF=0
2420 IF KY$="X" THEN 2360
2430 IF KY$="Z" THEN 20
2440 LS=0:INPUT" PRINTED LISTING? (Y/N)";KY$
2450 IF KY$="Y" GOSUB740
2460 PRINT:PRINT"ADDRESS";:GOSUB 210
     :IF KY$="X" THEN 2350
2470 IF KY$="Z" THEN 20
2480 IF NR<>1 THEN 2460
2490 PC=DB
2500 INPUT "NUMBER OF LINES";NL$:IF NL$="X"
     THEN 2460
2510 IF NL$="Z" THEN 20
2520 NL=VAL(NL$):IFNL=0 THEN NL=1
2525 '*SET UP AND PRINT HEX
2530 IF NL=0 THEN 2500 ELSE LN$="":DB=PC:GOSUB 340
2540 FOR I=PC TO PC+LL-1
2550 SB=PEEK(I):GOSUB 350:NEXT I:PRINT LN$
2555 '*CHECK ABORT AFTER EACH LINE
2560 IF INKEY$="X" THEN 2460
2570 IF LS=0 THEN 2590
2580 PRINT#-2,LN$
2585 '*BLANK BELOW PC;PRINT ASCII
2590 LN$="       ":FOR I=PC TO PC+LL-1
2600 LN$=LN$+" ":BT=PEEK(I):IF CF=0 THEN 2620
2610 IF BT<128 THEN 2620 ELSE BT=BT-128
2620 IF BT<&H20 OR BT>&H5F GOTO 2640
2630 LN$=LN$+CHR$(BT)+" ":GOTO 2650
2640 LN$=LN$+". "
2650 NEXT I:PRINT LN$:NL=NL-1
2660 IF INKEY$="X" THEN 2460
2670 PC=PC+LL:IF LS=0 THEN 2530
2675 '*LINES ARE COUNTED BY TWO'S
2680 PRINT#-2,LN$:LC=LC-2:IF LC<>0 THEN 2530
2690 GOSUB 460:GOTO 2530
```

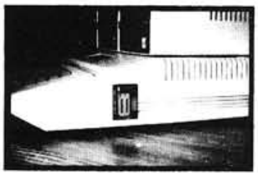
MICRO

VersaWriter & APPLE II:
The Keys to Unlimited Graphics

DRAWING TABLET
Although VersaWriter operates on a simple principle, it produces graphics which match or exceed those of other digitizers. Rugged construction, translucent base, easy to use — plugs directly into APPLE II.

GRAPHICS SOFTWARE
Easily the most capable and complete graphics software for the home computer available. Fast fill drawings in 100 colors. All text in five sizes, compile and display shapes, edit, move and much more!

UNIQUE OFFER
See VersaWriter at your local dealer and pick up a copy of our demonstration disk. The complete VersaWriter hardware and software package is a real bargain at $299. For more information call or write:



EZ Port Will Solve Your Game I/O Problem!

How many times have you gone through the hassle of changing from game paddles to joystick, VersaWriter, or any other device using the game I/O? First, you have to remove whatever is sitting on top of the Apple–a video terminal, disk drives, printer, etc.

Next you remove the computer cover and try to see what you're doing as you switch plugs to the I/O. Then you replace the computer cover and whatever was on top of the Apple.

After all this, you find that you can't run the program because the I/O device is plugged in backwards or is 'off by a pin'.

Sound familiar?

EZ PORT GAME I/O EXTENDER FOR APPLE II

WHAT IS EZ PORT?

EZ Port is a specially designed extension unit for the Apple game I/O port. It's a board with a socket and a two foot long cable which plugs into the internal I/O port. You attach EZ Port wherever you prefer on the outside-on the side, the back, or on top.

EZ Port has a ZIP DIP II socket (ZIP=zero insertion force). These sockets are meant to be plugged into many times and will not wear out like ordinary sockets. All you do is plug in the appropriate device (joystick, paddles, etc.) and flip the switch to the ON position. No pressure is exerted on the 16-pin plug until you switch, so all the connectors will last longer, too!

EZ PORT MAKES GAME I/O CHANGES CONVENIENT, QUICK & SAFE.

Suggested price **$24.95**
Ask your local computer retailer for EZ Port, or contact:

VersaComputing, Inc.
3541 Old Conejo Rd. Suite 104
Newbury Park, CA 91320
(805) 498-1956

Dealers inquiries welcome.

Apple and Apple II are registered trademarks of Apple Computer, Inc.

ZIP DIP is a trademark of Textool Products, Inc.

Versa Computing, Inc.

3541 Old Conejo Road, Suite 104
Newbury Park, CA. 91320 **(805)498-1956**

Now You Have "The Choice"

Reliability, Quality and *Performance* at a reasonable *Price. Systems International, Inc.* is pleased to offer an alternate to *Ohio Scientific* microcomputers. Our systems are 100% compatible to *OSI OS-65D* and *OS-65U* software to preserve your software development investment. Just load your present floppies and run; no conversion of software needed or required. *Compare the Following Facts and Make "The Choice."*

Ohio Scientific, Incorporated **Standard Features** **C2-OEM**	**Systems International, Incorporated** **Standard Features** **The Choice II**
48K Memory, 1 MHz	48K Memory, 2 MHz
One Serial Port	2 Serial Ports, 1 Parallel Port
Dual 8" Single Sided Floppies	Dual 8" Double Sided Floppies
Plug in Boards with Many Connections	Single Board Construction for Reliability
120 Volt 60, Hz Operation Only	120/240 Volt 50/60 Hz Operation
90-Day Warranty	180-Day Warranty
Weight 80 Pounds	Weight 40 Pounds
Size W17" × H9½" × D23½"	Size W12½" × H13½" × D16"
	Shippable by UPS in Factory Carton
Extra Cost Items	**Extra Cost Items**
Second Serial Port Parallel Port 2 MHz	*None*

Now Compare the Bottom Line and Make "The Choice"

Total Retail $4,925.00 *	**Total Retail $4,525.00** **240 Volt 50 Hz Operation add $50.00**

Future plans include the *"Choice III"* which is 100% compatible to the OSI C2-D 8" Winchester System. Also Multi-user Winchester Hard Disk System that is 100% compatible with Ohio Scientific OS-65U Level 3 Software.

Remember — There is "The Choice" to choose a better system!

Domestic and International Dealer and Distributor Inquiries are Invited. Discounts to 40%

* Ohio Scientific Price List June/July 1981

Systems International Incorporated

15920 Luanne Drive
Gaithersburg, Maryland 20760
U.S.A.
Tel. (301) 977-0100 Twx# 710-828-9703

Sii

500 Chesham House
150 Regent Street
London W1R 5.F.A England
Tel. 01-439-6288 Tlx 261426

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

Formatting AIM Assembler Listings: A PL/65 Approach

This program, developed with Rockwell's PL/65 compiler, reformats AIM assembler listings. The new listings are much easier to read than the standard 20-column assembler format.

Christopher J. Flynn
2601 Claxton Drive
Herndon, Virginia 22071

Have you assembled and printed a 50-line program with the AIM's ROM assembler? Works great, doesn't it? But how about a 100-line program or a 500-line program? As you know, the assembler listings can get pretty hard to read.

Here is a program that will run on any AIM. It will make those tiny, cramped, assembler listings very easy to read. But, you'll need to beg, borrow, or buy a printer or Teletype to connect to your AIM.

What's the Problem?

First of all, I think the AIM assembler is terrific. When you consider that it all fits in a 4K ROM you've got to be impressed. Remember, though, that the assembler has to talk to a 20-column printer. The assembler's designers had to make some tough decisions. How were they going to fit all the characters that an assembler normally produces on a 20-column line? Let's see what they did.

Figure 1A shows a typical line of an assembly language program. The line contains a label, an opcode, an operand, and a comment. Run this through the assembler and see what happens. Look at figure 1B. Our single line of source code produced three lines of printed output! The AIM assembler has a way of turning modest source listings into lengthy strips of thermal paper.

Figure 1 *A:* Typical line of an assembly language program.

```
CRLF LDA #$0D ; SEND CR
```

Figure 1 *B:* Listing produced by AIM assembler.

```
= =0F00 CRLF .
A90D LDA #$0D
;SEND CR
```

Fixing the Problem

My first attempt at fixing the problem was based on wishful thinking. What if the AIM assembler really didn't produce multiple output lines for most of the input lines? What if, in the case above, the output line were really 60 characters long? AIM's printer would go to a new line every 20 characters giving the appearance of three lines. So, I hooked up my printer and hoped for the best. But, I still ended up with a lot of short lines.

Next, I tried directing the AIM assembler to tape. I wrote a short BASIC program that read the tape and produced a nicely formatted output. There were two things wrong with this approach, however. First, a BASIC program had to be loaded and run every time an assembler listing was needed. Second, in my version of the AIM assembler at least, there is a bug which prevents the tape ouput file from being closed properly. This results in the loss of the last few lines of the listing.

This little experiment prompted me to develop a solution that finally worked. How about a user output routine? If we could trap each output line before it were printed, we could then decide how to reformat it. Then we could send the reformatted line on its way to be printed. Sounds simple enough, doesn't it?

Study figure 2 very carefully. It contains, in tabular format, the specifications for the user output routine. I'll just go over the highlights.

Based on the first character of the ouput line, the line is categorized as one of four types. It can be an address line, a comment line, a page eject or title line, or lastly, an object code line. Next, the output routine decides what horizontal tab positions to use. This lets us, for example, line up the object code of each assembler instruction in the same place each time. Depending on the type of line and the current tab position, a carriage return-line feed (CRLF) sequence may be required before actually printing the line. Again, depending on the type of line, we may or may not issue a CRLF after printing the line.

There are certain other little subtleties that we must consider. For example, suppose you are one of those programmers with a habit of commenting

Figure 2

Specifications
Assembler Reformatter

First Character Of Line	Type of Line	Tab Position	CRLF Before Printing	CRLF After Printing	Notes
=	Address	1	TAB>0	No	
;	Comment	TABINS or TABCMT	TAB> TMPCMT	Yes	Comments can continue on next line.
_ ($5F)	Page Eject	N/A	N/A	N/A	Not printed. Skip to new page.
Anything Else	Object Code	TABINS	TAB > TABINS	No	Print space after object code.

your source code. The output routine will line up the comments with either the object code or near the middle of the page. (We'll get into that option later.) What happens if we encounter a long comment line? If you're using a 132-column printer, not much happens. Most likely, though, you have a 72- or 80-column printer. So, an additional requirement is that we gracefully handle long comment lines by continuing them in the proper position on a new line.

The key to getting pretty AIM assembler listings hinges upon our being able to take advantage of the assembler's seemingly peculiar behavior. As the assembler produces each output line, we intercept the line in a user output routine. Then, at our leisure, we can decide exactly where and how we want to print the line.

Writing the Program

Perhaps you were expecting a 100-byte position-independent ROMable machine language program? Sorry to disappoint you. Take a look at listing 1. That strange looking program is written in a language called PL/65. And that strange looking program is the user output routine.

PL/65 is a language with its own set of syntax rules. Rockwell markets a two-chip PL/65 ROM compiler for the AIM 65. The PL/65 compiler translates PL/65 programs into assembly language source code rather than machine code. We have to pass the generated assembly code through the AIM assembler to end up with executable machine code. If you are interested in PL/65 programming, make sure you get both the PL/65 ROMs and the assembler ROM. (Before I frighten readers away, the assembler reformatter can be used without PL/65.)

Look at listing 1 again. PL/65 probably looks like a language you already know — PL/1, ALGOL, or Pascal. But as you study the listing, you'll notice how closely the language is tailored to the 6502. Do you see how the data declarations and definitions resemble what we do in assembly language? Look at how assembly language statements have been included directly in the PL/65 source code. PL/65 seems to offer the advantages of modern programming languages while still allowing us to take advantage of the CPU architecture.

I won't attempt to go over the code line by line but there are several important points to note. First of all, there are two assembly language JMP instructions at the very beginning of the program. These are vectors for device-dependent initialization and character output routines, respectively. Secondly, notice that the program will be entered at ASMOUT each time. Here we test the carry flag to determine if the output routine is being called for the first time or not. With that out of the way, the bulk of the program is almost self-explanatory. Just keep figure 2 handy. The last 20 lines contain the assembly language code needed to drive my printer.

Loading the Program

The PL/65 ROMs are not required to use the asssembler reformatter. Listing 2 shows the hex dump of the reformatter's machine code. You'll have to key the machine code in by hand using the AIM monitor. Begin entering the machine code at $0200. The program code continues up through and including $0438. An 80-byte line buffer occupies locations $0439 through $0488. Device-dependent code begins at $0489. I am using an RS-232 printer with my AIM. See the attached text box for a description of how I got the two to talk to each other.

After you've finished entering the machine code into your AIM, you'll probably want to do a little tinkering with the program. Look at figure 3. It contains the addresses of critical routines and constants.

For the assembler reformatter to work properly, you must supply two of your own subroutines to interface to the particular printer that you are using. If you have a 1200-baud RS-232 printer with handshaking, you can use the same subroutines that I am using.

The first subroutine should contain any initialization logic that your printer may require. Since I am using the AIM's serial port, my initialization logic consists of setting two AIM monitor locations ($A417 and $A418) for 1200 baud as described in Section 9.2.3 of the *AIM 65 User's Guide*. An RTS instruction should be the last instruction in the initialization subroutine.

The second subroutine is a character output subroutine. It should transmit the contents of the accumulator to your printer. In my case, I first check the printer's handshaking line. When the printer is ready, I output the character by means of the AIM monitor serial output routine at $EEA8. The last two instructions of the character output routine must be:

```
E6 F7  INC TAB
60     RTS
```

These subroutines can be located wherever it's most convenient for you. Just be sure to modify the two vectors at $0200 and $0203 accordingly.

The various constants shown in figure 3 let you format assembly listings almost any way that you want. There's one little trick, though, that I'd better let you know about. I like to begin printing in column 6 to allow me to punch holes for a three ring binder. As you can see from the program listing, I tab over 5 spaces every time I do a CRLF. (If you do not want this feature, just set location $03FC to $00.) Notice from listing 1 that TAB gets reset to zero even after we've tabbed over these five spaces. Just keep this in mind if you decide to alter any other tab settings.

Using the Reformatter

The reformatter is really very easy to use. Suppose you're ready for a hard copy listing of your assembly language program. The assembler will ask you "LIST?". Respond "Y". You will be asked for the output device "LIST-OUT=". Respond "U". Don't forget to set the AIM user output vector at $010A to $0206. Otherwise you'll have to start all over again.

When the assembler asks you "OBJ?", be sure you answer "Y". If you respond "N", you may see some strange things happening to your listing. Respond appropriately to the "OBJ-OUT=" prompt. Don't forget that you can suppress object code generation by answering "X".

Listing 3 shows what the reformatter can do with some sample source code input. Notice how the comments are printed. Sometimes they line up with the object code, other times they print to the right of the operand field. This feature is triggered by coding a special command in the assembly language source code itself. Suppose you have a section of code that you want to document with comments. In front of these comments place a line containing just a semi-colon and a carriage return. When the reformatter encounters this line, it will line it, and any comment lines that immediately follow it, with the object code.

I caution you to be careful in your use of address space. As presented, the reformatter occupies locations $0200 through $049E. Please don't try to store your source program, assembler object code, or symbol table in this area.

What's Next?

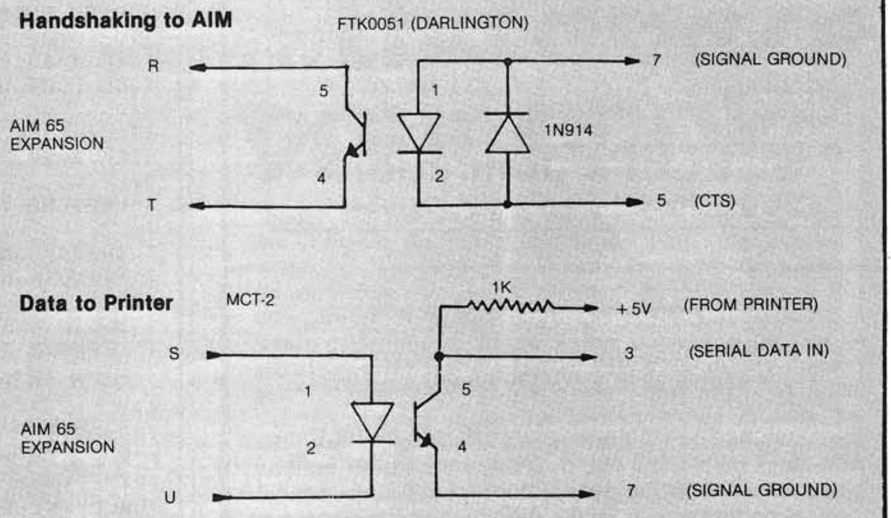
I hope that you'll agree that the reformatted assembly listings are very easy to read and work with. But, perhaps you would like to carry this idea a little farther. If we sequence-numbered each line, printed the address of each instruction, and interchanged the label and object code fields, our listings would be indistinguishable from those of other assemblers. Or, we could print a sorted symbol table.

Christopher Flynn owns an AIM with 32K of RAM. His software interests include assembly language and BASIC, and he is beginning to experiment with fig-FORTH. Flynn is employed by the Fairfax County government as a systems analyst for the county's tax systems.

Figure 3 Addresses
Assembler Reformatter

Item	Description	Address	Default Value
1	JMP to device initialization	$0200	4C 89 04
2	JMP to device output	$0203	4C 94 04
3	Limit of comment line	$02CD	$47 (71)
4	Lines per page (occurs twice)	$0320	$37 (55)
5		$03CD	$37 (55)
6	No. lines skipped between pages	$03DE	$0A (10)
7	Starting tab of each new line	$03FC	$05 (5)
8	Instruction field tab	$0436	$0F (15)
9	Comment field tab (occurs twice)	$0437	$28 (40)
10		$0438	$28 (40)
11	Device initialization code	$0489	---
12	Device output routine	$0494	---
13	Start of AIM text buffer	$049F	---

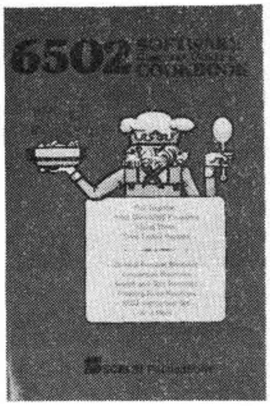
Connecting an RS-232 Printer to the AIM



To use an RS-232 printer with the AIM, you'll need to convert the AIM's built-in 20 mA loop to the proper RS-232 levels. Application Note No. 8 from Rockwell shows one way of doing this. You might like to try the circuit that I use. Since it does not generate true RS-232 signals, it is not guaranteed to work with all printers. It works just fine with an Integral Data Systems IP-125, though.

Next, you'll need to think about software. The AIM monitor has a serial output routine called OUTTTY, located at address $EEA8. It will work with most common baud rates. However, you must set locations $A417 and $A418 with the proper timing constants *before* you call OUTTTY for the first time. The timing constants are listed in Section 9.2.3 of the *AIM 65 User's Guide.*

If you plan to send data to your printer at a rate faster than 300 baud, then you'll probably be concerned with handshaking. Handshaking simply lets the printer tell the AIM whether the printer is ready for more data or not. However, the AIM has no provision for accepting a handshaking signal on its serial port. Since I wasn't using the serial keyboard input for anything, I decided that that was a good place to connect the printer's handshaking line. On my printer, the handshaking line will either be high or low. If you study the schematic of the AIM's serial keyboard input, you'll find that the signal works its way back to PB6 of the AIM's VIA. Thus, to determine if the printer is busy, all we need to do is test PB6 for a high or low. As shown in listing 1, a BIT instruction takes care of this.



6502 COOKBOOK

For the machine language programmer who knows it is time to get serious! Here are search and sort routines, utilities, I/O & interrupt procedures, conversion routines, stack operations. Contains flow charts, source listings, explanations. Machine codes provided in both octal and hexadecimal. Tens of thousands of our programmer's cookbooks are in use.

Order publication: 6502 SOFTWARE COOKBOOK
Price in United States: just $12.95 + $1.00 s/h by mail.

Please include remittance with order. Allow 3 - 4 weeks for delivery. MasterCard & VISA credit cards accepted. Our phone line for credit card orders is (203) 888-1946. Foreign price list available. Write for more information.

☐ Check here for descriptive literature & catalog.

Name: ______________________________
Addr: ______________________________
City: ________________ State: ___ Zip: _____
MC/VISA # ________________ Bank: _____
Signature: ____________________________

SCELBI Publications
35 Old State Road, Oxford, CT 06483

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.

$64.95 plus shipping (5 lbs.)

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

***Listing 1:* Assembler Formatter in PL/65**

```
;"ASSEMBLER FORMATTER";
;;
;"CHRIS FLYNN  2/81";
;;
;;
;"AIM SUBROUTINES";
;;
;DEF OUTTTY=$EEA8;
;;
;"AIM RAM";
;;
;DEF CNTH30=$A417;
;DEF CNTL30=$A418;
;;
;"AIM I/O PORT";
;;
;DEF SYSORB=$A800;
;;
;"ZERO PAGE RAM";
;;
;DEF *=$F0;
;DCL CHR,LINECT,PNT,I,J;
;DCL HCHR,TMP,TAB;
;DCL R0[2],R1,R2,R3;
;;
;"USER OUTPUT VECTOR";
;;
;DEF *=$010A;
;DCL UOUT WORD INIT[ASMOUT];
;;
;;
;DEF *=$0200;
;;
;"VECTORS TO USER I/O HANDLERS";
;;
;'INITTY JMP INIT';
;'TTYOUT JMP OUTCHR';
;;
;"BEGIN USER OUTPUT ROUTINE";
;"PERFORM INITIALIZATION IF CARRY CLEAR
  UPON ENTRY";
;;
;ASMOUT:
;'BCS LINES';
;LINECT=0;
;CALL INITTY;
;CALL CLEAR;
;TAB=0;
;RETURN;
;;
;"STORE LINE IN LBUFF";
;;
;LINES:
;UNSTACK CHR;
;IF CHR ↑=$0D THEN

;     DO;
;          LBUFF[PNT]=CHR;
;          INC PNT;
;          RETURN;
;     END;
;;
;"LINE HAS BEEN BUILT";
;;
;;
;;
;"TEST FOR LABEL LINE";
;"SEND 13 CHARS & NO CR";
;;
;IF LBUFF[0]='=' THEN
```

```
;     DO;
;         IF TAB>0 THEN
;             CALL CRLF;
;         FOR I=0 TO 12
;             DO;
;                 HCHR=LBUFF[I];
;                 IF HCHR>$1F THEN
;                     DO;
;                         'LDA HCHR';
;                         CALL TTYOUT;
;                     END;
;             END;
;         TMPCMT=TABCMT;
;     END;
;ELSE DO;
;;
;;
;"TEST FOR COMMENT LINE";
;"TAB TO TMPCMT & PRINT";
;;
;;
;IF LBUFF[0]=';' THEN
;     DO;
;         IF LBUFF[1]=$00 THEN
;             DO;
;                 TMPCMT=TABINS;
;             END;
;         IF TAB > TMPCMT THEN
;             DO;
;                 CALL CRLF;
;             END;
;         TMP=TMPCMT-TAB;
;         FOR I=1 TO TMP
;             DO;
;                 'LDA #$20';
;                 CALL TTYOUT;
;             END;
;         FOR I=0 TO 79
;             DO;
;                 IF TAB>71 THEN
;                     DO;

;                    CALL CRLF;
;                    FOR J=1 TO TMPCMT
;                        DO;
;                            'LDA #$20';
;                            CALL TTYOUT;
;                        END;
;                      END;
;                 HCHR=LBUFF[I];
;                 IF HCHR>$1F THEN
;                     DO;
;                         'LDA HCHR';
;                         CALL TTYOUT;
;                     END;
;             END;
;         CALL CRLF;
;     END;
;ELSE DO;
;;
;;
;"TEST FOR PAGE EJECT";
;;
;IF LBUFF[0]=$5F THEN
;     DO;
;         TMP=55-LINECT;
;         FOR I=1 TO TMP
;             DO;
;                 CALL CRLF;
;             END;
;         TMPCMT=TABCMT;
```

(Continued)

Dithertizer II™...

...the eye of your apple.®

Though it is very simple to use, the Dithertizer II represents the ultimate in video digitizing using the Apple II computer. The Dithertizer is an interface card which converts video input into digitized images. Because the Dithertizer II is a frame grabber, DMA type digitizer, it offers extreme high speed in the conversion process (it grabs an entire frame in 1/60th of a second). The camera supplied with the package is the Sanyo model VC1610X. Cabling is supplied for this camera so as to have the Dithertizer II system up and running in minutes. The video camera used for input must have external sync to allow for the frame grabber technology employed for digitizing. If a camera other than the model recommended is used, wiring adaptations by the user may be required. Software is supplied with the board to allow you to display up to 64 pseudo grey levels on your Apple's screen. The number of grey levels may be changed with one keystroke. The intensity and contrast of the image are controllable via game paddles. Also supplied is software for image contouring for those interested in movement detection or graphic design applications.

The Dithertizer II package is available ready to run with camera, interface card and the software described above for only: **$650.00**

Dithertizer II interface card and software (without camera): **$300.00**

Computer Station
11610 Page Service Dr.
St. Louis, MO 63141
(314) 432-7019

Apple II is a registered trademark of Apple Computer, Inc.
Ditherthizer II is a trademark of Computer Station, Inc.

```
;     END;
;ELSE DO;
;;
;;
;;
;"TAB ALL ELSE TO TABINS";
;;
;IF LBUFF[0]>$1F THEN
;     DO;
;         IF TAB > TABINS THEN
;             DO;
;                 CALL CRLF;
;             END;
;         TMP=TABINS-TAB;
;         FOR I=1 TO TMP
;             DO;
;                 'LDA #$20';
;                 CALL TTYOUT;
;             END;
;         FOR I=0 TO 63
;             DO;
;                 IF I=7 THEN
;                     DO;
;                         'LDA #$20';
;                         CALL TTYOUT;
;                     END;

;                HCHR=LBUFF[I];
;                IF HCHR>$1F THEN
;                    DO;
;                        'LDA HCHR';
;                        CALL TTYOUT;
;                    END;
;            END;
;        TMPCMT=TABCMT;
;    END;
```

```
;END;
;END;
;END;
;;
;"PROCESSING OF OUTPUT LINE COMPLETED";
;"CLEAR LINE BUFFER AND RETURN TO CALLER";
;;
;CALL CLEAR;
;RETURN;
;;
;;
;;
;;
;"CARRIAGE RETURN, LINE FEED";
;"NOTE: ALL OUTPUT IS INDENTED 5 SPACES";
;;
;CRLF:
;'LDA #$0A';
;CALL TTYOUT;
;'LDA #$0D';
;CALL TTYOUT;
;INC LINECT;
;IF LINECT > 55 THEN
;     DO;
;         LINECT=0;
;         FOR J=1 TO 10
;             DO;
;                 'LDA #$0A';
;                 CALL TTYOUT;
;                 'LDA #$0D';
;                 CALL TTYOUT;
;             END;
;     END;
;FOR J=1 TO 5
;     DO;
;         'LDA #$20';
```

(Continued)

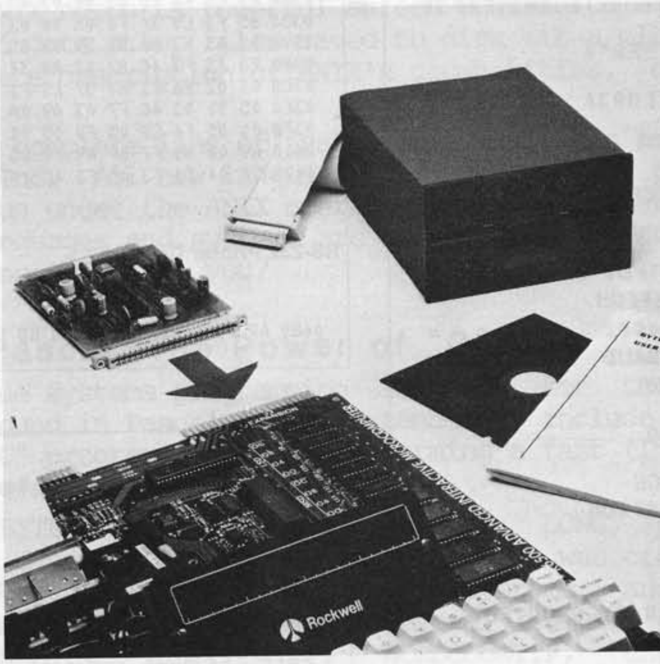
DOS FOR AIM-65®: $499.00 COMPLETE!

INSTANT EASY ACCESS TO YOUR AIM-65 PROGRAMS!

MANIPULATE OBJECT FILES AT THE TOUCH OF A KEY.

NEW BYTE-DOS DISC OPERATING SYSTEM.

153K BYTES ON-LINE. INEXPENSIVELY!



BYTE-DOS COMMANDS

- **FORMAT** - Prepares New Disc
- **BAD SECTOR SCAN** - QC's Disc
- **DUMP** - RAM File to Disc
- **LOAD** - Disc File to Ram
- **LIST** - Prints Disc File Names
- **ERASE** - Eliminate a File
- **UNERASE** - File Recovery
- **RENAME** - Rename a File
- **COPY** - Copy 1 File
- **TRANSFER** - Copy Complete Disc
- **KOMPRESS** - Optimizes Disc Space

AND MORE!

IT'S EASY AND FUN TO USE!

BYTE MICROSYSTEMS
CORPORATION
1477 ELKA AVENUE, SAN JOSE, CA 95129 408/446/0559

• BYTE-DOS OPERATES WITH THE AIM-65 MONITOR, BASIC, ASSEMBLER, EDITOR, PL/65 AND FORTH. ACCESS IS THROUGH PROGRAM I/O AND F1-F3 KEYS. BYTE-DOS SUPPORTS ONE OR TWO 5¼" OR 8" FLOPPYS!

• BYTE-DOS $499 SYSTEM INCLUDES DISC CONTROLLER CARD, TEAC FD-50A DISC DRIVE (153K FORMATTED BYTES), BYTE-DOS SOFTWARE SYSTEM, CABLES, CONNECTORS, MANUALS. COMPLETE! JUST PLUG IT TOGETHER AND USE!

AIM-65 is a trademark of Rockwell International

master charge THE INTERBANK CARD

VISA

SEND ME THE FULL STORY!

Name ____________

Company ____________

Address ____________

City ________ State ______ Zip ______

Telephone ____________ ☐ home ☐ work

☐ I own an AIM-65.

☐ I'm thinking of buying an AIM-65.

☐ Send full BYTE-DOS Data.

```
;           CALL TTYOUT;
;       END;
;TAB=0;
;RETURN;
;;
 ;;
 ;;
 ;"CLEAR LINE BUFFER";
 ;;
 ;CLEAR:
;FOR I=0 TO 79
;       DO;
;               LBUFF[I]=0;
;       END;
;PNT=0;
;RETURN;
;;
;;
;"TAB CONSTANTS";
;;
;DCL TABINS BYTE INIT[15];
;DCL TABCMT BYTE INIT[40];
;DCL TMPCMT BYTE INIT[40];
;;
;"LINE BUFFER";
;;
;DCL LBUFF[80];
;;
;;
;;
;"BEGIN USER I/O DEVICE '
 DRIVERS"
;;
;*BEGIN OUTPUT DEVICE
 INITIALIZATION
;INIT LDA #$02
;SET 1200 BAUD
;STA CNTH30
;LDA #$FD
;STA CNTL30
;RTS
;*END DEVICE
 INITIALIZATION
;;
;;
;;
;*BEGIN CHARACTER OUTPUT
;OUTCHR
;BIT SYSORB
;BVS OUTCHR
;JSR OUTTTY
;INC TAB
;RTS
;*END CHARACTER OUTPUT
;;
;;
;ENDPGM:
;EXIT;
```

Listing 2: **Vector to User Output Routine**

```
010A 06 02
```

Assembler Reformatter Program Code

```
0200 4C 89 04 4C 94 04 B0 0F A9 00 85 F1 20 00 02 20 15 04 A9 00 85 F7 60 68
0218 85 F0 A9 0D C5 F0 D0 03 4C 2E 02 A5 F2 A8 A5 F0 99 39 04 E6 F2 60 A9 3D
0230 CD 39 04 F0 03 4C 77 02 A9 00 C5 F7 90 03 4C 44 02 20 C0 03 A9 00 85 F3
0248 C9 0C F0 05 90 03 4C 6E 02 A5 F3 AA BD 39 04 85 F5 A9 1F C5 F5 90 03 4C
0260 67 02 A5 F5 20 03 02 E6 F3 A5 F3 4C 48 02 AD 37 04 8D 38 04 4C BC 03 A9
0278 3B CD 39 04 F0 03 4C 15 03 A9 00 CD 3A 04 F0 03 4C 91 02 AD 36 04 8D 38
0290 04 AD 38 04 C5 F7 90 03 4C 9E 02 20 C0 03 AD 38 04 38 E5 F7 85 F6 A9 01
02A8 85 F3 C5 F6 F0 05 90 03 4C BF 02 A9 20 20 03 02 E6 F3 A5 F3 4C AA 02 A9
02C0 00 85 F3 C9 4F F0 05 90 03 4C 0F 03 A9 47 C5 F7 90 03 4C F2 02 20 C0 03
02D8 A9 01 85 F4 CD 38 04 F0 05 90 03 4C F2 02 A9 20 20 03 02 E6 F4 A5 F4 4C
02F0 DC 02 A5 F3 AA BD 39 04 85 F5 A9 1F C5 F5 90 03 4C 08 03 A5 F5 20 03 02
0308 E6 F3 A5 F3 4C C3 02 20 C0 03 4C BC 03 A9 5F CD 39 04 F0 03 4C 46 03 A9
0320 37 38 E5 F1 85 F6 A9 01 85 F3 C5 F6 F0 05 90 03 4C 3D 03 20 C0 03 E6 F3
0338 A5 F3 4C 2A 03 AD 37 04 8D 38 04 4C BC 03 A9 1F CD 39 04 90 03 4C BC 03
0350 AD 36 04 C5 F7 90 03 4C 5D 03 20 C0 03 AD 36 04 38 E5 F7 85 F6 A9 01 85
0368 F3 C5 F6 F0 05 90 03 4C 7E 03 A9 20 20 03 02 E6 F3 A5 F3 4C 69 03 A9 00
0380 85 F3 C9 3F F0 05 90 03 4C B6 03 A9 07 C5 F3 F0 03 4C 99 03 A9 20 20 03
0398 02 A5 F3 AA BD 39 04 85 F5 A9 1F C5 F5 90 03 4C AF 03 A5 F5 20 03 02 E6
03B0 F3 A5 F3 4C 82 03 AD 37 04 8D 38 04 20 15 04 60 A9 0A 20 03 02 A9 0D 20
03C8 03 02 E6 F1 A9 37 C5 F1 90 03 4C F7 03 A9 00 85 F1 A9 01 85 F4 C9 0A F0
03E0 05 90 03 4C F7 03 A9 0A 20 03 02 A9 0D 20 03 02 E6 F4 A5 F4 4C DD 03 A9
03F8 01 85 F4 C9 05 F0 05 90 03 4C 10 04 A9 20 20 03 02 E6 F4 A5 F4 4C FB 03
0410 A9 00 85 F7 60 A9 00 85 F3 C9 4F F0 05 90 03 4C 31 04 A5 F3 A8 A9 00 99
0428 39 04 E6 F3 A5 F3 4C 19 04 A9 00 85 F2 60 0F 28 28
```

RS-232 Printer Driver

```
0489 A9 02 8D 17 A4 A9 FD 8D 18 A4 60 2C 00 A8 70 FB 20 A8 EE E6 F7 60
```

Listing 3A: **Source Listing of Sample Program**

```
*=$0200
OUTTTY =$EEA8 ;AIM SERIAL OUTPUT ROUTINE
;
;SAMPLE PROGRAM TO ILLUSTRATE ASSEMBLER REFORMATTER.
;
CRLF LDA #$0D ;SEND CR
JSR OUTTTY
LF
LDA #$0A ;SEND LF
JSR OUTTTY
RTS ;BACK TO CALLER
.END
```

Listing 3B: **Assembler Listing of Sample Program**

```
                    PASS 1
                    PASS 2
==0000                        *=$0200
==0200 OUTTTY                 =$EEA8          ;AIM SERIAL OUTPUT ROUTINE
                    ;
                    ;SAMPLE PROGRAM TO ILLUSTRATE ASSEMBLER REFORMATTER.
                    ;
==0200 CRLF         A90D      LDA #$0D        ;SEND CR
                    20A8EE    JSR OUTTTY
==0205 LF           A90A      LDA #$0A        ;SEND LF
                    20A8EE    JSR OUTTTY
                    60        RTS             ;BACK TO CALLER
                              .END
```

PROGRAMMING THE APPLE JUST BECAME EASIER!

ANIX V1.0

A DOS 3.3 Compatible, Unix-like Operating System

ANIX 1.0 is a "Tiny-UNIX™" that is completely file structure upwards compatible with DOS 3.3. ANIX incorporates a healthy subset of UNIX'S™ capabilities, and a superset of CP/M's™ capabilities.

In addition to the ANIX operating system, Lazer provides over 30 utility programs on the ANIX disk. While the list is much too long to describe here, many of the programs found on the ANIX disk are selling alone for between $30 and $60. ANIX supports powerful screen editing commands and will automatically time and date stamp files saved to disk (if a clock card is installed). For a complete description of ANIX's capabilities, call or write Lazer.

ANIX is the start of a complete line of system software tools available from Lazer MicroSystems, Inc. All new languages and applications programs available from Lazer will run under the ANIX operating system. Lazer Pascal is available now. Other languages and systems are in the works. Productive programmers are already using ANIX, are you?

PASCAL

The Elegance of Pascal, The Power of "C"

Lazer Pascal is a unique systems programming language for the Apple II. It combines features found in Pascal and is extended to include several features found in the "C" programming language forming a fast (1500-2000 lines/min) and powerful program creation system.

Lazer Pascal supports BYTE, CHAR, BOOLEAN, INTEGER, LONG, pointer, string, array, static, and dynamic data types. Lazer Pascal was created to replace 6502 machine language as the choice of systems and game programmers. Included with Lazer Pascal are several sample programs including: ANIX.P, TSTPARMS.P, LINECOUNT.P, WORDCOUNT.P, CHARCNT.P, EXPAND.P, COMPRESS.P, CRYPT.P, and TRANSLIT.P. Better yet, the 6502 source listings (LISA v2.5 format) for the compiler, P-code interpreter, and other utilities are included.

ANIX, Lazer Pascal, and DISASM/65 were all written by Randy Hyde, the author of "USING 6502 ASSEMBLY LANGUAGE", LISA, SPEED/ASM, DOSOURCE 3.3, and other fine software products. Additional information on Lazer's software products can be obtained by calling or writing Lazer MicroSystems, Inc.

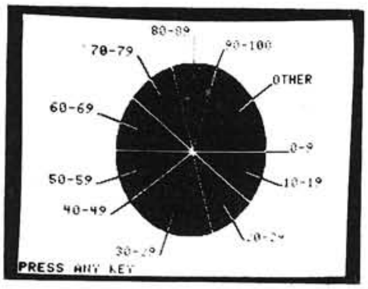
ANIX:	$49.95
LAZER PASCAL:	$39.95

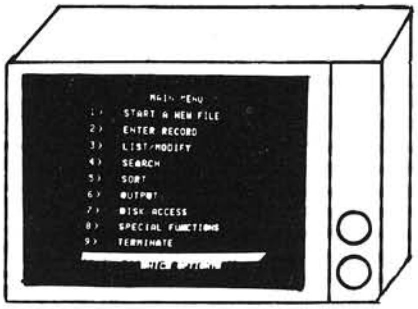
UNIX™ Bell Labs.
APPLE II™ Apple Computer, Inc.
CP/M™ Digital Research, Inc.

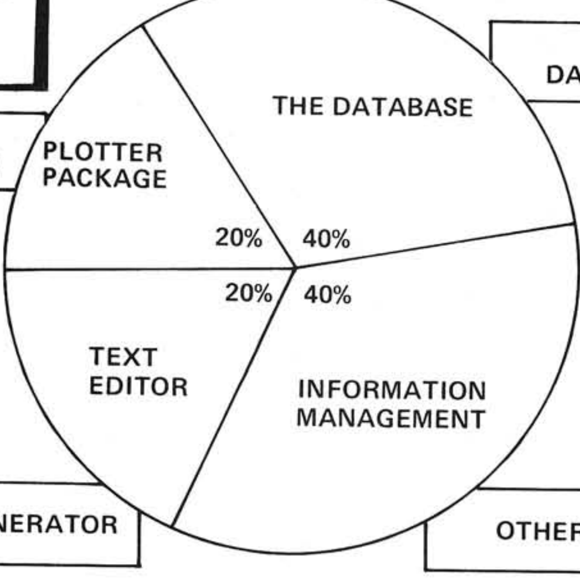
Other fine Lazer Products:

Lower Case +Plus II:	$29.95
Lower Case +Plus:	$64.95
Keyboard +Plus:	$99.95
Character Set +Plus:	$24.95

Lazer MICRO SYSTEMS INC.
1791-G Capital
Corona, CA 91720
(714)735-1041


80-89
90-100
78-79
OTHER
60-69
0-9
50-59
10-19
40-49
20-29
30-39
PRESS ANY KEY


MAIN MENU
1) START A NEW FILE
2) ENTER RECORD
3) LIST/MODIFY
4) SEARCH
5) SORT
6) OUTPUT
7) DISK ACCESS
8) SPECIAL FUNCTIONS
9) TERMINATE


THE DATABASE
PLOTTER PACKAGE
20%
40%
20%
40%
TEXT EDITOR
INFORMATION MANAGEMENT


ODOMETER MILEAGE STATEMENT

MICRO

Microbes and Updates

Dear MICRO:

There is an error in the "Pascal Tutorial: Part 2" in MICRO 43:57. The error has to do with the status of the file SYSTEM.WRK.TEXT in the Apple Pascal Operating System.

The manual for the Pascal software emphasizes that the workfile is treated differently from other files. Essentially, only the commands G(et, S(ave, and N(ew should be used with the file SYSTEM.WRK.TEXT. The commands C(hange, T(ransfer, and R(emove should *never* be used with the workfile.

The reason we must be careful when working with the workfile is that Pascal "remembers" the status of the workfile, and its integrity can only be assured if we operate on the workfile with, and only with, the established workfile commands. For example, if we R(emove the workfile and attempt to E(dit it, the message WORKFILE LOST will be given. But if we clear the workfile with N(ew, the editor will realize that there is no workfile and ask for a file to edit in the usual way.

There is one other point I'd like to make concerning the S(ave command in the filer. If the file is saved on the Root Volume (in the example from the article: LEDGER or APPLE1:LEDGER rather than, say, #5:LEDGER), the workfile will be *renamed* LEDGER.TEXT and LEDGER.CODE. A subsequent N(ew command will result in WORKFILE CLEARED. On the other hand, if the workfile is saved on some other volume, it will be *copied* to that volume without disturbing SYSTEM.WRK.TEXT and .CODE on the Root Volume. The filer responds to the N(ew command with THROW AWAY CURRENT WORKFILE? (Correct answer is "Y", since the workfile has been saved on another volume.)

Paul K. Fessler
Sales Manager
Computers Plus
6120 Franconia Road
Alexandria, VA 22310

Dear MICRO:

Here are corrections to a few typographical errors in my article, "Precision Programming" (MICRO 42:06).

On page 8 in the IF-THEN-ELSE figure, the expression in the diamond reads x = y; it should read: x > = y.

In the first paragraph on page 9, the second line contains words in quotes; these should read: "leading zero?".

Also on page 9, in the third column near the top, the *while* statement should read: *while* ((i < = n), etc. Further down the column, the if statement should read: *if* i > n, etc.

On page 10, in the third paragraph in the first column, a ">" is left out. The statement should read: TABLE I = KEY or I>N.

Al Hamilton
12090 Brookston Dr.
Springdale, OH 45240

Dear MICRO:

There is an error in my article, "The PET from A to D" (MICRO 41:60). The voltage-controlled oscillator circuit in the article is erroneously referred to as the 74LS 235. The correct designation is 74LS 325. I regret any inconvenience the error may have caused to readers.

John R. Sherburne
4418 Andes Drive
Fairfax, VA 22030

Dear MICRO:

The following errors were noted in the article, "Ultimate Ping-Pong for PET," by Werner Kolbe (42:67). While the HEX addresses were correct, the assembly was in error.

```
0AF8- 10 F8      BPL L1 (should be)
0AF8- 10 F8      BPL L13

0C80- 85 BF      ST *RRYS (should be)
0C80- 85 BF      STA *RRYS

0CBF- 80 4B E8 SA W26 (should be)
0CBF- 80 4B E8 STA W26

0CF4- 90 02      BCC L3 (should be)
0CF4- 90 02      BCC L53
```

Otherwise this is a very good program.

Albert I. Reuss
13978 Morgan Avenue
P.O. Box 388
Clearlake, CA 95422

Dear MICRO:

I am enjoying immensely the article "Function Generator and Library Manager" by Ray Cadmus (MICRO 42:94). I thought you might be interested in a quick and dirty fix for this print-using routine. It isn't fancy, but it takes care of a problem cited. It also implements a trailing minus sign (see below).

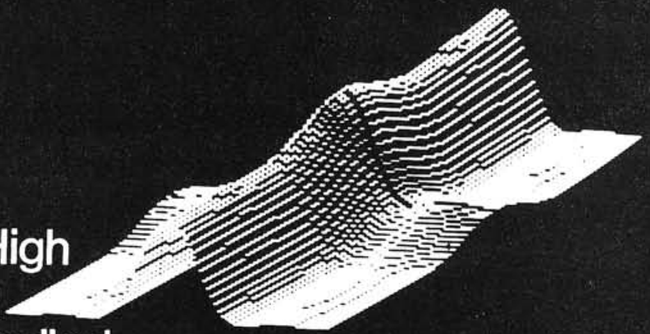
One problem which still puzzles me is roundoff. The original routine rounded .025 to .02. Changing the expression "INT ((P + .005) * 100)" to "INT ((.005 + P) * 100)" (Mr. Cadmus' line 8004, my line 8010) fixes the problem, but don't ask me why!

Charles F. Taylor, Jr.
587 F Sampson Lane
Monterey, CA 93940

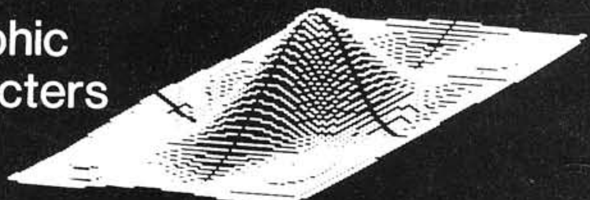
Taylor Listing

```
8000  REM                                 *** PRINT USING ***

8002  REM  FOR $ AND CENTS FORMAT
8004  REM  P= VALUE: PL= LENGTH (INCL SIGN)
8006  REM  P$ = FIELD ACTUALLY PRINTED
8008 SG =  SGN (P):P =  ABS (P)
8010 P$ =  STR$ ( INT ((.005 + P) * 100))
8012  IF  LEN (P$) < 3 THEN P$ =  LEFT$
         ("000",(3 -  LEN (P$))) + P$
8014 P$ =  LEFT$ (P$,( LEN (P$) - 2)) + "." +  RIGHT$ (P$,2)
8016 P$ =  RIGHT$ ("              " + P$, PL - 1)
8018  IF SG < 0 THEN P$ = P$ + "-":P = SG * P: GOTO 8022
8020 P$ = P$ + " "
8022  RETURN
```

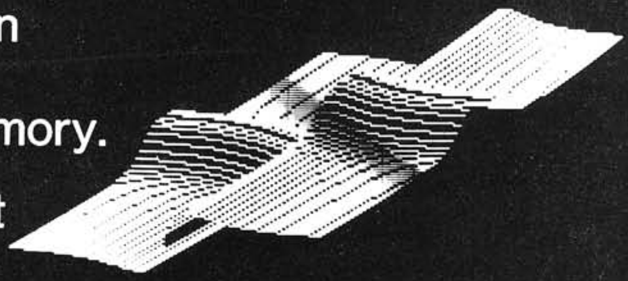
GRAPHICS
FOR OSI COMPUTERS



- ☆ 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.

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

Superboard Expansion System

This easily constructed motherboard can vastly increase the capabilities and applications of the C1P and Superboard II. Memory and I/O plug-in boards double the memory and allow easy interfacing to external devices.

D.W. Kammer
Department of Physics
Albion College
Albion, Michigan 49224

The Ohio Scientific C1P and its single-board version, the Superboard II, are two of the best buys on the current small computer market. With 8K Microsoft BASIC-in-ROM, an ASCII keyboard plus video and cassette interfaces, these computers are capable of most of the requirements of a hobby computer.

However, there are two serious limitations. A maximum of 8K of random access memory (RAM) is available and the input and output (I/O) capabilities are meager. The memory limitation is particularly important when using an assembler/editor (which takes up most of the 8K RAM) for machine code programming. The only I/O available is a 6850 ACIA with an unpopulated RS-232 port and a joystick port. It is therefore difficult to interface the computer to all the peripheral devices that make a computer fun after tiring of BASIC programming. OSI sells an expansion board that provides extra memory (with no I/O ports), but it costs as much as a Superboard II.

The most general approach to these problems is the construction of a motherboard interfaced to the computer bus. This enables the construction of a variety of circuit boards which are simply plugged into edge connectors on the motherboard. Your imagination is now the only limiting factor on what the computer can do. This article describes 8K RAM memory and I/O plug-in boards, but I have built boards for burning EPROMs, music and sound generation, Morse code/RTTY transmit and receive, and serial I/O for printer operation.

The Motherboard

Access to the computer system bus is gained through J1, a 40-pin DIP socket on the computer board. The pin identification for this socket is given in table 1. No RESET signal is provided by OSI, but it is useful for certain peripheral ICs. A wire should be soldered from the CPU side of the BREAK key on the keyboard to pin 11 on the DIP socket. The DATA lines are buffered by 8T28s (not supplied by OSI) on the main board; the other lines are connected directly to the 6502 CPU.

The 6502 does not have the drive capability to handle a motherboard so buffers must be added. Figure 1 shows the buffer circuit which goes between the computer and the edge connectors. All signals except for the interrupts, DATA and RESET lines, go through 74LS241 octal bus line drivers. The interrupt lines are buffered by a 74LS17 open-collector buffer driver. The 8T28 data buffers require a data direction (DD) signal which is low for reading data and high for writing data. Any peripheral circuits must have logic for controlling this line. Note that the interrupt and DD lines have pull-up resistors because several plug-in boards may require access to them. By using open-collector TTL ICs on the plug-in boards and the pull-ups, this line contention is easily solved.

The motherboard uses 44-pin 0.156 in. spacing edge connectors because of the availability of prototype boards for

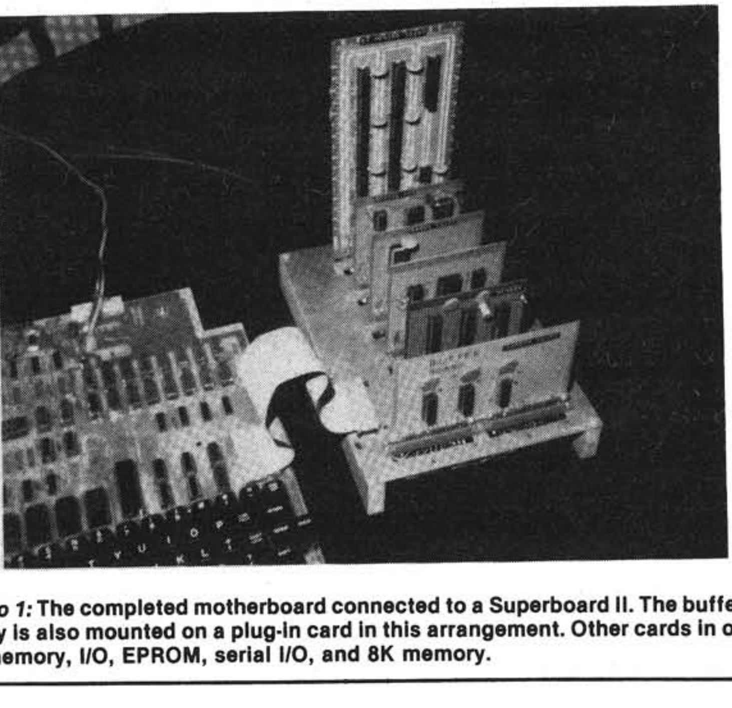


Photo 1: **The completed motherboard connected to a Superboard II. The buffer circuitry is also mounted on a plug-in card in this arrangement. Other cards in order: 3K memory, I/O, EPROM, serial I/O, and 8K memory.**

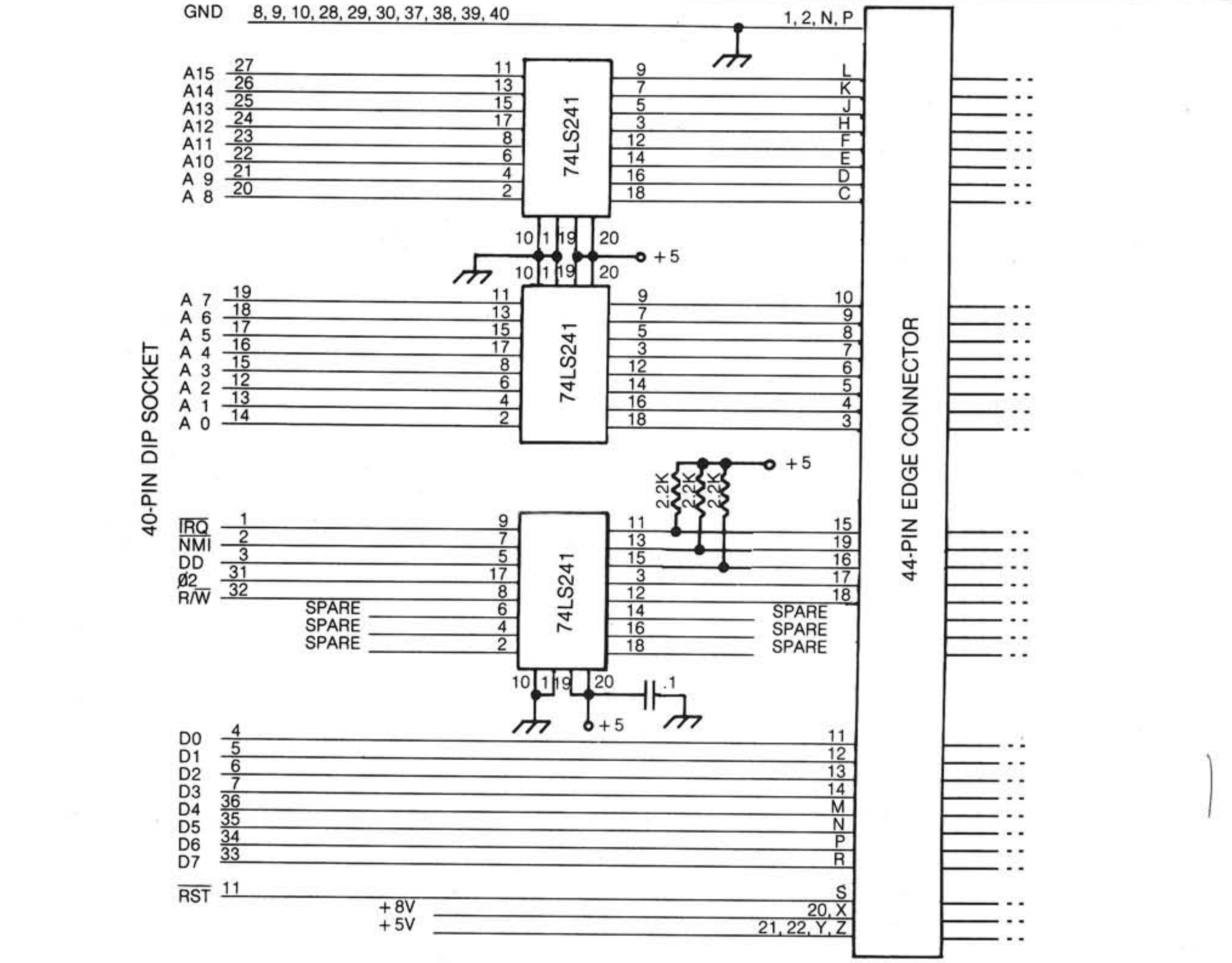


***Figure 1:* The motherboard circuit showing full buffering of all lines but data and reset. Additional edge connectors are wired in parallel with the one shown.**

them. Both Vector and Radio Shack have a variety of suitable boards. The connectors can be mounted on a sheet of perf board or aluminum after suitable slots are cut and simply wired in parallel. Power and ground lines must use heavy wire; 16 gauge copper as used in home wiring works well. A commercially etched motherboard (Electronic Systems, P.O. Box 21638, San Jose, CA 95151, Part No. 102) is available, which has room for ten 44-pin edge connectors. A 40-pin DIP socket should be mounted in some convenient place to receive the signals from the computer through a double 40-pin DIP jumper. The bus definition is given in table 2, and photo 1 shows one possible arrangement of the fully populated motherboard.

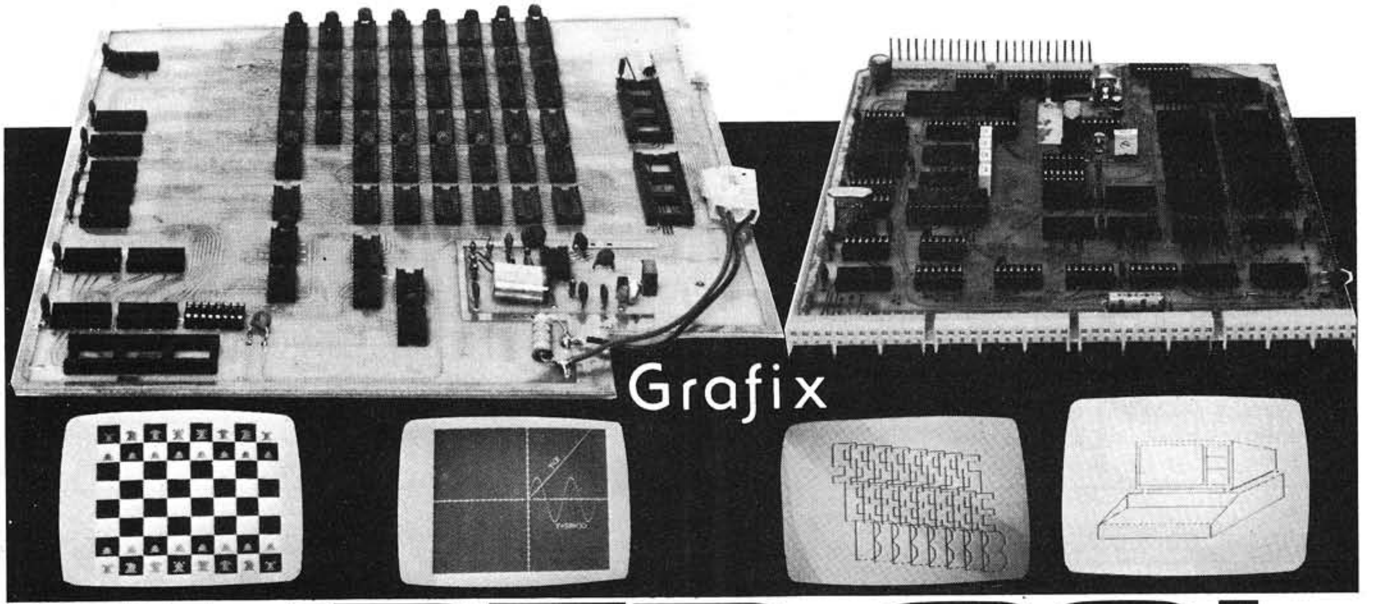
The power supply of the C1P may not be sufficient to meet the requirements of the motherboard, particularly if a large amount of memory is driven. This should be checked to prevent damage, and a suitable power supply

***Table 1:* OSI 600 Board J1 Expansion Connector Pinout**

1	IRQ	21	AB9
2	NMI	22	AB10
3	DD	23	AB11
4	DB0	24	AB12
5	DB1	25	AB13
6	DB2	26	AB14
7	DB3	27	AB15
8	Ground	28	Ground
9	Ground	29	Ground
10	Ground	30	Ground
11	RES (*)	31	Phase 2
12	AB2	32	R/W
13	AB1	33	DB7
14	AB0	34	DB6
15	AB3	35	DB5
16	AB4	36	DB4
17	AB5	37	Ground
18	AB6	38	Ground
19	AB7	39	Ground
20	AB8	40	Ground

***Table 2:* KIM-4 Expansion Connector Pinout**

E-1	Ground	E-A	Ground
E-2	(Sync)	E-B	AB0
E-3	(RDY)	E-C	AB1
E-4	IRQ	E-D	AB2
E-5	(S.O.)	E-E	AB3
E-6	NMI	E-F	AB4
E-7	RES	E-H	AB5
E-8	DB7	E-J	AB6
E-9	DB6	E-K	AB7
E-10	DB5	E-L	AB8
E-11	DB4	E-M	AB9
E-12	DB3	E-N	AB10
E-13	DB2	E-P	AB11
E-14	DB1	E-R	AB12
E-15	DB0	E-S	AB13
E-16	DD	E-T	AB14
E-17	(nc)	E-U	AB15
E-18	(nc)	E-V	Phase 2
E-19	+8v	E-W	R/W
E-20	+8v	E-X	(Phase 2)
E-21	+5v	E-Y	+5v
E-22	Ground	E-Z	Ground



SUPER OSI EXPANSION BOARDS

If your Challenger can't generate displays like those shown above **WHAT ARE YOU WAITING FOR?** The SEB-1 High Resolution Graphics and Memory Board (for CIP and Superboard II) and the SEB-2 High Resolution Graphics and Disk Controller Board (for C2/4/8) simply 'plug-in' to your computer and give you instant access to over 6000 individually addressable pixels in up to 8 colors! Your Hi-Res screen can go from 32×16 alphanumerics to 256×192 point graphics in 11 software selectable modes. The standard video of your computer is left intact, so that none of your current software library is outmoded. Use the graphics for Business, Scientific, Education, or Gaming displays that were impossible—until now!

Installation of either board requires absolutely NO modification of your computer—they just 'plug-in'. Nor do they preclude your using any other OSI-compatible hardware or software. In addition to the Hi-Res Graphics the SEB-1 gives C1 & Superboard II users 16K of additional user memory (over and above that memory devoted to the graphics), two 16 bit timers/counters, an on-board RF modulator, and a parallel port with handshaking. The SEB-2 gives OSI 48-pin BUS users an OSI hardware/software compatible Disk controller, and an RF modulator that can be user-populated.

FOR OSI 1P, 2-4P, 2-8P, C4P, C8P

	SEB-1	SEB-2
Assembled and Tested	**$249.00 (5K RAM)**	**$239.00 (1K RAM)**
Kit	**$165.00 (No RAM)**	**$145.00 (No RAM)**

	SEB-1	SEB-2
Bare Board & Manual	**$ 59.00**	**$ 59.00**
Manual only	**$ 5.00**	**$ 5.00**

SOFTWARE

TARG-IT! **$11.95**

An action-packed, super fast arcade game where you try to destroy Ram ships and the Evil Spectre inside the Crystal city. 8K.

PLANETARY LANDER **$11.95**

Your computer generates contour maps of the surface of a strange planet as you descend looking for the landing pad. As you make your final approach, the computer displays side and top views of the terrain and your ship as you try to land while avoiding the swarms of asteroids above the planet's surface. 8K.

PROGRAMMA-TANK **$9.95**

Programma-Tank is a complex simulation of a battle between two robot tanks as well as a fairly painless way to learn assembly language. The opponents program their tanks in 'Smalltank' a miniature programming language, which is much like assembly language. The two strategies are then pitted against one another until a victor emerges. 8K.

RUINS OF TALAMARINE **$16.95**

This is the first of our "Epic Quests" programs. Epic Quests could be described as graphic adventures, but they are much more. Can you retrieve the stolen Jewels from the ruined Talamarine castle? This series of two programs allows you to create a character with the qualities that you want to do battle with the forces of darkness. 8K.

CODENAME: BARRACUDA **$19.95**

Another of our "Epic Quests", in which you are an agent working for the US Government trying to rescue a defecting Soviet professor. The problem is that a middle-eastern government has kidnapped the professor and is forcing him to develop atomic weapons for them. The three programs (8K ea.) in this series allow you to create, train and send your character on his mission.

DUNGEONS OF ZORXON **$19.95**

Can you explore the Dungeons of Zorxon in this "Epic Quest"? Step into a nether world of monstrous beasts and fortunes in gold where magic reigns supreme. Three programs 8K each.

COMING: SEB-3 80×24 Video/Disk Controller (C2/4/8), SEB-4 48K Memory RAM/ROM (C2/4/8), SEB-5 8K RAM/Disk/Sound/Clock/Voice (C1 & Superboard).

Write for **FREE** catalog
(For International requests,
please supply 2 oz. postage)

ORION® SOFTWARE ASSO.
147 MAIN ST, P O BOX 310, OSSINING, NY 10562

master charge VISA®

used to drive the motherboard if power requirements are excessive. The 8V line given in the bus definition is used if you have a separate power supply and makes the bus more versatile. This voltage must be regulated down to + 5V on your peripheral boards.

The 8K RAM Memory Board

The circuit for the memory board is shown in figure 2. Low-cost 2114L 450ns 8K static RAMs are used for memory and a 74LS138 1-of-8 decoder/demultiplexer decodes the address lines. Two common TTL gates (one open-collector) generate the DD signal. The address for the memory is $2000-$3FFF which is just above the 8K memory supplied on the computer board. It would be easy to change its location by placing unused gates in the DD decoder ICs between the address lines and the 74LS138, or by simply interchanging these address lines. The memory map of the C1P allows up to 40K of contiguous random access memory, to $9FFF.

All the ICs fit on a Vector type VCT-3677 DIP board with room to spare. The Radio Shack type 276-156 prototype board is too small to hold more than 3K of memory. Either wire-wrap or solder construction can be used. This board requires up to 1.5A power supply current, so be sure the supply is stiff enough and that adequate filtering and bypassing are used.

The I/O Board

Figure 3 shows the circuit for the simple I/O board. The 6522 Versatile Interface Adapter (VIA) is indeed one of the most versatile I/O chips available as

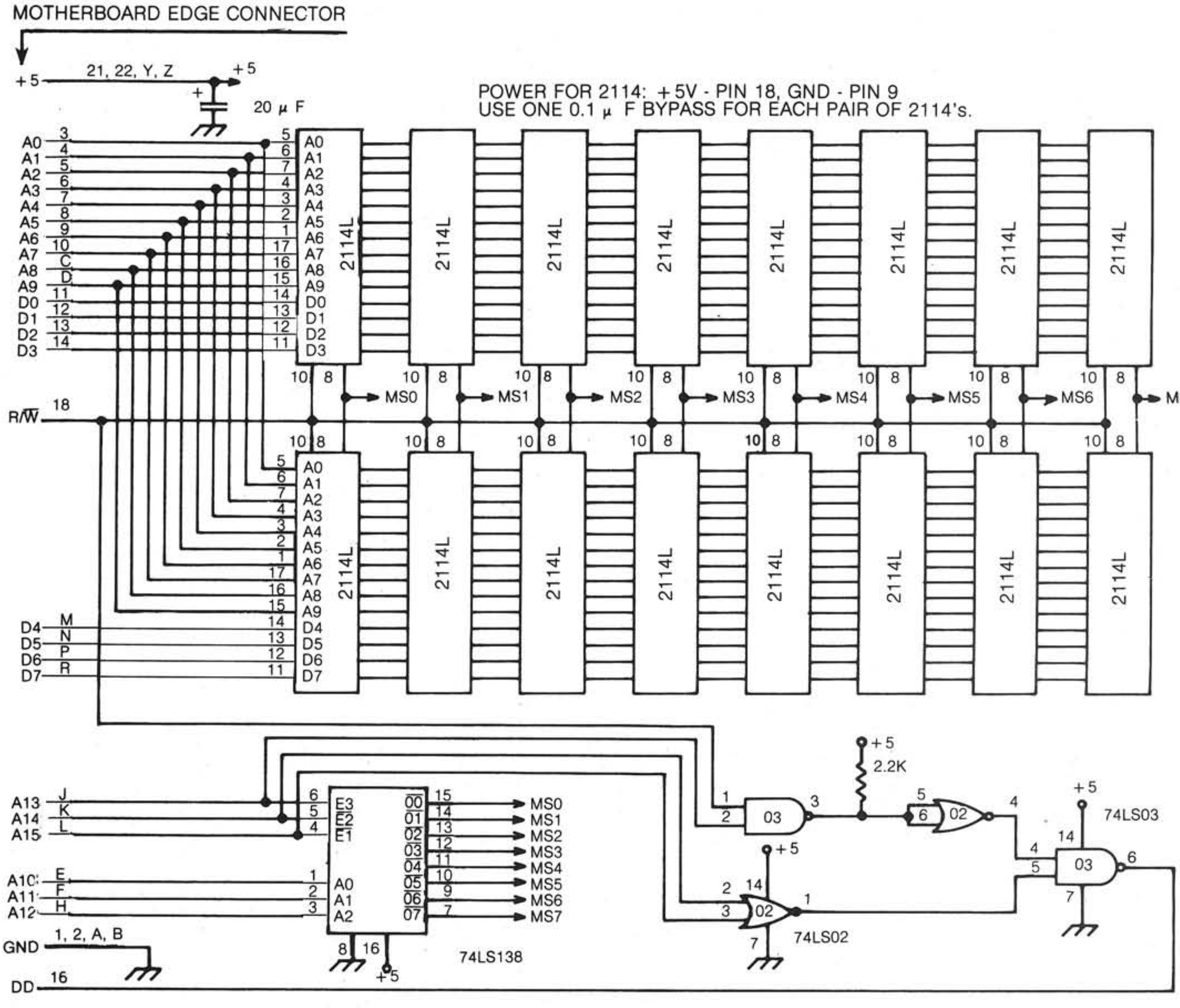


***Figure 2:* The 8K random access memory board. This circuit requires three support ICs, one an open-collector type for data direction decoding.**

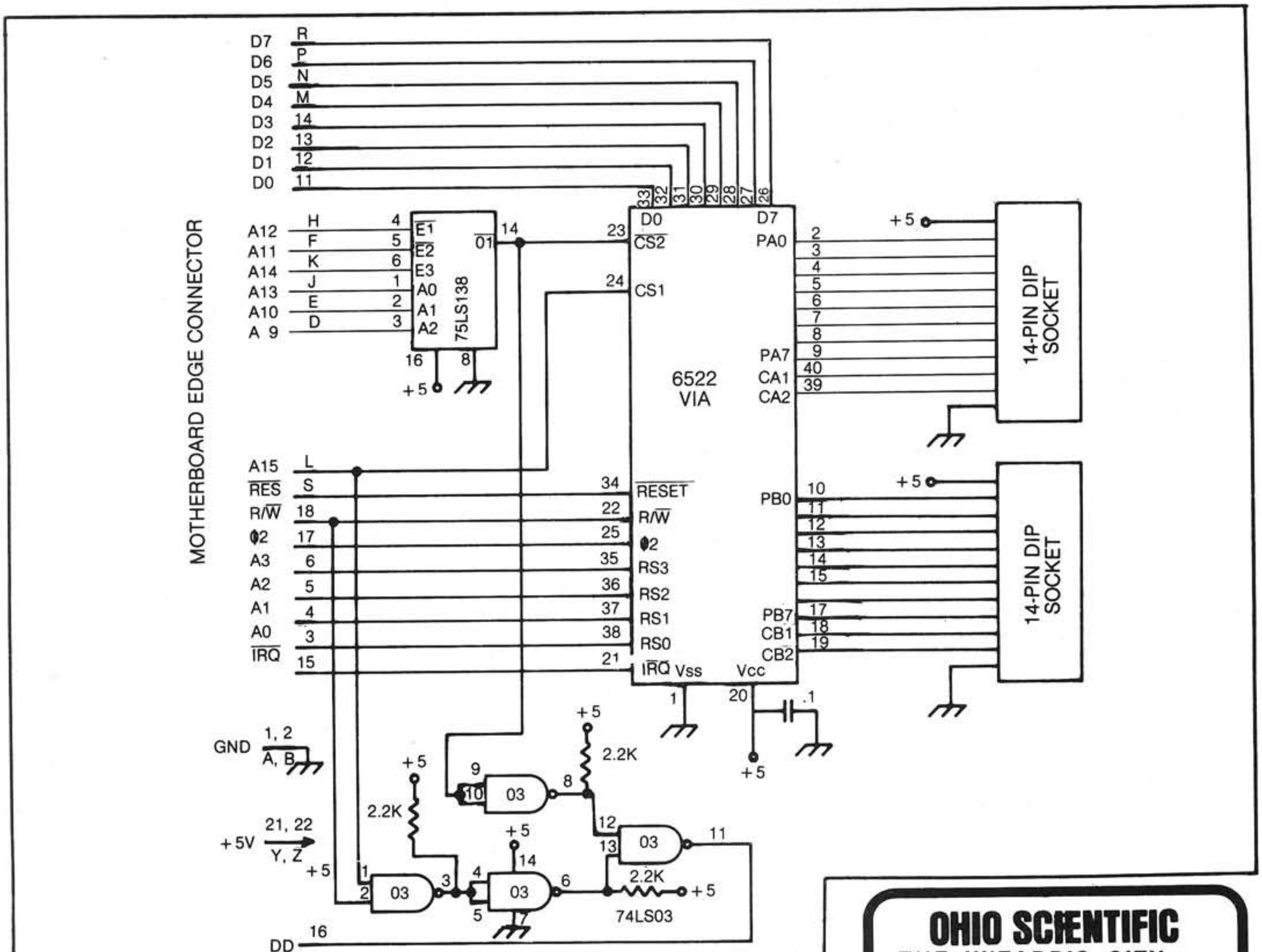


Figure 3: **The I/O board uses a 6522 Versatile Interface Adapter which provides 16 bidirectional I/O lines.**

detailed in a recent MICRO article (32:65). Besides two bi-directional 8-bit parallel ports, four handshaking lines, two timers and a shift register are included. It is also one of the easiest chips to program to do what is wanted. Just two support TTL ICs are required to handle address decoding and the DD signal. It would be easy to add another VIA using one of the 74LS138 unconnected output lines for address decoding; this would give 32 I/O lines. The 6520 or 6821 Peripheral Interface Adapter (PIA) could also be used with some modification of the address decoding.

The address for the VIA is $E000-$E00F, but the address bus is not fully decoded in the circuit, so these addresses are echoed to $E1FF. This wasted space is so high in memory that no problem should exist.

The circuit fits easily on a Radio Shack type 276-156 prototype board and can be wired in a short time. The I/O and handshaking lines are brought to two 14-pin DIP connectors. It would also be a good idea to bring +5V to these connectors in order to supply current to low-power peripheral circuits connected to the ports.

Conclusion

For anyone who likes to design and build hardware, a Superboard II with the peripheral motherboard is about the cheapest way to obtain a sophisticated system that is easily interfaced to any original circuit design. It indeed opens new horizons for the computer hobbiest.

David Kammer teaches physics at Albion College and became interested in microcomputers upon purchasing an ELF computer several years ago. Current interests include expanding his Superboard II (in progress: interfacing a Stringy Floppy© tape drive and an 80 × 24 video driver) and using a SYM to collect and analyze data in physics experiments.

MICRO

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.

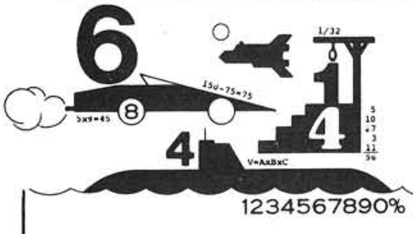
Car Jump—Make your stunt car jump the ramps. Each correct answer will increase the number of buses your car must jump over. These problems involve calculating the areas of different geometric figures.

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

USING ATARI'S COUNTDOWN TIMERS

The Atari Personal Computer System maintains five countdown timers, updated at a 1/60th-second rate. Although these timers were designed for use by the Atari Operating System, they may also be used by BASIC and Assembly language programs.

Mike Dougherty
7659 West Fremont Ave.
Littleton, Colorado 80123

The Atari Personal Computing System must perform several functions in order to maintain the interactive nature of its operating system. These functions, detailed in the two publications, *Atari Personal Computer System Operating System User's Manual* and *Hardware Manual*, available from Atari, include:

- updating the real-time clock buffer
- updating the attract mode byte
- updating the software countdown timers
- updating the color, joystick, paddle, and lightpen shadow registers
- turning off the keyboard speaker
- performing the software key repeat function

Listing 1

```
1 REM  S T O P W A T C H
2 REM
3 REM by Mike Dougherty
4 REM
5 REM USING A VBLANK INTERVAL TIMER
6 REM TO IMPLIMENT A STOPWATCH.
7 REM JOYSTICK 1 IS USED AS THE
8 REM STOPWATCH TRIGGER.
9 REM
10 TIME=0:GRAPHICS 0:POKE 752,1:SETCOLOR 2,3,4
20 POSITION 5,2:PRINT "                              "
30 POSITION 5,3:PRINT "      S T O P W A T C H       "
40 POSITION 5,4:PRINT "                              "
50 POSITION 5,5:PRINT "  PRESS   TRIGGER TO START    "
60 POSITION 5,6:PRINT "  RELEASE TRIGGER TO STOP     "
70 POSITION 5,7:PRINT "                              "
80 POSITION 13,10:PRINT " SECONDS "
90 IF STRIG(0)=0 THEN GOTO 90
100 IF STRIG(0)=1 THEN GOTO 100
200 REM
201 REM MAIN LOOP:
202 REM
203 REM _DOUNTIL TRIGGER IS RELEASED
204 REM ___WAIT FOR TIMEOUT
205 REM ___RESTART TIMMER
206 REM ___OUTPUT SOUND BEEP, TIME
207 REM ___UPDATE TIME FOR NEXT LOOP
208 REM _ENDDO
209 REM
210 IF PEEK(554)<>0 THEN GOTO 210
220 POKE 540,6
230 LOUD=10-LOUD:SOUND 0,121,10,LOUD
240 POSITION 16,12:PRINT "          ":POSITION 16,12:PRINT TIME/10
250 TIME=TIME+1
260 IF STRIG(0)=0 THEN GOTO 200
1000 REM
1001 REM SHUT DOWN STOPWATCH,
1002 REM WAIT FOR USER TO RESTART
1003 REM
1010 SOUND 0,0,0,0
1020 POSITION 5,14:PRINT "  PRESS TRIGGER TO RESTART    "
1030 POSITION 5,14:PRINT "                              "
1040 IF STRIG(0)=1 THEN GOTO 1020
1050 GOTO 10
```

***Table 1:* Atari Countdown Timer Addresses**

Timer CDTMx	Countdown Value CDTMVx	Countdown Flag CDTMFx	Countdown Completion Address CDTMAx
CDTM1	536. ($0218)	---	550. ($0226)
CDTM2	538. ($021A)	---	552. ($0228)
CDTM3	540. ($021C)	554. ($022A)	---
CDTM4	542. ($021E)	556. ($022C)	---
CDTM5	544. ($0220)	558. ($022E)	---

where x denotes the timer number 1, 2, 3, 4, or 5
Note: the real-time clock uses the three bytes 18., 19., and 20. ($0012-$0014) with the low order stored *last*.

To operate properly, these functions must be executed at regular and frequent intervals. However, the time required to perform these functions should not degrade the video output of the computer. Atari solved this problem by performing the above functions while the video output was blanked and the CRT gun was vertically returning for the start of the next video scan. Because of this, the routine performing these functions was called VBLANK. VBLANK is inititated by an interrupt generated at the end of a complete video scan, each 1/60th of a second. This interrupt forms the time base of the real-time clock buffer and the interval timers — subjects of this article.

In all, the Atari Personal Computer System maintains five countdown timers, CDTM1, CDTM2, CDTM3, CDTM4, and CDTM5. If one of these 16-bit timers is non-zero, it is decremented (counted down) at a 1/60th of a second rate until zero, one count per VBLANK interrupt. While the three countdown timers CDTM3, CDTM4, or CDTM5, count down to zero, a corresponding byte flag, CDTMF3, CDTMF4, or CDTMF5, is set to the value of 255 ($FF). When one of these timers reaches zero, the corresponding flag is set to zero. The two remaining countdown timers, CDTM1 and CDTM2, do not set any software flags. Instead, when one of these two timers counts down to zero, a call is made to an assembly language subroutine.

The address of this assembly language subroutine is located in RAM and may be modified to point to an arbitrary user subroutine. Thus CDTM3, CDTM4, and CDTM5, allow for the timing of events by monitoring software flags, while CDTM1 and CDTM2 allow a user-completion assembly language subroutine to be executed after a time period ranging from 1/60th of a second to 65535/60th of a second. Table 1 gives the specific memory locations and Atari labels for the five timers.

Listing 2

```
1 REM  S T O P W A T C H
2 REM
3 REM by Mike Dougherty
4 REM
5 REM USING THE VBLANK REAL TIME
6 REM CLOCK TO IMPLEMENT A STOPWATCH.
7 REM JOYSTICK 1 IS USED AS THE
8 REM STOPWATCH TRIGGER.
9 REM
10 TIME=0:GRAPHICS 0:POKE 752,1:SETCOLOR 2,3,4
20 POSITION 5,2:PRINT "                            "
30 POSITION 5,3:PRINT "     S T O P W A T C H      "
40 POSITION 5,4:PRINT "                            "
50 POSITION 5,5:PRINT "  PRESS    TRIGGER TO START "
60 POSITION 5,6:PRINT "  RELEASE TRIGGER TO STOP   "
70 POSITION 5,7:PRINT "                            "
80 POSITION 13,10:PRINT " SECONDS "
90 IF STRIG(0)=0 THEN GOTO 90
100 IF STRIG(0)=1 THEN GOTO 100
110 T=PEEK(20)+6:IF T>255 THEN T=T-256
200 REM
201 REM MAIN LOOP:
202 REM
203 REM _DOUNTIL TRIGGER IS RELEASED
204 REM ___WAIT FOR RTC TO REACH T
205 REM ___COMPUTE NEXT VALUE OF T
206 REM ___COUNT THIS DELAY
207 REM ___OUTPUT SOUND BEEP, TIME
208 REM _ENDDO
209 REM
210 IF PEEK(20)<>T THEN GOTO 210
220 T=T+6:IF T>255 THEN T=T-256
230 TIME=TIME+1
240 LOUD=10-LOUD:SOUND 0,121,10,LOUD
250 POSITION 16,12:PRINT "          ":POSITION 16,12:PRINT TIME/10
260 IF STRIG(0)=0 THEN GOTO 200
1000 REM
1001 REM SHUT DOWN STOPWATCH,
1002 REM WAIT FOR USER TO RESTART
1003 REM
1010 SOUND 0,0,0,0
1020 POSITION 5,14:PRINT "  PRESS TRIGGER TO RESTART  "
1030 POSITION 5,14:PRINT "                            "
1040 IF STRIG(0)=1 THEN GOTO 1020
1050 GOTO 10
```

Note that these timers were originally designed with the Atari operating system in mind. Various routines in the Central Input and Output Component of the operating system (CIO) use the timers to measure data transmission to and from peripheral devices. However, for memory-based applications, no apparent conflicts in countdown timer usage have been observed. In addition, there are times when the operating system is executing time critical code — times when the VBLANK routine is cut short. During such an interrupt, only the real-time clock and CDTM1 are updated, with the rest of the countdown timers skipped until the next VBLANK interrupt. Thus it is possible for the rest of the countdown timers to occasionally miss a "tick." Since the amount of time critical code in the operating system is small, missing an occasional 1/60th of a second tick is usually tolerable for real-time applications.

Listing 1 gives a simple application of the software flag countdown timer, CDTM3. (CDTM4 or CDTM5 could have been used equally well.) In this application, CDTM3 is used to implement a stopwatch accurate to 1/10th of a second (with only an occasional 1/60th of a second VBLANK tick being missed due to time critical code). Listing 1 uses the trigger of joystick 1 (TRIGGER(0)) to control the starting and stopping of the stopwatch. Since the timing is being performed automatically by VBLANK in the background, the program STOPWATCH does not need to worry about accuracy. Once the countdown timer is started with a value of 6, STOPWATCH has up to 1/10th of a second to display the time, set the sound voices, and any other desired "bells and whistles." This allows STOPWATCH considerable freedom to perform other functions while the timer is still maintaining accurate time.

For applications that cannot tolerate the occasional missing of a 1/60th of a second VBLANK tick, the real time clock buffer may be used. The real-time clock buffer is a three-byte, upward-counting timer, incremented with every VBLANK interrupt, time critical code or not. Since other portions of the operating system utilize the real-time clock, listing 2 implements the stopwatch without modifying this real-time clock value. This version of STOPWATCH will not lose time.

Listing 3

```
1 REM DEMONSTRATION OF THE BACKGROUND
2 REM PROCESSING CAPABILITIES OF THE
3 REM ATARI COMPUTER SYSTEM.
4 REM COUNTDOWN TIMMER 1 WILL BE USED
5 REM TO TRIGGER A BACKGROUND ROUTINE
6 REM THAT CHANGES THE SOUND 0 OUTPUT.
7 REM ONCE STARTED, THIS ROUTINE DOES
8 REM NOT REQUIRE ATTENTION FROM THE
9 REM USER.  LISTINGS, EDITING, ETC.
10 REM MAY BE PERFORMED WHILE THE
11 REM BACKGROUND IS RUNNING -- STOP
12 REM BY HITTING SYSTEM RESET.
13 REM
14 REM by Mike Dougherty
15 REM
1000 REM ..........................
1001 REM
1002 REM INITIALIZE THE BACKGROUND
1003 REM CODE, POKING INTO PAGE 6.
1004 REM
1010 DIM BYTE$(2)
1020 READ ADDRESS
1030 READ BYTE$:IF BYTE$="**" THEN 2000
1040 BYTE=0
1050 V=ASC(BYTE$(1))-48:IF V>9 THEN V=V-7
1060 BYTE=BYTE*16+V
1070 V=ASC(BYTE$(2))-48:IF V>9 THEN V=V-7
1080 BYTE=BYTE*16+V
1090 POKE ADDRESS,BYTE
1100 ADDRESS=ADDRESS+1
1110 GOTO 1030
1500 DATA 1536
1510 DATA A9,0B
1520 DATA 8D,26,02
1530 DATA A9,06
1540 DATA 8D,27,02
1550 DATA 68
1560 DATA AD,0A,D2
1570 DATA 8D,00,D2
1580 DATA A9,A8
1590 DATA 8D,01,D2
1600 DATA AD,0A,D2
1610 DATA 4A
1612 DATA 4A
1614 DATA 4A
1616 DATA 4A
1618 DATA 09,01
1620 DATA 8D,18,02
1630 DATA 60
1640 DATA **
1650 REM
2000 REM ..........................
2001 REM
2002 REM THE FUNCTION IS IN PROTECTED
2003 REM MEMORY -- PRINT OUT THE
2004 REM APPROPRIATE USR COMMAND AND
2005 REM POSITION THE CURSOR SO THAT
2006 REM ONLY A CARRIAGE RETURN IS
2007 REM REQUIRED -- DO NOT LET THE
2008 REM USER USR TO THE WRONG PLACE !
2009 REM
2010 GRAPHICS 0
2020 PRINT "HIT RETURN TO START THE MUSIC !!"
2030 PRINT
2040 PRINT "X=USR(1536)"
2050 POSITION 0,0
```

Listing 4: Countdown Timer 1 Completion Routine

```
                   *=$0600          FREE SPACE IN BASIC
              ;
              ;    USER ROUTINE TO CHANGE VOICE #0 OF BASIC AS
              ;    A BACKGROUND TASK.  COUNT DOWN TIMER 1 IS USED.
              ;
              ;    DEFINE HARDWARE REGISTERS:
              ;
              RANDOM=$D20A          ATARI HARDWARE RANDOM NUMBER GENERATOR
              AUDF1=$D200           VOICE 0 FREQUENCY REGISTER
              AUDC1=$D201           VOICE 0 NOISE AND LOUDNESS REGISTER
              ;
              ;    DEFINE COUNT DOWN TIMER 1
              ;
              CDTMA1=$0226          COMPLETION ROUTINE ADDRESS
              CDTMV1=$0218          COUNT DOWN VALUE ADDRESS
              ;
              ;    INITIALIZE COMPLETION ROUTINE ADDRESS AND FALL
              ;    THROUGH TO ROUTINE TO START IT ALL.  THE INITIALIZATION
              ;    IS CALLED AS:  USR(1536).  THE COMPLETION ROUTINE
              ;    IS CALLED BY VBLANK AS A NORMAL SUBROUTINE.
              ;
A9 0B         INIT  LDA   #TIMER&$FF     LO BYTE OF ADDRESS
8D 26 02            STA   CDTMA1         TO CDTM1 COMPLETION ADDR
A9 06               LDA   #TIMER/256     HI BYTE OF ADDRESS
8D 27 02            STA   CDTMA1+1       COMPLETION ADDRESS INITIALIZED
68                  PLA                  DROP # OF USR ARGUMENTS
              ;
              ;    COUNT DOWN TIMER 1 COMPLETION ROUTINE -- SIMPLY
              ;    MODIFY THE SOUND OF VOICE 0 TO PROVE THAT THIS
              ;    WAS EXECUTED, AND RESTART TIMER FOR THE NEXT CHANGE.
              ;
AD 0A D2      TIMER LDA   RANDOM         GET A RANDOM BYTE
8D 00 D2            STA   AUDF1          SET AS FREQUENCY OF VOICE 0
A9 A8               LDA   #$A8           TONE=$0A; LOUDNESS=$08
8D 01 D2            STA   AUDC1          VOICE 0 CONTROL REGISTER
AD 0A D2            LDA   RANDOM         BYTE FOR TIMER DURATION
4A                  LSR   A              REDUCE TO 0-15 -- SOUNDS BETTER
4A                  LSR   A
4A                  LSR   A
4A                  LSR   A
09 01               ORA   #$01           MAKE CERTAIN TIME AT LEAST A ONE
8D 18 02            STA   CDTMV1         TIME OF ZERO IS TROUBLE
60                  RTS                  UNTIL NEXT CDTM1 COMPLETION
                    .END
```

The first two countdown timers, CDTM1 and CDTM2, allow a much more sophisticated timer control. To make use of this sophistication, assembly language programming is required. (The following discussion applies to CDTM2 as well as CDTM1, with an appropriate change in addresses.)

While VBLANK is servicing its interrupt, CDTM1 is checked for a non-zero value. If CDTM1 is non-zero, then CDTMV1 is decremented (counted down) by one. If CDTM1 goes to zero after being decremented, then the subroutine JTIMR1 is called. JTIMR1 simply jumps indirectly to the address stored at CDTMA1, the CDTM1 completion routine. Thus to use CDTM1, the following steps must be performed:

> POKE an assembly language subroutine into memory. This subroutine should perform a normal return *via* the RTS instruction.
>
> POKE the starting address (low, high) into CDTMA1 and CDTMA1 + 1
>
> start the timer by POKEing a value into CDTMV1
>
> continue to execute the BASIC program.

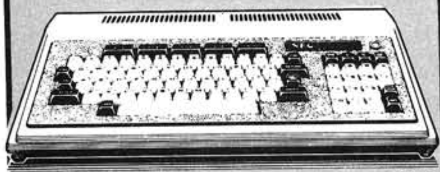
(Continued on page 82)

ORDER TOLL-FREE

consumer computers Mail Order DISCOUNTS

FREE SHIPPING*
*on all pre-paid cash orders

NEC
Microcomputer



PC-8001A 32K Computer	CALL
PC-8012A I/O Unit w/32K RAM	CALL
PC-8013A Dual Mini-Disk Drive Unit	CALL
PC-8001 Multi Cardware (FDI/O & 32K)	CALL
CP/M 2.2 Operating System for NEC	129
WordStar configured for NEC	299
SuperCalc configured for NEC	279
NEC Wordprocessor & Accounting Software	CALL

Many more software packages and languages; (Pascal, Fortran, Cobol, etc) are available configured for the NEC 8001A Computer.
Please call or write for a product price list.

ATARI

ATARI 800 16K
CALL FOR BEST PRICE

Atari 400 w/16K	349
410 Program Recorder	65
810 Disk Drive	449
825 80 col. 7x8 Dot matrix impact printer	699
822 40 col. Quiet Thermal Printer	349
850 Interface Module	159
Atari 16K Ram Module	69
Axlon Ramcram 32K Module	189
Asteroids, Missile Command and Star Raiders	35 ea.

SPECIAL ATARI CATALOG AVAILABLE

Video Monitors

Amdek/Leedex Video 100 12" B&W	149
Amdek/Leedex Video 100G 12" Green Phospher	179
Amdek (Hitachi) 13" Color w/audio output	389
NEC 12" Green Phospher Display JB-1201M	CALL
NEC 12" Lo-Res Color Display	CALL
NEC 12" Hi-Res RGB Color Display	CALL
Sanyo 9" B&W Display	185
Sanyo 9" Green Phospher Display	CALL
Sanyo 12" B&W Display	269
Sanyo 12" Green Phospher Display	285
Sanyo 13" Color Display	449
Zenith 12" Green Phospher Display ZVM-121	149

ZENITH 12" GREEN

$149

S-100
California Computer Systems

Floppy Disk Controller	**$369**
64K Dynamic Ram Board, 200ns	**$499**
Z-80 CPU board w/monitor ROM	**$269**
16K Static memory board, 200ns	369
32K Static memory board, 200ns	599
S-100 12 Slot Mainframe	475
4-Port Serial Interface	299
2-Port Serial/2-Port Parallel Interface	299
4-Port Parallel Interface	229

commodore
VIC20™

Personal Computer

Color · Sound · Graphics
Call or write for more info.
Disk drives available soon!

$259

Printers

Silentype
w/Apple II interface
$349

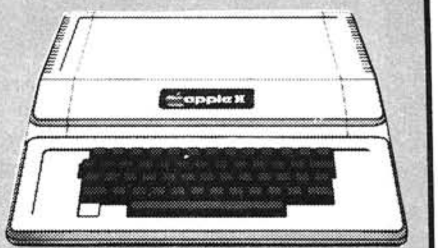
Epson
MX-80 or
MX-80 FT
CALL

Anadex 9501 w/2K Buffer	1349
C. Itoh Starwriter 25 CPS daisywheel	1449
C. Itoh Starwriter 45 CPS daisywheel	1649
Epson MX-70	CALL
Epson MX-80 & MX-80 F/T	CALL
Epson MX-100	CALL
NEC 8023 Impact Dot Matrix	695
NEC Spinwriters (Latest models)	CALL
Paper Tiger IDS-445G w/graphics	699
Paper Tiger IDS-460G w/graphics	949
Paper Tiger IDS-560G w/graphics	1249
Silentype Printer w/Apple interface	349
Qume Sprint Daisywheels (Latest models)	CALL

ORDER TOLL FREE
800-854-6654

In California and outside continental U.S.
(714) 698-8088
Telex 695-000 Beta CCMO

apple computer
Authorized Dealer



APPLE II PLUS
16K's, 48K's, 64k's*

*48K Apple with 16K RamBoard

CALL FOR BEST PRICES

APPLE DISK DRIVES

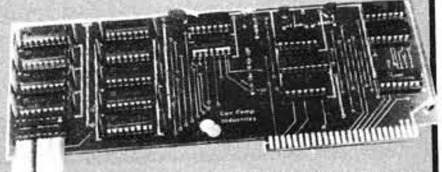
DRIVE ONLY OR W/CONT & DOS 3.3

CALL FOR PRICES

SPECIAL APPLE CATALOG AVAILABLE

16K RAMBOARD *by ConComp*
for Apple II Computers

FOR ONLY **$129.95**



AVAILABLE NOW

Ordering information: Phone orders using VISA, MASTERCARD, AMERICAN EXPRESS, DINER'S CLUB, CARTE BLANCHE, bank wire transfer, cashier's 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, Calif. 92041

SPEED POWER EFFICIENCY *for* OSI

65D3 SYSTEMS

FBASIC: BASIC Compiler $155/$10

- **FAST** efficient machine code written with the ease of BASIC.
- **SPEED**-optimized, native-code compiler.
- **INTEGER** subset of OSI-supplied BASIC.
- **DISK** based: No problem with size of source or object files.
- **EXTENSIONS** to BASIC:
  - Simple interface to system hardware and software.
  - Direct access to 6502 registers.
  - Array initialization.
  - Optional absolute array locations.
  - WHILE and other structures.
  - Simple technique for combining compiler and interpreter advantages.
- **FULL** system: utilities (plus source), manual, and many useful examples.

R-EDIT: Edit any program or text with ease! $40

- **FULL CURSOR** control. Edit anywhere on the screen.
- **INSERT**, replace, add, delete.
- **ONE KEY** stroke and you're editing.
- **BASIC**, assembler, etc. can all be edited without reloading editor.
- **RAM**-resident: Always ready!
- **SYSGEN** relocates R-EDIT and customizes.

SPUL-65: Printer Spooler $95/$10 Virtual Indirect File

- **STOP WAITING** for your printer!
- **PROCESS** words, write programs...all while printing!
- **QUEUE** lets you pile on print jobs.
- **MULTIPLE COPIES** printed with top and bottom page margins.
- **SYSGEN** relocates SPUL65 and gives extensive customization.
- **INDIRECT FILE** commands produce disk files giving you:
  - A virtually unlimited temporary file.
  - A link between incompatible files; for example, use WP-2 for extensive BASIC editing.
  - Ability to merge multiple program segments.

XREF: BASIC Cross Referencer $25

- **TABULATES:**
  - Referenced line numbers.
  - Variable names (numeric, string, array).
  - Defined functions.
- **FAST** machine language program.
- **DISK** based: Handles large BASIC source files on any drive.

CP/M to OSI Disk Translation

Frustrated by all those good CP/M disks that won't run on your OSI CP/M system? It's that special OSI disk format! And we can fix that. Just send us your disk, $15, and you'll soon have an OSI compatible disk.

drc software

Data Resource Corporation
Suite 204
1040 Lunaai Street
Kailua, HI 96734 (808) 261-2012

Manual orders applied to software purchases. Programs supplied on 8-in, single-density, single-sided disks. Hawaii residents add 4% tax.

Speedy Routine for C1P Joystick

Take the "slow motion" out of your real-time games with this speedy joystick input routine for your C1P. Application driver routines are also provided.

John Krout
5108 N. 23rd Rd.
Arlington, Virginia 22207

Before I bought my computer, I used to spend a lot of pocket change and time in the local video arcades. Now I keep the change and invest the time in attempts to surpass the latest commercial amusement electronics in software at home. Like many computer users, I enjoy developing realtime graphics games almost as much as playing them.

A major problem with games that use interpretive BASIC is execution speed. Everyone recognizes that a faster game is a greater challenge to the player. BASIC on the Challenger 1P stands up well in speed comparisons with any other machine on the market at a similar price. But BASIC software, which juggles a number of moving elements on screen, often decelerates past the threshold of boredom.

Joysticks are the usual input devices for the realtime games I develop. A trigger joystick such as those used in the Atari cartridge video game can be easily connected to the keyboard port of the Challenger 1P. Many software vendors offer inexpensive instructions for this hardware modification. To the computer, each joystick looks as if it were a row of keys, and the status of its triggers can be sensed by examining a keyboard row as described in the Challenger 1P documentation.

Listing 1 is an example of a joystick examining routine in BASIC. The value of X, set in line 10, selects the row of keys scanned and, therefore, the joystick used. In this case, numeral keys 1 through 7 are scanned. The POKE in line 20 disables the BASIC overhead routine which scans the keyboard for a Control C program break by the player. Line 40 POKEs X to the keyboard port at location 57088, and sets the variable P equal to the result value PEEKed at the same location. The value of P reflects which keys, if any, have been depressed, or joystick triggers closed, by the player when line 40 is executed. Lines 30 and 50 set up a loop which repeats the keyboard scan in line 40 for execution speed comparison, and line 60 re-enables the Control C routine.

Listing 1

```
10 X=127
20 POKE 530,1
30 FOR I=1 TO 2000
40 POKE 57088,X:
   P=PEEK(57088)
50 NEXT
60 POKE 530,0
```

Listing 2

```
 5 0222                START *=$222
10 0222 203202         JSR $232
20 0225 A5AF           LDA 175
30 0227 8D00DF         STA 57088
40 022A AC00DF         LDY 57088
50 022D A900           LDA #0
60 022F 6C0800         JMP (8)
70 0232 6C0600         JMP (6)
```

Listing 3

```
10 DATA 32,50,2,165,175,
        141,0,223
20 DATA 172,0,223,169,0,
        108,8,0
30 DATA 108,6,0
40 FOR I=1 TO 19
50 READ A: POKE I+545,A
60 NEXT
70 POKE 11,34: POKE 12,2
```

Listing 4

```
10 X=127
30 FOR I=1 TO 2000
40 P=USR(X)
50 NEXT
```

Listing 2 is an assembly language subroutine which may be called from BASIC via the USR(X) function to replace the POKE and PEEK in line 40 of listing 1. This routine accomplishes four steps: transferring the value of the argument variable X from BASIC, POKEing the argument to the keyboard port, PEEKing at the keyboard port for the result value, and transferring the result value back to BASIC. It does all of this without the need for BASIC POKEs to turn off and on the Control C routine.

Lines 10 and 70 of listing 2 transfer the argument value from BASIC. This is done by calling the routine in ROM to which the address in locations 6 and 7 point. Since the JSR opcode does not include an indirect addressing option, line 10 calls line 70, which is simply a jump using indirect addressing to get to the ROM routine. When the processor encounters an RTS opcode in ROM, it will resume execution at line 20.

The ROM routine places the argument value in locations 174 and 175. The format is a 15-bit integer with a sign bit. This means that arguments greater than 32767 or less than −32768 will generate an error code in BASIC. The routine stores the sign bit and most significant 7 bits in location 174, and the least significant 8 bits in location 175.

Since the argument value for the keyboard port ranges from 0 to 255, only the least significant 8 bits of the argument are needed by the routine of listing 2 for further use. Therefore, line 20 loads the data in location 175 to the accumulator, and line 30 stores that value from the accumulator to the keyboard port. The value placed in location 174 by the ROM routine is ignored.

To return a result value to BASIC, another ROM routine is called via the indirect jump in line 60. This routine treats the data in index register Y as the least significant 8 bits, and the data in the accumulator as the sign bit plus the most significant 7 bits, of the integer value to be returned. Since the result in this case also ranges from 0 to 255, line 40 loads the result value from the keyboard port to index register Y, and line 50 loads a zero into the accumulator prior to the jump. When the processor again encounters an RTS opcode in ROM, it will return directly to the BASIC program, so no other RTS is necessary in listing 2.

Listing 3, a BASIC loader, places the decimal values (in DATA statements) of the assembled opcodes of listing 2 into memory beginning at location 546. This is the start of the unused memory space which ends at location 767. However, listing 3 could place the routine elsewhere in memory if space is initially reserved for it. The top of memory can be reserved for the routine by POKEing the start address of the reservation into locations 133 and 134 in low-high format. The high part is the address divided by 256, and the low part is the remainder. For instance, to reserve memory starting at location 8000:

```
5 POKE 133,64: POKE 134,31:
  POKE 129,64: POKE 130,31
```

will do the trick. Keep in mind that listing 2 is not relocatable, so the second and third values of data will have to be changed to match the new address of JMP (6).

Listing 5

```
10 DATA 32,50,2,165,175,141,0,223
20 DATA 172,0,223,169,0,108,8,0
30 DATA 108,6,0
40 FOR I=1 TO 19
50 READ A: POKE I+545,A
60 NEXT
70 POKE 11,34: POKE 12,2
80 FOR I=1 TO 32: PRINT: NEXT
90 INPUT"start address";X: PRINT
95 PRINT"address: data": PRINT
100 Y=PEEK(X)
105 Z$=STR$(X)+":"+STR$(Y)+"   ": L=LEN(Z$)
110 FOR I=1 TO L: POKE 54119+I, ASC(MID$(Z$,I,1)): NEXT
115 FOR I=1 TO 250: NEXT
120 P=USR(127): IF P=255 GOTO 120: REM stick centered
125 IF Y=255 GOTO 140
130 IF P=191 THEN Y=Y+1: POKE X,Y: GOTO 105: REM stick up
135 IF Y=0 GOTO 145
140 IF P=239 THEN Y=Y-1: POKE X,Y: GOTO 105: REM stick
    down
145 IF X=0 GOTO 160
150 IF P=247 THEN X=X-1: GOTO 100: REM stick left
155 IF X=65535 GOTO 165
160 IF P=223 THEN X=X+1: GOTO 100: REM stick right
165 IF P=127 THEN FOR I=1 TO 500: NEXT: END: REM trigger
    pressed
170 GOTO 120
```

Line 70 of listing 3 informs BASIC that the function USR(X) calls the routine beginning at location 546. The address of the USR(X) entry point is stored in low-high format by POKEs to locations 11 and 12. If you've had trouble implementing USR(X) routines in the past, it may be because the

Olympic Sales Company

SERVING YOU SINCE 1947

Main Showroom & Offices:
216 South Oxford Avenue
Los Angeles, CA 90004

WE HONOR
VISA and MASTERCHARGE

TELEX: 67 34 77
ORDER DESKS open 7 Days a Week!
7:00 AM to 7:00 PM Mon thru Sat
Sunday Noon to 5:00 PM
Order Desks: (213) 739-1130

TOLL-FREE (outside Calif.) 800-421-8045
TOLL-FREE (within Calif.) 800-252-2153

NO ONE UNDERSELLS OLYMPIC SALES

Write & request our new 112 pg catalog-please include $1.00 to defray postage & handling-includes many more items-from TV's to Watches!

All goods subject to availability; this ad supersedes all previous ads; we are not responsible for typographical errors; we will meet or beat any advertised prices if the competition has the goods on hand.
Minimum shipping and handling $4.95.
All orders subject to verification and acceptance.

apple computer

We are an authorized servicing Apple dealer for Apple II & III
Immediate delivery on all models—we carry an enormous inventory of Apple products at all times!

Immediate delivery
16K—32K—48K—64K—Plus or Integer in stock!
NEW, IMPROVED APPLE III 128K VERSION

Large Inventory of:
Disk with controller DOS 3.3
Second Disk Drives
Pascal—Fortran—Cobol languages
Dow Jones & Quote reporter
Graphics Tablets
Visicalc for Apple II & III
Smarterm 80 column card
Micromodem II by DC Hayes
and more. . .

ATTENTION: Immediate delivery
WE ALSO CARRY SOFTWARE!
Personal Software
Peach Tree Software
Microsoft
Micropro
Innovative
American
System Plus
and more. . .

NOW IN STOCK!

HEWLETT PACKARD
2 NEW DELUXE CALCS FROM HP!
Slim, shirt-pocket styling

NEW! HP-11C Advanced Programmable Scientific LCD *Retail: $130.00* — *Your Cost:* $114.95
NEW! HP-12C Advanced Programmable Financial LCD *Retail: $150.00* — *Your Cost:* $129.95

	Retail	Your Cost
HP-125 New Microcomputer 64K CPU/Terminal/Keyboard	3750.00	3195.00
HP-85 Microcomputer—built-in printer/monitor	3250.00	2499.00
HP-83 Microcomputer—built-in monitor	2250.00	1799.00
HP-2631B Printer, dot mtrx (ask for optns)	3950.00	3199.00
HP82905A 80 col printer, dot matrix	945.00	755.00
HP-2601A Letter quality prter, daisy wheel		3495.00
HP-82901M Dual master (256KB disk drive)	2500.00	1999.00
Call us for the lowest prices on 7 disk drives	Call us	Call us
HP-41CV New! 2.2K bytes of memory	325.00	256.00
Card Reader for 41CV	215.00	171.00
Printer for 41CV	385.00	294.00
Quad Ram	95.00	84.95
Optical Wand for 41CV	125.00	99.95
HP-41C Calculator	250.00	188.95
Memory mod. for 41C		26.95
HP-97 Progrmble printer	750.00	579.95
HP-34C Progrmble scientific	150.00	117.95
HP-38C Progrmble business R/E	150.00	119.95

We have the complete line of accessories, etc.

XEROX
Model 820-1 (5¼") 8" $3750.00
64K COMPUTER & WORD PROCESSOR
AS LOW AS $2995.00
Special discount available to Schools & Institutions—Inquire! Required software add'l.

NEW—FAMOUS CORVUS DISK DRIVES—5, 10, 20 MEGABYTES with fantastic new OMNINET Network System
Call us for the best prices in the USA!

Texas Instruments Home Computer
New—1982 Model with full typewriter-style keyboard, U/L case & more!
TI-99/4A New KEYBOARD $369.95 — *Retail* $950.00 *Your Cost* $369.95
New keyboard

	Retail	Your Cost
10" color monitor for 99/4	650.00	319.95
32K Exp. mem. module	399.95	314.95
Extended Basic, a MUST for 32K module	100.00	75.00
Speech synthesizer	150.00	129.95

We carry a large inventory of software, & accessories

	Your Cost
TI-30-2 LCS Stu Slide NEW	18.95
TI-35SP LCD SCIENTIFIC	22.50
TI-40 LCD Sci/NEW	28.95
TI-57 Prog. Scientific	39.95
TI-58C 480 Step. Prog.	89.95
TI-59 960 Step. Prog.	179.95
PC-100C Print/Plot	149.95
LCD-Programmer/NEW	59.95

ATARI Computer — Large inventory of periphls, access. etc.

		Retail	*Your Cost*
400	SPECIAL PRICE! 16K	595.00	339.95
	No language inc., opt'l basic,		54.95
800	16K Computer	1080.00	759.95
	SPECIAL! ATARI 800 48K Computer	1250.00	869.95

OHIO SCIENTIFIC
C8PDF—48K Retail: $3495.00 Y/C: $3195.00
- Dual 8" Drives
- 64 col x 32 line/color
- .7 MIPS — *FAST!*
- Many more stnd features

Fortran & Pascal available
Many other OSI products—at discounted prices

NEW! From TI—Series 10 Personal Information Terminal *Retail* 995.00 *Your Cost* 795.00

PRINTERS

• DIABLO (Letter Quality)	Retail	Your Cost
630 R102 bi-directnal/tractors	2965.00	2699.00
1640K109 keyboard, tractors	3072.00	2899.95
630 RO Receive only	2710.00	2499.95
1650K136 keyboard/tractors	3220.00	2999.95
• CENTRONICS dot matrix		
700-9 Parallel, heavy duty	1460.00	1199.95
704-9 Serial, heavy duty	1795.00	1599.95
737-1 Parallel	995.00	799.95
737-3 Serial	1045.00	899.95
704-11 Parallel	1870.00	1695.00
P-1 Electrostatic	495.00	189.95
• PAPER TIGER		
460	995.00	895.00
460G graphics	1094.00	969.95
560	1295.00	1099.00
560G graphics	1394.00	1195.00
445	795.00	695.00
445G	894.00	789.00

EPSON PRINTERS	Retail:	Your Cost
MX80	645.00	539.95
Optional Graftrax Chip 80		95.00
MX80 FT	745.00	659.95
MX80 + GRAFTRAX 80	695.00	579.95
MX80 FT + GRAFTRAX 80	795.00	689.95
MX100	995.00	789.95

WE ALSO HAVE . . .

• NOVATION Modems	Retail	Your Cost
CAT	199.95	159.95
D-CAT	199.95	159.95
APPLE-CAT Direct connect	349.95	314.95

SANYO MONITORS High resolution

13" Color (new) high quality	550.00	419.95
12" Green phosphorous	360.00	259.95
12" Black & white	340.00	239.95
15" Black & white	370.00	259.95
9" Black & white (the best seller)	235.00	169.95

AMDEK(Leedex) High Quality Monitors

100	12" B/W, 12 MHz	179.00	139.95
100-G	12" Green, 12 MHz	199.00	174.95
300-G	12" Green, 18 MHz	249.00	199.95
Color I	13" Color, NTSC comp. input, audio amp & speaker	449.00	339.95
Color II	13" Color, RBG input, hi res graphics, speaker	999.00	699.95

- HAZELTINE Video Display Terminals
- SHUGART Disk Drives
- DEC VT100 & VT103

Call us for your DISCOUNTED price TODAY!

Challenger 1P BASIC documentation published prior to 1981 stated incorrect locations for USR(X) entry point storage.

A BASIC program is shown in listing 4 which uses the USR(X) subroutine placed in memory by execution of listing 3. When listing 4 is executed, the value of P in line 40 is the same as that of P in line 40 of listing 1. You can confirm this by adding to listings 1 and 4:

```
45 PRINT P
```

so you can see the values of P generated as keys 1 through 7 are depressed.

Without the extra line 45, listings 1 and 4 can be timed, when executed, to determine the speed improvement for joystick input obtained through the USR(X) routine. On a Challenger 1P with the standard processor clock rate, listing 1 executed in 26 seconds and listing 4 executed in 6 seconds.

As useful as fast joystick input may be in game programs, never let it be said that joysticks are of no use elsewhere. For instance, listing 5 replaces and enhances some functions of the monitor ROM in the Challenger 1P.

The monitor allows you to examine and change individual bytes, but it displays addresses and corresponding data only in hexadecimal. Also, by holding down the Return key while in data mode, you can view consecutive addresses counting up. The counting speed is fixed at a rate which allows only the speediest reader to examine each byte in this manner, so most of us have to tap Return repeatedly when we are looking for a particular value somewhere in a certain range of addresses.

Listing 5 uses a joystick connected to the uppermost keyboard row to control a decimal display of addresses and data. Pushing the stick to the right will increment the address, and to the left will decrement the address. The data in an address can be incremented by pushing the joystick up, and decremented by pushing the joystick down. The program is ended by pushing the joystick trigger.

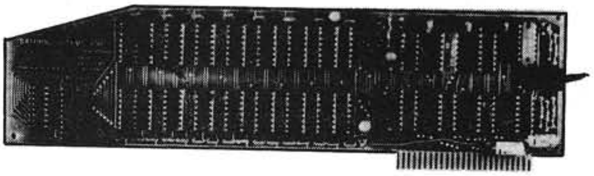
Lines 10 through 70 are the loader of listing 3. Line 80 clears the screen, and lines 90 through 100 set up the initial display parameters. The variable X is the current address, and the variable Y is the data stored at address X. Lines 105 and 110 create a character string based on X and Y, and POKE that string to the screen.

Line 115 is a delay loop which is executed after each display. If you prefer faster or slower operation of the program, decrease or increase the terminal value of the loop.

Line 120 calls the fast joystick input routine and stores the result value in variable P. If the joystick is not pressed, the routine is called again without altering the display. Line 130 increments and stores Y if the joystick is pushed up. Since the maximum value of data is 255, line 125 prevents Y from being incremented beyond that limit. Likewise, line 135 prevents Y from being decremented past zero, and line 140 decrements and stores Y if the joystick is pressed down.

The limits of X are zero and 64K, expressed in lines 145 and 155. Line 150 decrements X if the joystick is pushed left, and line 160 increments X if the joystick is pushed right. Finally, if the trigger is pressed, line 165 executes a delay loop to avoid keybounce and the program ends.

Whether your Challenger 1P simply satisfies a craving for arcade games, or is the focus of more serious developments, the joystick is a most useful input device. Now it will go to work for you more quickly than before.



MICRO

Short Subjects

List-Corrupted SYM-bols

Len Green, 15 Yotam Street, Achuza
34 675 Haifa, Israel

The SYM monitor has a built-in command to fill any length block of RAM with a specified byte. For example, FILL AA-1000-2FFF fills pages 10 through 2F with the byte AA. Presumably other 6502 users either possess extended monitors which have this facility, or have made their own very short and simple routines for performing same.

SYM also has a monitor command for listing a specified byte. There are innumerable routines in assemblers for listing individual bytes and/or strings. Although I assume that many simple programs must exist, I have not seen one which performs the REVERSE operation; i.e., lists all bytes which are not a particular specified one. This is also extremely useful in practice!

This program was one of my very early ones. Although less than 50 bytes long, it is probably not completely optimized. Since I use it almost daily in several different ways, it was one of the first programs that I customized and included when I eventually graduated to burning my own Utility EPROMs. (Then I found it additionally helpful in checking stages of preparation for EPROMing, and checking the erased portions of the EPROMs themselves.)

This program is fully relocatable without alteration; e.g., to EPROM. Its checksum from $1000 to $1030 is #1D94. It uses zero-page locations $F0 to $F4 inclusive, which don't clash with SYM's BAS-1 or RAE-1, but any five will do. The functions of these locations have been fully outlined in the listing, together with those of the four fairly standard SYM monitor routines.

Before starting any complicated programming, I generally fill all available RAM with the byte FF. (I have a very short routine for automating this process, including the relevant page 0 and 1 regions.) At a later stage, I can very easily check whether any quiescent areas of memory have been tampered with. This is convenient for keeping track of the several different zones of the Resident-Assembler-Editor and BASIC with their multitudinous patches, and for testing whether any rogue-bytes have been introduced anywhere, or any zero-page addresses altered unintentionally.

To check, for example, whether a 2716 EPROM from $F000 to $F7FF has been fully erased (#FF), fill $F0 to $F4 with FF 00 F0 FF F7 and then GO/1000/CR. Un-erased locations will be displayed in the form:

```
F017,EF
F48A,FE
F7BD,00 etc.
Prompt
```

If you haven't invented an equivalent routine before me, I hope you will find this one as useful as I have.

```
0800            ;*
0800            ;* LIST CORRUPTED SYMBOLS
0800            ;*
0800            ;*      BY LEN GREEN
0800            ;*
1000                    ORG $1000
1000            ;
1000                    OBJ $800
1000            ;
1000            BYTE    EPZ $F0                 ;TEST BYTE ('SYM'-BOL)
1000            BEGAD   EPZ $F1                 ;BEGIN ADDRESS, LOW
1000            ENDAD   EPZ $F3                 ;END ADDRESS, LOW
1000            ;
1000            ;SYM MONITOR ROUTINES
1000            ;
1000            CRLF    EQU $834D               ;PRINT CRLF
1000            COMMA   EQU $833A               ;PRINT 'COMMA'
1000            OUTBYT  EQU $82FA               ;PRINT HEX BYTE IN ACC.
1000            OUTXAH  EQU $82F4               ;PRINT 2 HEX BYTES(FROM X & A)
1000            ;
1000 A200       START   LDX #$00
1002 A1F1               LDA (BEGAD,X)           ;GET BYTE
1004 C5F0               CMP BYTE                ;CORRECT
1006 F014               BEQ OK                  ;BYTE (SYM-BOL?)
1008 204D83             JSR CRLF                ;ELSE PRINT LINE
100B A6F2               LDX BEGAD+1             ;ADDRESS-HI TO X
100D A5F1               LDA BEGAD               ;ADDRESS-LO TO A
100F 20F482             JSR OUTXAH              ;PRINT X & A
1012 203A83             JSR COMMA               ;PRINT COMMA
1015 A200               LDX #$00
1017 A1F1               LDA (BEGAD,X)           ;LOAD ERROR BYTE
1019 20FA82             JSR OUTBYT              ;AND PRINT IT
101C A5F2       OK      LDA BEGAD+1
101E C5F4               CMP ENDAD+1             ;HI-ADDRESS END?
1020 D007               BNE INCAD
1022 A5F1               LDA BEGAD               ;ELSE CHECK LO-ADD
1024 C5F3               CMP ENDAD               ;LO-ADDRESS END?
1026 D001               BNE INCAD
1028 60                 RTS                     ;END OF ROUTINE!
1029            ;
1029 E6F1       INCAD   INC BEGAD               ;INC LO-BEG-ADD
102B D002               BNE CONT                ;LO-ADDRESS =0?
102D E6F2               INC BEGAD+1             ;INC HI-BEG-ADD
102F 50CF       CONT    BVC START               ;ALWAYS
1031            ;
                        END
```

Applesoft Input Anything Revisited

Lee Reynolds, 5760 NW 60th Ave.
Apt. B-101, Fort Lauderdale, FL 33319

Most of the "input anything" routines for characters like the comma and colon in Applesoft, depend upon using a machine language routine, which directly uses the monitor's character input logic. This routine is different because it uses Applesoft's GET. This method offers some advantages over the use of machine language in that the program can more directly control what is displayed on the screen, and can process certain types of input in special ways.

I wrote the routine listed here for use at work (accounts receivable, inventory control, etc.). It demonstrates some programming practices which readers might like to emulate in writing user-friendly programs.

Before you use the routine, you must declare the string array Z1$ by a statement like:

```
10 DIM Z1$(1): Z1$(0) = “#”:
  Z1$(1) = CHR$(34)
```

An example of using the routine:

```
1000 VTAB 5: HTAB 1: PRINT
  “CUSTOMER NUMBER ";:
  N=4: X=0: GOSUB 200
```

The customer number will be returned in the string XX$ following the GOSUB 200. The arguments N and X passed to the routine are the maximum

Here is the listing of the routine:

```
   200 XX$ = "": IF N THEN PRINT Z1$(X);: FOR I = 1 TO N:
CALL -1036: NEXT:  PRINT Z1$(X);: FOR I = 1 TO N+1: PRINT CHR$(8): NEXT
   210 GET A$: IF A$ = CHR$(13) THEN PRINT: RETURN
   220 IF A$ <> CHR$(8) THEN 260
   230 L = LEN(XX$)-1: IF L>0 THEN PRINT A$;: GOTO 250
   235 IF L = 0 THEN PRINT A$;
   237 IF L < 0 THEN CALL -198
   240 XX$ = "": GOTO 210
   250 XX$ = LEFT$(XX$,L): GOTO 210
   260 IF LEN(XX$) = 0 AND ASC(A$) = 20 THEN INVERSE:
PRINT "T";: NORMAL:  GOTO 280
   263 IF N >0 AND LEN(XX$) = N THEN CALL -198: GOTO 210
   265 IF A$ = CHR$(21) THEN XY = PEEK(36): YZ = PEEK(37): XX =
SCRN(XY,2*YZ) + 16*SCRN(XY,2*YZ+1)-128: A$ = CHR$(XX): GOTO 275
   270 IF ASC(A$) < 32 OR A$ = ":" OR A$ = "," THEN 210
   275 PRINT A$;
   280 XX$ = XX$ + A$: GOTO 210
```

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

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

L I S P

for the Apple II

The gnosis version of P-LISP has been acknowledged as the finest and most complete available for Apple microcomputers, and, with the addition of floating point math and HI-RES graphics, it becomes an indispensable tool for educators, scientists, business executives, mathematicians, or applications requiring artificial intelligence. This excellent program is now available for only $199.95 (DOS 3.3 only).

Included in an attractive binder is a ninety page user's manual which will aid you in creating your P-LISP programs. This manual is also available separately for $20.00, which is fully refundable on purchase of the program.

P-LISP will run on a 48K or larger APPLE II/II+, and will take advantage of ALL available memory. Supplied with the interpreter are several sample programs including a complete ELIZA.

For those of you who do not fully understand P-LISP, we have available the P-LISP Tutorial for $25.00. This expertly written text is bound in a handsome binder and is packaged to include a disk containing all the sample programs referenced in the text at no extra charge.

Applesoft in ROM or a language card is needed for floating point math

gnosis
formerly Pegasys Systems, Inc.
4005 Chestnut Street—Philadelphia, PA 19104
Orders Only: 800-523-0725—Penna. Residents: 215-387-1500
Pennsylvania residents add 6% sales tax Apple is a trademark of Apple Computer Inc

number of characters to be input and the type of input prompt to print. If X = 0, then the program is expecting the user to type in a number. The prompt would look like this on the screen at line 5:

```
CUSTOMER NUMBER #   #
```

Notice that the # signs delimit the area where the user is allowed to type his answer. When X = 1, the program is expecting alphanumeric input (any characters, not necessarily numeric), so the delimiters for what is to be input are quote marks, as in:

```
1010 VTAB 7: HTAB 1: PRINT
  "ADDRESS ";: N=24: X=1:
  GOSUB 200
```

The screen looks like this on line 7:

```
ADDRESS "          "
```

You can also pass N = 0 to the routine, in which case there will be no delimiters printed. Here is an example of how we use this option:

```
2000 VTAB 10: HTAB 1: PRINT
  "DATE MM/DD/YY";: HTAB
  7: N=0: GOSUB 200
```

In this case, the user is expected to type in the date over the prompt "MM/DD/YY" on the screen.

Another special feature that this routine incorporates is one I have found very useful in file-maintenance-type functions, such as adding, changing and deleting customers from the Customer File. Before doing the GOSUB 200, the program displays a message "(TYPE CTRL-T TO TERMINATE)" at line 23 on the screen. This tells the program user that when he is finished adding, changing, or deleting customers he should reply to the "CUSTOMER NUMBER" question by typing control-T (and carriage return). The routine will display an INVERSE T on the screen so that the user can easily verify what was typed. Then special logic after the GOSUB 200 terminates the add/change/delete process.

Here are a few more notes on how the routine operates:

200 The CALL −1036 moves the cursor forward inside the loop so that on the "Change" function for the user's input, he can see what he originally typed, and just copy over what might already be correct, using the forward arrow key.

210 Testing for CHR$(13) is to end input when carriage return is typed by the user.

220 CHR$(8) is the back-arrow.

230 If the user has not attempted to backspace past the beginning of the field, we perform the backspace and truncate the input.

237 If the user tries to backspace before the field beginning, then CALL −198 rings the bell.

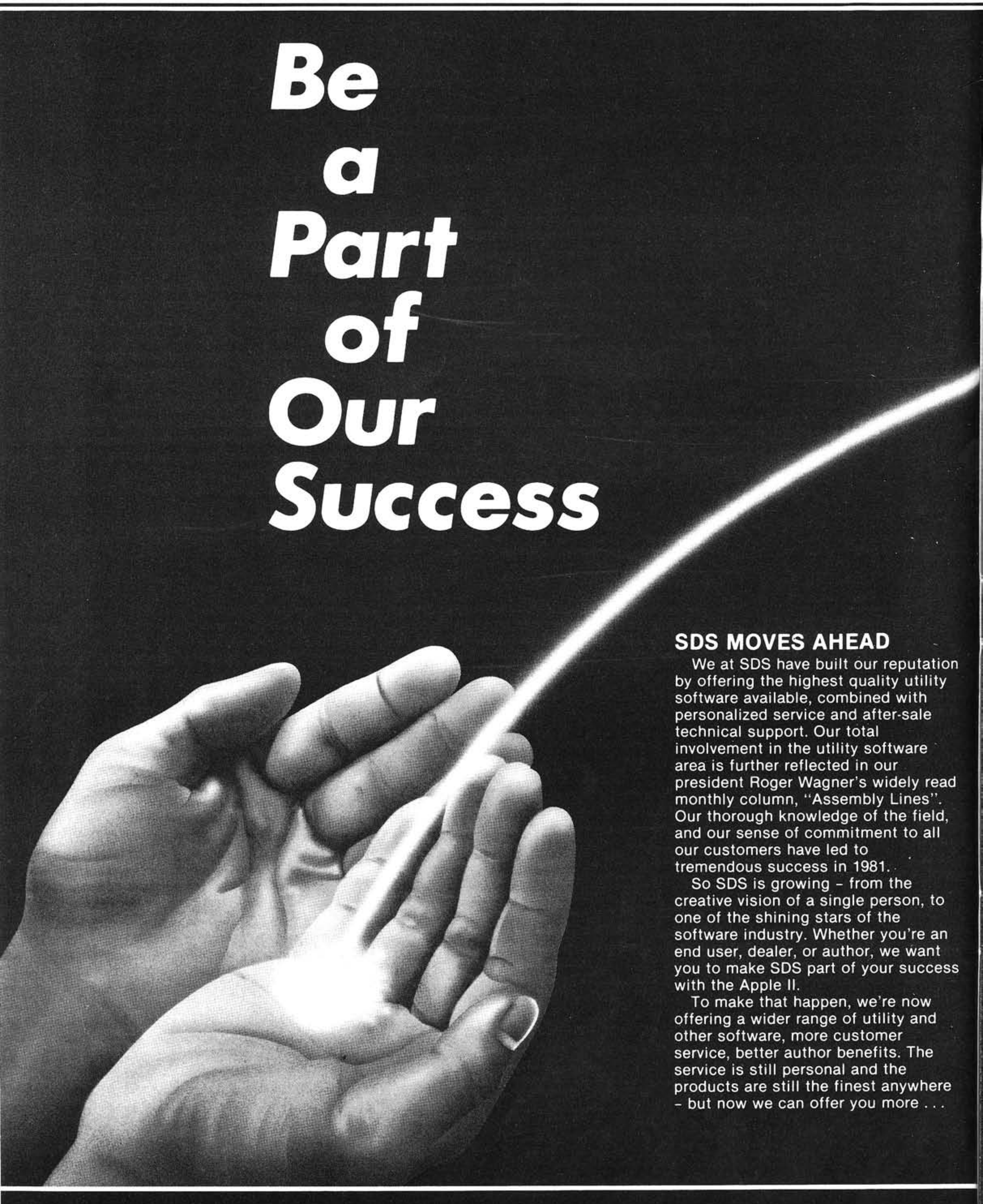
260 If control-T is the first typed character, print INVERSE T.

263 If the user exceeds the right margin of the field, ring bell.

265 If the key typed is forward arrow, get character beneath cursor by means of formula on page 87 of Applesoft manual, and add it to input string.

270 If the user types any other control-character, or a comma or colon, ignore it.

MICRO



Be a Part of Our Success

SDS MOVES AHEAD

We at SDS have built our reputation by offering the highest quality utility software available, combined with personalized service and after-sale technical support. Our total involvement in the utility software area is further reflected in our president Roger Wagner's widely read monthly column, "Assembly Lines". Our thorough knowledge of the field, and our sense of commitment to all our customers have led to tremendous success in 1981.

So SDS is growing – from the creative vision of a single person, to one of the shining stars of the software industry. Whether you're an end user, dealer, or author, we want you to make SDS part of your success with the Apple II.

To make that happen, we're now offering a wider range of utility and other software, more customer service, better author benefits. The service is still personal and the products are still the finest anywhere – but now we can offer you more . . .

southwestern data systems™

MORE NEW PROGRAMS

ROUTINE MACHINE - A terrific way to put machine language in your Applesoft programs. Simply choose from a pre-written library of subroutines. No knowledge of machine language is required.

AMPERSOFT PROGRAM LIBRARY - A set of additional machine language subroutines for use with Routine Machine.

MERLIN - A powerful, full-featured 6502 macro assembler.

ASCII EXPRESS: THE PROFESSIONAL A professional version of our best selling communication package that transfers files of ANY size or type. Supports all 80 column boards, interactive macros, and much more.

P-TERM: THE PROFESSIONAL - A Pascal version of Bill Blue's highly acclaimed communication software.

DOUBLETIME PRINTER - Increase your Apple's productivity by simultaneously printing text or binary files from one program while you run other programs.

UNIVERSAL GRAPHICS - The ultimate Hi-Res printing utility to transform your creations into permanent records on ANY graphic printer. Includes picture cropping, inverse inking, horizontal or vertical printouts, and a range of magnifications.

MORE DEALER SUPPORT

SDS is committed to the continuing support of the Apple dealer. As part of our new program, SDS offers:

- Informative point-of-sale materials
- Comprehensive national advertising
- Head Start Program to insure priority delivery of all new products
- Special dealer discount on in-store display items
- Comprehensive line of wide application software products
- Full-time technical support staff

SDS
southwestern
data systems™
P.O. Box 582M Santee, CA 92071
Tel. 714-562-3670

MORE AUTHOR BENEFITS

SDS is an industry pioneer in innovative software for the Apple II. When you join the SDS team, you get the benefits of a professional support staff experienced in providing all the help you need to market your program. Other benefits include:

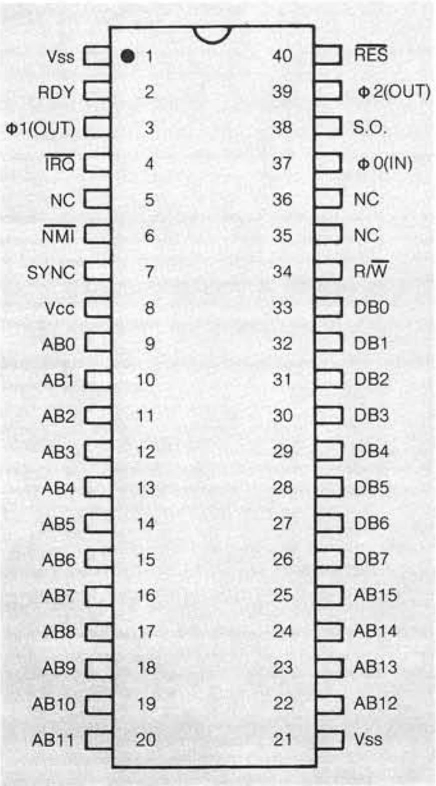
- Technical programming assistance
- Unique copy protection with limited backups
- Successful marketing strategies
- Assistance in writing the manual
- Professional product artwork
- Quality advertising
- Superior packaging
- National distribution
- Highest royalties paid monthly
- Customer service support

Here's the chance for you to market your program with the finest publisher in the software industry. Let Southwestern Data Systems' reputation and proven track record go to work for you. Be a part of our success - call today.

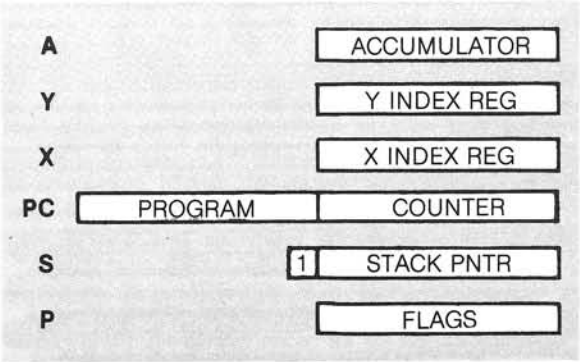
6502 Microprocessor

6502 — most prominent member of the 65XX family of microprocessors
- designed and manufactured by MOS Technology
- manufactured by Synertek and Rockwell
- used in Apple, AIM, SYM, KIM, OSI, PET, VIC, Atari, Acorn, and many other computers
- also used for many dedicated applications, including process controllers,arcade games, and Atari and Mattel home video games

6502 Pins



6502 Registers



Flags

MSB LSB

N	V	—	B	D	I	Z	C

N — negative result
V — overflow
B — BRK instruction
D — decimal mode
I — IRQ disable
Z — zero result
C — carry

Unsigned Comparisons

A < M	BCC	yes
A = M	BEQ	yes
A > M	BCC	no
	BNE	yes
A ≥ M	BCS	yes
A ≠ M	BNE	yes
A ≤ M	BCC	yes
	BEQ	yes

6502 Instruction Set

ADC	Add with carry	**DEC**	Decrement memory	**ROL**	Rotate left
AND	Logical AND	**DEX**	Decrement X	**ROR**	Rotate right
ASL	Arithmetic shift left	**DEY**	Decrement Y	**RTI**	Return from interrupt
BCC	Branch if carry clear	**EOR**	Exclusive OR	**RTS**	Return from subroutine
BCS	Branch if carry set	**INC**	Increment memory	**SBC**	Subtract with carry
BEQ	Branch if result = 0	**INX**	Increment X	**SEC**	Set carry
BIT	Test bit	**INY**	Increment Y	**SED**	Set decimal mode
BMI	Branch if minus	**JMP**	Jump	**SEI**	Set interrupt disable
BNE	Branch if result ≠ 0	**JSR**	Jump to subroutine	**STA**	Store accumulator
BPL	Branch if plus	**LDA**	Load accumulator	**STX**	Store X
BRK	Break	**LDX**	Load X	**STY**	Store Y
BVC	Branch if overflow clear	**LDY**	Load Y	**TAX**	Transfer A to X
BVS	Branch if overflow set	**LSR**	Logical shift right	**TAY**	Transfer A to Y
CLC	Clear carry	**NOP**	No operation	**TSX**	Transfer SP to X
CLD	Clear decimal mode	**ORA**	Logical OR	**TXA**	Transfer X to A
CLI	Clear interrupt disable	**PHA**	Push A	**TXS**	Transfer X to SP
CLV	Clear overflow	**PHP**	Push P status	**TYA**	Transfer Y to A
CMP	Compare to accumulator	**PLA**	Pull A		
CPX	Compare to X	**PLP**	Pull P status		
CPY	Compare to Y				

	ACCUMULATOR / IMPLIED*	IMMEDIATE	ZERO PAGE	ZERO PAGE INDEXED BY X	ABSOLUTE	ABSOLUTE INDEXED BY X	ABSOLUTE INDEXED BY Y	INDIRECT INDEXED BY X	INDIRECT INDEXED BY Y	FLAGS
LDA	—	A9	A5	B5	AD	BD	B9	A1	B1	N Z
STA	—	—	85	95	8D	9D	99	81	91	
ADC	—	69	65	75	6D	7D	79	61	71	N Z C V
SBC	—	E9	E5	F5	ED	FD	F9	E1	F1	N Z C V
AND	—	29	25	35	2D	3D	39	21	31	N Z
EOR	—	49	45	55	4D	5D	59	41	51	N Z
ORA	—	09	05	15	0D	1D	19	01	11	N Z
CMP	—	C9	C5	D5	CD	DD	D9	C1	D1	N Z C
ASL	0A	—	06	16	0E	1E	—	—	—	N Z C
LSR	4A	—	46	56	4E	5E	—	—	—	N Z C
ROL	2A	—	26	36	2E	3E	—	—	—	N Z C
ROR	6A	—	66	76	6E	7E	—	—	—	N Z C
LDX	—	A2	A6	B6†	AE	—	BE	—	—	N Z
STX	—	—	86	96†	8E	—	—	—	—	
CPX	—	E0	E4	—	EC	—	—	—	—	N Z C
DEX	CA*	—	—	—	—	—	—	—	—	N Z
INX	E8*	—	—	—	—	—	—	—	—	N Z
LDY	—	A0	A4	B4	AC	BC	—	—	—	N Z
STY	—	—	84	94	8C	—	—	—	—	
CPY	—	C0	C4	—	CC	—	—	—	—	N Z C
DEY	88*	—	—	—	—	—	—	—	—	N Z
INY	C8*	—	—	—	—	—	—	—	—	N Z
DEC	—	—	C6	D6	CE	DE	—	—	—	N Z
INC	—	—	E6	F6	EE	FE	—	—	—	N Z
BIT	—	—	24	—	2C	—	—	—	—	7 Z 6

* Implied
† Zero Page, Y

N	negative result
V	overflow
Z	zero result
C	carry
6	set V if bit 6
7	set N if bit 7

Transfer

		FLAGS
TAX	AA	N Z
TAY	A8	N Z
TSX	BA	N Z
TXA	8A	N Z
TXS	9A	
TYA	98	N Z

Stack Operations

		FLAGS
PHA	48	
PHP	08	
PLA	68	N Z
PLP	28	Restored

Syntax Examples

Accumulator	ASL A
Immediate	LDA #nn
Zero Page	LDA nn
Zero Page, indexed by X	LDA nn, X
Absolute	LDA nnnn
Absolute, indexed by X	LDA nnnn, X
Absolute, indexed by Y	LDA nnnn, Y
Indirect, indexed by Y	LDA (nn, X)
Indirect, indexed by Y	LDA (nn), Y
Relative	BCC nn
Absolute indirect	JMP (nnnn)

nn — single-byte number
nnnn — double-byte number

Branch

BCC	90
BCS	B0
BEQ	F0
BMI	30
BNE	D0
BPL	10
BVC	50
BVS	70

Clear & Set

CLC	18
CLD	D8
CLI	58
CLV	B8
SEC	38
SED	F8
SEI	78

Program Flow

JMP	4C	Absolute
JMP	6C	Absolute Indirect
JSR	20	
RTS	60	
RTI	40	
BRK	00	
NOP	EA	

Least Significant Digit

Most Significant Digit

	0	1	2	4	5	6	8	9	A	C	D	E
0	BRK	ORA (nn, X)			ORA nn	ASL nn	PHP	ORA #nn	ASL A		ORA nnnn	ASL nnnn
1	BPL	ORA (nn), Y			ORA nn, X	ASL nn, X	CLC	ORA nnnn, Y			ORA nnnn, X	ASL nnnn, X
2	JSR	AND (nn, X)		BIT nn	AND nn	ROL nn	PLP	AND #nn	ROL A	BIT nnnn	AND nnnn	ROL nnnn
3	BMI	AND (nn), Y			AND nn, X	ROL nn, X	SEC	AND nnnn, Y			AND nnnn, X	ROL nnnn, X
4	RTI	EOR (nn, X)			EOR nn	LSR nn	PHA	EOR #nn	LSR A	JMP nnnn	EOR nnnn	LSR nnnn
5	BVC	EOR (nn), Y			EOR nn, X	LSR nn, X	CLI	EOR nnnn, Y			EOR nnnn, X	LSR nnnn, X
6	RTS	ADC (nn, X)			ADC nn	ROR nn	PLA	ADC #nn	ROR A	JMP (nnnn)	ADC nnnn	ROR nnnn
7	BVS	ADC (nn), Y			ADC nn, X	ROR nn, X	SEI	ADC nnnn, Y			ADC nnnn, X	ROR nnnn, X
8		STA (nn, X)		STY nn	STA nn	STX nn	DEY		TXA	STY nnnn	STA nnnn	STX nnnn
9	BCC	STA (nn), Y		STY nn, X	STA nn, X	STX nn, Y	TYA	STA nnnn, Y	TXS		STA nnnn, X	
A	LDY #nn	LDA (nn, X)	LDX #nn	LDY nn	LDA nn	LDX nn	TAY	LDA #nn	TAX	LDY nnnn	LDA nnnn	LDX nnnn
B	BCS	LDA (nn), Y		LDY nn, X	LDA nn, X	LDX nn, Y	CLV	LDA nnnn, Y	TSX	LDY nnnn, X	LDA nnnn, X	LDX nnnn, Y
C	CPY #nn	CMP (nn, X)		CPY nn	CMP nn	DEC nn	INY	CMP #nn	DEX	CPY nnnn	CMP nnnn	DEC nnnn
D	BNE	CMP (nn), Y			CMP nn, X	DEC nn, X	CLD	CMP nnnn, Y			CMP nnnn, X	DEC nnnn, X
E	CPX #nn	SBC (nn, X)		CPX nn	SBC nn	INC. nn	INX	SBC #nn	NOP	CPX nnnn	SBC nnnn	INC nnnn
F	BEQ	SBC (nn), Y			SBC nn, X	INC nn, X	SED	SBC nnnn, Y			SBC nnnn, X	INC nnnn, X

MICRO

PET Vet

By Loren Wright

FORTH for PET

Because of the international standards that exist for FORTH, there are few differences in the various FORTH implementations for the PET. The differences come in the packages of add-on commands, the accessory programs, the documentation, and the support you receive from the manufacturer after you buy the software.

I obtained one version of FORTH for this column — FORTH for CBM/PET by L.C. Cargile, Jr., and Michael Riley ($50 from AB Computers, 252 Bethlehem Pike, Colmar, PA 18915). This is a full fig-FORTH, with the FORTH-79 extensions available as a convenient add-on.

First, I recommend that you copy the disk and make a customized FORTH version that includes "PET-style" input functions. This replaces the primitive, teletype-style input with the convenient and familiar screen-editing of the PET. Why wasn't this included in the original version of this FORTH? It wouldn't comply to the standards! Each 2040/4040 disk holds 150 "screens," while each 8050 disk holds 480.

In addition to FORTH itself, the disk provides several screens, which include the editor, error messages, printer drivers, assembler extensions, string functions, and useful sample programs. One will print the calendar for any month of any year. Figure out how it works, and you have learned a lot about FORTH. The manual does a good job of documenting this implementation. There is a complete fig-FORTH glossary, a listing of the screen contents, and a memory map of the kernel words. Although there is a large section entitled "If You've Never Tried FORTH...," you are better off learning the language elsewhere. AB advertises a FORTH Metacompiler for $30, which I was unable to try. This program creates compressed object code, which can be executed directly (without any FORTH software).

Other FORTH versions, which I was not able to evaluate, are available from FSS ($50 - $70, 1903 Rio Grande, Austin, TX 78705) and from Microtech ($75, P.O. Box 102, Langhorne, PA 19047).

A First Look at the SuperPET

I recently spent several hours with a SuperPET, Commodore's newest computer. It consists of an 8032 with a 6809 processor board, 64K of additional memory, an RS-232 interface, and a great deal of software and documentation. The software consists of interpreted versions of four popular, high-level languages (APL, BASIC, FORTRAN, and Pascal) and a 6809 Assembly Language Development package. These programs were developed at the University of Waterloo, Canada, for use in an educational environment. Because the same language interpreters have been written for the IBM 370 and other mainframe computers, the software (and the 8032) can be used to develop programs for a mainframe computer.

The extra 64K of memory is used to hold the language interpreters, which run under the control of the 6809. The 32K in the 8032 is used for program storage. The 64K is divided into sixteen 4K blocks which are switched in and out of the $9000 block. This bank switching is accommodated automatically by the Waterloo software, but it is up to the user to do the switching (by manipulating a few bytes) when the 6502 is in control. Familiar 6502-based programs like Wordpro and VisiCalc can only use the original 32K, although I understand that new versions of the programs are on the way.

I concentrated on learning to use Waterloo microPascal, which was a real joy to use. Just type in the source program and run it, as you would a BASIC program. You get immediate feedback with comprehensive error messages, so it is easy to write and debug programs. The implementation of Pascal is the most complete of the ones I tried. There is also an interactive debugger, which allows you to trace program execution and examine variable values at any point in the program. The manual includes a series of sample programs (also included on the disk), which serve as a tutorial. While the examples cover many features of the language, they aren't comprehensive, by any means.

You can probably see how the interactive nature of the Waterloo interpreters would fit into the classroom. High schools and community colleges could teach these languages without owning or even having access to a mainframe computer. Organizations with the mainframe computer and the appropriate interpreters could have their programmers do a great deal of their development work, perhaps even at home, using the SuperPETs. Then, using the RS-232 port, the SuperPET-developed program can be fed to the mainframe computer, directly or over the phone lines. Programming could become another cottage industry!

Waterloo microCOBOL is under development, and should be available before too long.

Finally, don't confuse the SuperPET with the 8096 (as I did!), which also uses an additional 64K of RAM. This conversion does not include a 6809 or RS-232. None of the Waterloo interpreters will work in an 8096. However, the 8096 will be supported with new versions of word processors, data base managers, and programs like VisiCalc. As I mentioned last month, a UCSD Pascal will be available soon for the 8096.

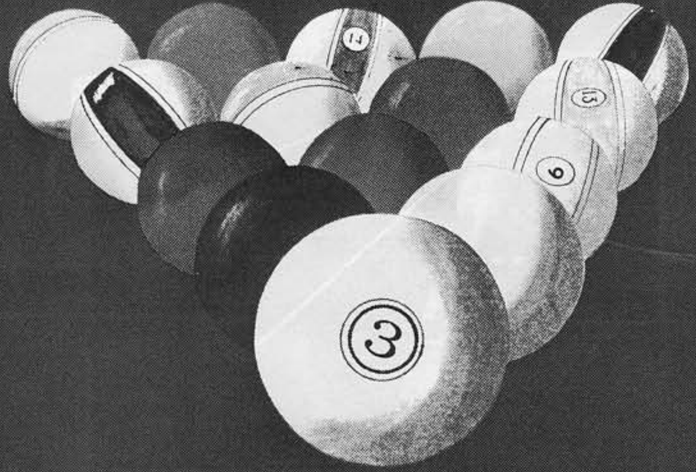
The 64K upgrade for an 8032 is now available for $500. The SuperPET upgrade for the 8032 ($995) is not yet available.

Overlooked?

I receive review copies of many more products than I can possibly cover in this column. I have put several of these to good use, which certainly attests to their value. These include the Programmer's Toolkit (Palo Alto IC's), MAE (assembler editor, Eastern House Software), Wordpro 3 (Professional Software), and VisiCalc (Personal Software). In addition, Commodore has lent us a great deal of equipment, including a 4032, 8032, 4022, 2040, 8050, VIC, and VIC 1515 printer. I thank all of these companies for their support.

MICRO

POOL 1.5™ ATARI™ OWNERS,



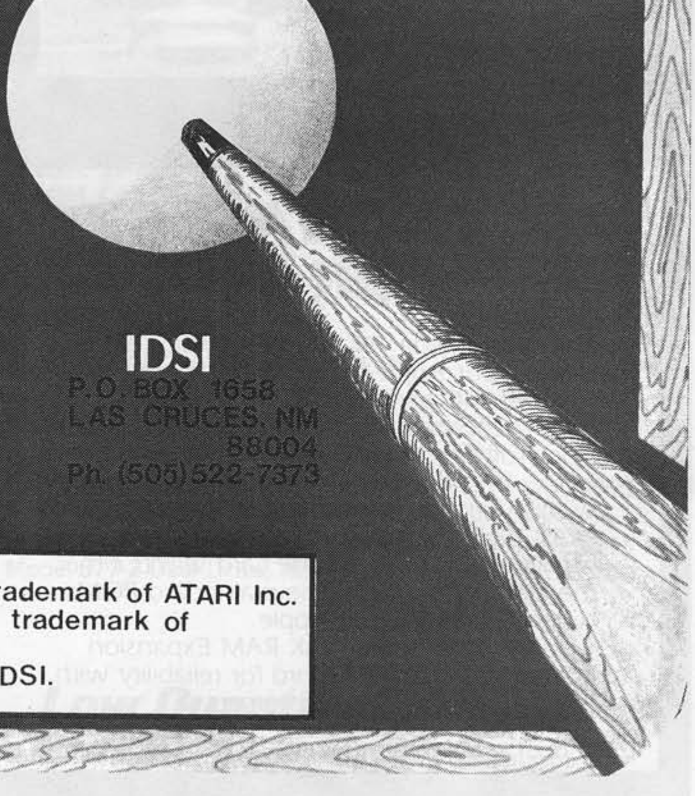
NOW YOU TOO CAN PLAY POOL 1.5 ON YOUR ATARI 48K DISK SYSTEM.

ALSO AVAILABLE FOR THE APPLE II™ 48K DISK SYSTEM.

PRICE: $34.95

- **High resolution graphics- Life like motion.**
- **Instant replay on any shot.**
- **Superfine aiming.**
- **Pool room sound effects**
- **Choice of 4 popular games**

See your local computer dealer or order directly from IDSI. Specify ATARI or APPLE.



IDSI
P.O. BOX 1658
LAS CRUCES, NM
88004
Ph. (505) 522-7373

ATARI is the registered trademark of ATARI Inc.
APPLE II is the registered trademark of APPLE Computer Inc.
POOL 1.5 is trademark of IDSI.

Credit Box Creator

After writing a long Applesoft program, it is good to append an information credit block at the end where it cannot easily be deleted. The routines here enable you to do that, and demonstrate some clever error-trapping techniques which may come in handy on your next programming task.

Sandy Mossberg
50 Talcott Road
Port Chester, New York 10573

Have you ever wished to place an information box at the end of your program? Better yet, have you wanted all the lines of that box to be numbered 65535 so that deletion of the box could not be accomplished with ease (Applesoft forbids numbers greater than 63999 to be typed from the keyboard)? This program allows you to do just that without entering the monitor.

An explanation of potential program usage is provided within the information section of the program. The production of a sample output enables corrections to be made before the box is appended to your loaded (target) program. End-of-program data allows for the deletion of the box if desired.

The main program, entitled CRBC (listing 1), is accessed through an EXEC file named CREDIT BOX CREATOR (listing 2). The latter file stores pointers for the beginning (decimal 103-104) and the end (175-16) of the loaded program in one of several scratchpad areas (decimal 1912-1915) of text page one. Beginning-of-program pointers are then reset to 256 bytes greater than the end of the loaded program. This is done so that when CRBC is EXECed, it will load in at an address that is 256 bytes beyond the end of the target program. Note that the byte immediately preceding the start of CRBC is set to zero.

In the main program, extensive error trapping is utilized. GET (rather than INPUT) statements are employed to acquire text material, with each line of text formed by concatenation and stored in the string, I$(*). This method enables input to contain "forbidden" characters such as commas and colons and provides simple text editing capabilities. REM statements in the DATA INPUT section of listing 1 document all traps and filters. In a 48K configuration, forced garbage collection is not necessary.

Once the sample output is accepted, the true end of your program is located and the box is appended by POKEing data into consecutive memory locations beginning at the second of the three zero-bytes that signal the end-of-program. REM statements in lines 1200-1660 of listing 1 document this sequence. Finally, target program markers are restored and the ending message is printed.

I hope you will find CREDIT BOX CREATOR enjoyable and the programming principles useful.

Listing 1

```
10  REM

              CRBC
         BY SANDY MOSSBERG

20  REM

30  TEXT : CALL  - 936: POKE  - 1
     6298,0
40  GOSUB 6030
50  GOSUB 5010
200  REM

         DATA INPUT

210  GOSUB 3010
220  VTAB 5: PRINT "USE NO MORE T
     HAN 17 CHARACTERS PER LINE"
230  PRINT "TYPE ONE OR MORE SPAC
     ES FOR A BLANK LINE";
240  PRINT "PRESS <CTRL-B> TO RET
     URN TO PRIOR LINE"
250  PRINT "PRESS <CTRL-C> TO ABO
     RT PROGRAM"
260  FOR I = 1 TO 40: PRINT "-";:
      NEXT I
270  POKE 34,10: REM  SET WINDOW
     TOP TO POSITION #11 (0 IS
     TOP POSITION) SO THAT
     INSTRUCTIONS AND HEADER
     WILL REMAIN ON CRT.
280  VTAB 11: INPUT "NUMBER OF TE
     XT LINES: ";LL$:LL =  VAL (L
     L$)
290  IF LL$ = "" THEN  VTAB 11: HTAB
     1: CALL  - 868: GOTO 280: REM
      TRAPS <RETURN>
300  FOR I = 1 TO  LEN (LL$)
310  IF  ASC ( MID$ (LL$,I,1)) <
     48 OR  ASC ( MID$ (LL$,I,1))
      > 57 THEN  GOSUB 340: VTAB
     11: CALL  - 958: GOTO 280: REM
     TRAPS NON-NUMERIC INPUT
320  NEXT I
330  GOTO 350
340  VTAB 14: HTAB 1: INVERSE : PRINT
     " PLEASE ENTER NUMERIC DATA
     ONLY ": NORMAL : FOR J = 1 TO
     3000: NEXT J: RETURN : REM
     ERROR MESSAGE
350  DIM I$(LL + 4),TM$(LL + 4): VTAB
     14
360  POKE 34,12
```

(Continued)

Listing 1 *(Continued)*

```
370  FOR I = 1 TO LL
380  PRINT "LINE #"I;
390  PRINT ": ——————————————";
     : IF I > 9 THEN  HTAB 9: CALL
      - 868: PRINT ":——————————
     —————";: REM  TEXT INPUT
     LINES
400 K = 0: REM K=LINE POSITION
410 K = K + 1
420  HTAB (9 + K): IF FL THEN FL =
     0: PRINT "-";: CALL  - 1008:
      REM FLAG SET BY PRESSING<--
      WHICH ERASES LAST LETTER
      ON LINE.
430  GET I$: PRINT I$;: REM  ALL
     INPUT WITH GET STATEMENTS.
440 I$(I) = I$(I) + I$: REM
     I$(I)=ONE LINE OF TEXT WHEN
     CONCATENATION COMPLETE.
450  IF K = 1 AND I$ =  CHR$ (13)
      THEN I$(I) = "": CALL  - 99
     8: HTAB 1: CALL  - 958: GOTO
     380: REM  TRAPS <RETURN>
     BEFORE ANY TEXT IS TYPED.
460  IF K = 1 AND I$ =  CHR$ (8) THEN
     I$(I) = "": HTAB 1: CALL  -
     958: GOTO 380: REM  TRAPS
     <-- BEFORE ANY TEXT IS
     TYPED.
470  IF I$ =  CHR$ (3) THEN 2130:
      REM END ON <CTRL-C>
480  IF I = 1 AND I$ =  CHR$ (2) THEN
      HTAB 1: CALL  - 868: GOTO 3
     80: REM        IF ON LINE #1
     DISABLE <CTRL-B>
490  IF I$ =  CHR$ (2) THEN I$(I)
      = "":I$(I - 1) = "":I = I -
     1: CALL  - 998: HTAB 1: CALL
      - 958: GOTO 380: REM GO TO
     PRIOR LINE ON <CTRL-B>
500  IF I$ =  CHR$ (13) THEN I$(I
     ) =  LEFT$ (I$(I), LEN (I$(I
     )) - 1): GOTO 560: REM   NEW
      LINE. REMOVE RETURN
     (<CTRL-M>) FROM STRING.
510  IF K = 2 AND I$ =  CHR$ (8) THEN
     I$(I) = "":K = K - 1:FL = 1:
      GOTO 420: REM ALLOWS FIRST
     LETTER OF LINE TO BE ERASED.

520  IF I$ =  CHR$ (8) THEN I$(I)
      =  LEFT$ (I$(I), LEN (I$(I)
     ) - 2):K = K - 1:FL = 1: GOTO
     420: REM <--
530  IF  LEN (I$(I)) > 15 THEN  PRINT
      CHR$ (7);: REM SOUND
     WARNING BELL AFTER 15
     LETTERS PRINTED.
540  IF  LEN (I$(I)) > 17 THEN  PRINT
     : PRINT : INVERSE : PRINT "M
     ORE THAN 17 CHARACTERS. PLEA
     SE REENTER!": NORMAL : FOR J
      = 1 TO 3000: NEXT :I$(I) =
     "": POKE 37, PEEK (37) - 4: CALL
      - 958: GOTO 380: REM TRAPS
     INPUT > 17 LETTERS.
550  GOTO 410
560  NEXT I
570  POKE 34,0: REM RESET WINDOW
     TOP.
700  REM
        SAMPLE OUTPUT

710  GOSUB 3110
720  VTAB 6
730  FOR I = 1 TO LL:TM$(I) = I$(
     I): NEXT I: REM  TM$(I)
     EQUATED TO I$(I) AND WILL BE
     USED FOR SORTING TO FIND THE
     LONGEST LINE INPUT. THIS
     HELPS CONSTRUCT BOX SIZE.

740  REM SORT LINE LENGTHS USING
     BUBBLE SORT
750  FOR J = 1 TO LL - 1
760  FOR I = 1 TO LL - J
770  IF  LEN (TM$(I)) <  LEN (TM$
     (I + 1)) THEN TM$ = TM$(I):T
     M$(I) = TM$(I + 1):TM$(I + 1
     ) = TM$
780  NEXT I
790  NEXT J
800  REM  TM$(1) NOW CONTAINS
     LONGEST LINE
810 LG =  LEN (TM$(1))
820 BB =  LEN (TM$(1)) + 14
830  PRINT "65535 REM ";: FOR I =
     1 TO LG + 4: PRINT "*";: NEXT
     I: PRINT
840  PRINT "65535 REM *";: HTAB (
     BB): PRINT "*"
850  FOR I = 1 TO LL
860  PRINT "65535 REM * "I$(I);: HTAB
     (BB): PRINT "*"
870  NEXT I
880  PRINT "65535 REM *";: HTAB (
     BB): PRINT "*"
890  PRINT "65535 REM ";: FOR I =
     1 TO LG + 4: PRINT "*";: NEXT
     I: PRINT
900  PRINT : PRINT
910  PRINT "IS THIS CORRECT? (Y/N
     ) ";: GET I$: IF I$ = "N" THEN
      HTAB 1: CALL  - 868: INVERSE
     : PRINT " LET'S START AGAIN
     ": NORMAL : FOR I = 1 TO 300
     0: NEXT : CLEAR : GOTO 50

920  IF I$ <  > "Y" THEN  HTAB 1:
      CALL  - 958: GOTO 910
930  PRINT : PRINT : HTAB 6: FLASH
     : PRINT " ONE MOMENT ": NORMAL

1000  REM

        CONFIGURE FINAL
        OUTPUT STRINGS

1010  FOR I = LL TO 1 STEP  - 1:I
     $(I + 2) = I$(I): NEXT I: REM
     INC SUB BY 2 TO PROVIDE ROOM
     FOR TOP & BOTTOM OF BOX
1020 I$(1) = "":I$(2) = ""
1030  FOR I = 1 TO LG + 2
1040 I$(1) = I$(1) + "*": REM
     LINE #1
1050 I$(2) = I$(2) + " ": REM
     LINE #2
1060  NEXT I
1070 I$(LL + 3) = I$(2): REM
     PENULTIMATE LINE=LINE #2
1080 I$(LL + 4) = I$(1): REM
     LAST LINE=LINE #1
1090  FOR I = 1 TO LL: REM
     EQUALIZE LINES
1100  FOR J = 1 TO LG + 2 -  LEN
     (I$(I + 2))
1110  IF J = 1 THEN I$(I + 2) = "
      " + I$(I + 2): GOTO 1130
1120 I$(I + 2) = I$(I + 2) + " "
1130  NEXT J: NEXT I
1200  REM
        FIND END OF PROGRAM

1210  REM PGM MARKERS HAVE BEEN
     EXECED INTO SCRATCHPAD
     STORAGE AREA OF TEXT PG 1
     (1912-1915)
```

Listing 1 *(Continued)*

```
1220  REM START IN 1912-1913
            END IN 1914-1915

1230  REM END OF PGM (EOP) MARKER
      MAY NOT GIVE TRUE EOP, THUS
      WE MUST CALC TRUE EOP BY
      FINDING 3 CONSECUTIVE ZERO
      BYTES, THE 2ND OF WHICH
      REPRESENTS THE BEGINNING
      ADDRESS OF THE BOX TO BE
      APPENDED

1240 M =  PEEK (1914) +  PEEK (19
     15) * 256 - 6: REM M=ADDRESS
     6 BYTES BEFORE EOP
1250  FOR I = M TO M + 16: IF  PEEK

     (I) = 0 AND  PEEK (I + 1) =
     0 AND  PEEK (I + 2) = 0 THEN
     TE = I + 1:I = M + 16: REM
     TE=ADDRESS OF BOX START
1260  NEXT I
1270 T1 = TE: REM SAVE TE
1500  REM

         POKE BOX INTO MEMORY

1510  FOR J = 1 TO LL + 4
1520 LS =  LEN (I$(J))
1530 NL = TE + 8 + LS
1540  POKE (TE),(NL -  INT (NL /
     256) * 256): POKE (TE + 1), INT
     (NL / 256): REM LO/HI BYTES
     OF NEXT LINE#
1550 T = T + 8 + LS: REM   TOTAL
     BYTES CREATED
1560 T2 = T: REM SAVE IT
1570  POKE (TE + 2),255: POKE (TE
      + 3),255: REM  LO/HI BYTES
     OF PRESENT LINE#
1580  POKE (TE + 4),178: POKE (TE
      + 5),42: REM  REM+*
1590  FOR I = 1 TO LS: POKE (TE +
     5 + I), ASC ( MID$ (I$(J),I,
     1)): NEXT I: REM      STRING
      ASCII VALUES
1600  POKE (TE + 6 + LS),42: REM
     *
1610  POKE (TE + 7 + LS),0: REM
        END OF LINE
1620 TE = NL
1630  NEXT J
1640  POKE (NL),0: POKE (NL + 1),
     0: REM EOP ZERO MARKERS
1650 T = T + 2: REM  TOTAL BYTES
     USED
1660 T2 = T: REM SAVE IT
2000  REM

         RESTORE MARKERS & END

2010  POKE 103, PEEK (1912): POKE
     104, PEEK (1913): POKE ( PEEK
     (103) +  PEEK (104) * 256 -
     1),0: REM  OLD PGM START
2020 T = T + T1: REM NEW PGM END
2030  POKE 175,(T -  INT (T / 256
     ) * 256): POKE 176, INT (T /
     256): REM  NEW PGM END
2040  GOSUB 3210
2050  VTAB 6
2060  PRINT " (1) STARTING AND EN
     DING PROGRAM": PRINT "      M
     ARKERS HAVE BEEN RESET.": PRINT

2070  PRINT " (2) "T2" BYTES GENE
     RATED THIS BOX.": PRINT
2080  PRINT " (3) SINCE LINE #655
     35 IS 'FORBIDDEN' BY";: PRINT
     "     APPLESOFT, YOU CANNOT
     DELETE IT IN": PRINT "      A
      CONVENTIONAL MANNER.  TO DE
     LETE": PRINT "     THE BOX,
     LOAD YOUR PROGRAM AND THEN";
      : PRINT "     POKE "T1",0:PO
      KE "T1 + 1",0."
2090  VTAB 18: HTAB 12: INVERSE :
      PRINT " END OF PROGRAM "
      : NORMAL

2100  POKE  - 16298,0: POKE  - 16
     300,0
2110  POKE  - 16368,0: TEXT
2120  CALL  - 998
2130  END
3000  REM
        HEADER1

3010  HOME
3020 TT$ = "==========": GOSUB 61
     00
3030 TT$ = "DATA INPUT": GOSUB 61
     00
3040 TT$ = "==========": GOSUB 61
     00
3050  RETURN
3100  REM

        HEADER2

3110  HOME
3120 TT$ = "=============": GOSUB
     6100
3130 TT$ = "SAMPLE OUTPUT": GOSUB
     6100
3140 TT$ = "=============": GOSUB
     6100
3150  RETURN
3200  REM

        HEADER3

3210  HOME : VTAB 2
3220 TT$ = "=================":GOSUB
     6100
3230 TT$ = "FOR YOUR INTEREST":GOSUB
     6100
3240 TT$ = "=================":GOSUB
     6100
3250  RETURN
3300  REM

        HEADER4

3310  HOME
3320 TT$ = "============": GOSUB
     6100
3330 TT$ = "INSTRUCTIONS": GOSUB
     6100
3340 TT$ = "============": GOSUB
     6100
3350  RETURN
5000  REM

        INSTRUCTIONS

5010  GOSUB 3310
5020  VTAB 6: PRINT "DO YOU WANT
     INSTRUCTIONS? (Y/N) ";: GET
     I$: IF I$ = "N" THEN  RETURN

5030  IF I$ < > "Y" THEN  VTAB 6
     : HTAB 1: CALL  - 868: GOTO
     5020
5040  POKE 34,4: HOME
5050  PRINT " 1.IN APPLESOFT NO L
     INE NUMBER GREATER": PRINT "
        THAN 63999 CAN BE TYPED I
     N FROM THE": PRINT "   KEYBO
     ARD.  THIS UTILITY WILL ENAB
     LE": PRINT "   YOU TO CREATE
      A BOX CONSISTING OF": PRINT
     "   MULTIPLE REM STATEMENTS
     WITH LINE"
5060  PRINT "   NUMBER 65535, THE
      HIGHEST THAT CAN BE";: PRINT
     "   PRODUCED WITH TWO BYTES
     ($FFFF).": PRINT
5070  PRINT " 2.THE CONTENTS OF T
     HE BOX MIGHT CONTAIN";: PRINT
     "   COPYRIGHT MATERIAL, EXPL
     ANATORY DATA,";: PRINT "   O
     R EVEN A TABLE OF VARIABLES.
        ITS": PRINT "   USE WILL B
     E LIMITED ONLY BY YOUR": PRINT
     "   INGENUITY!": PRINT
5080  PRINT " 3.TO USE THIS UTILI
     TY FIRST LOAD YOUR": PRINT "
        PROGRAM AND THEN 'EXEC CR
     EDIT BOX": PRINT "   CREATOR
     '.": PRINT
5090  GOSUB 6120: HOME
5100  PRINT " 4.PLAN THE NUMBER O
     F LINES TO USE FOR": PRINT "
        PRINTING TEXT (17 CHARACT
     ERS PER LINE";: PRINT "   IS
      MAXIMAL).  DO NOT CONCERN Y
     OURSELF";: PRINT "   WITH TH
     E ACTUAL BOX, SINCE IT WILL
     BE";: PRINT "   CONSTRUCTED
     BY THE PROGRAM.  SIMPLE"
5110  PRINT "   EDITING FEATURES
     ARE AVAILABLE, AND": PRINT "
        PRESSING A WRONG KEY LIKE
     LY WILL NOT": PRINT "   CAUS
     E THE PROGRAM TO CRASH BECAU
     SE OF";: PRINT "   EXTENSIVE
      ERROR TRAPPING.": PRINT
5120  PRINT " 5.A SAMPLE BOX IS P
     RODUCED.  YOU SHOULD";: PRINT
     "   CHECK IT FOR ACCURACY.
     IF THE OUTPUT";: PRINT "   I
     S ACCEPTED, THE BOX WILL BE
     APPENDED";: PRINT "   TO THE
      PROGRAM IN RAM MEMORY.": PRINT

5130  PRINT " 6.BE SURE TO OBSERV
     E THE DATA PRESENTED";: PRINT
     "   AS THE PROGRAM ENDS.  HA
     VE FUN!!!"
5140  GOSUB 6120
5150  POKE 34,0: RETURN
6000  REM

        TITLE PAGE

6010  REM  TITLE PAGE FROM
6020  REM  SF APPLE CORE

6030  POKE  - 16368,0
6040  VTAB 6:TT$ = "CREDIT BOX CR
     EATOR"
6050  GOSUB 6090: VTAB 9:TT$ = "B
     Y SANDY MOSSBERG"
6060  GOSUB 6090
6070  VTAB 15: PRINT "THIS UTILIT
     Y APPENDS TO THE END OF YOUR
     ": PRINT "PROGRAM A BOX THAT
      MAY CONTAIN CREDITS,": PRINT
     "COPYRIGHT DATA, EXPLANATORY
      MATERIAL OR": PRINT "A VARI
     ABLE TABLE.  BE IMAGINATIVE!
     "
6080  GOSUB 6120: RETURN
6090  REM

         PRINT CENTER

6100 WIDTH = 20 - ( LEN (TT$) / 2
     ): IF WIDTH < = 0 THEN  PRINT
     TT$: RETURN
6110  HTAB WIDTH: PRINT TT$: RETURN
```

(Continued on page 63)

GET FREE SOFTWARE FOR YOUR APPLE !

HOW? JUST ORDER ONY OF THE ITEMS BELOW, AND SELECT YOUR FREE SOFTWARE FROM THE BONUS SOFTWARE SECTION, USING THE FOLLOWING RULE: FOR THE FIRST $100.00 WORTH OF MERCHANDISE ORDERED TAKE 1 ITEM; FOR THE NEXT $200.00 WORTH OF MERCHANDISE ORDERED TAKE ANOTHER ITEM; FOR THE NEXT $300.00 TAKE A THIRD ITEM, ETC. ALL AT NO COST.

HARDWARE BY APPLE

APPLE II PLUS, 48k	1199
DISK DRIVE+CONTROLLER (3.3)	535
DISK DRIVE only	455
Language System w. Pascal	397
Silentype Printer & Interface	360
Integer or Applesoft Firmware Card	159
Graphics Tablet	645
Parallel Printer Interface Card	149
Hi-Speed Serial Card	155
Centronics Parallel Intfce.	175

HARDWARE BY OTHERS

HAYES MICROMODEM II	300
HAYES SMART MODEM	239
HAYES S100 MODEM	339
VIDEX VIDEOTERM 80 W. GRAPHICS	275
MICROSOFT Z80 SOFTCARD	299
MICROSOFT 16K RAMCARD	159
CORVUS 10MB HARD DISK	4750
SSM A10 SERIAL/PARALLEL A&T	189
MICRO-SCI Disk & Controller	495
TYMAC DOUBLE DOS 3.2/3.3	35

VIDEO MONITORS

Leedex-Video-100 12" B&W w/Cable	139
Leedex 12" Green w/Cable	165
Leedex 13" COLOR MONITOR & Cable	399
SUP-R-TERM RF MODULATOR	29

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
CPS Multifunction Bd.	239

SOFTWARE FOR APPLE

APPLE FORTRAN	159
APPLE PILOT	125
DOS 3.3	50
DOS TOOL KIT	65
APPLE PLOT	59
D. J. REPORTER	45
D. J. NEWS	45
PORTFOLIO	45
SHELL GAMES	25
ELEMENTARY DEAR APPLE	25

SOFTWARE BY OTHERS

APPLE FORTRAN by MICROSOFT	159
APPLE BASIC COMPILER by MICROSOFT	315
APPLE COBOL by MICROSOFT	599
VISICALC 3.3	169
VISIPLOT	155
VISIPLOT/VISITREND	199
VISIDEX	169
CCA DATA MGT.	79
DB MASTER by STONEWARE	189
DATACAPTURE 4.0	55
Z – TERM	65
ON-LINE APPLESOFT COMPILER	89

SOFTWARE BY PEACHTREE

GEN. LEDGER	219
A/R	219
A/P	219
PAYROLL	219
INVENTORY	219
MAIL LIST	219

WORD PROCESSING SOFTWARE FOR APPLE

PEN-ULTIMATE	235
WORD STAR	245
EZ WRITER Prof. Sys.	195
EZ WRITER	89
MUSE SUPER TEXT II	139
APPLE-WRITER	69
PROGRAMMA APPLE PIE 2.0	110
MAGIC WAND	345
WORDPOWER	50

EPSON PRINTERS

MX - 70 w/Graftrax	415
MX - 80	515
MX - 80 FT	615
MX - 80 w. GRAFTRAX	575
MX - 80 FT w. GRAFTRAX	675
MX - 100 FT w. GRAFTRAX	775
APPLE PAR. INTFCE (for all Epson)	75
MX -70/80 FRICTION FEED Adaptor	75

OTHER PRINTERS

IDS 445 w. GRAPHICS + 2K Buff.	750
IDS 460 w. GRAPHICS	899
IDS 460	825
IDS 560	1099
IDS 560 w. GRAPHICS	1150
CENTRONICS 737	699
CENTRONICS 739	799
CCS Centronics Par. Intfce & Cable	135
NEC SPINWRITER 5510 RO	2795
C. ITOH 25 CPS DAISYWHEEL	1750
C. ITOH 45 CPS DAISY WHEEL	2025
WATANABE MI-PLOT PLOTTER	1150
DIABLO 630 DAISY w. P. Wheel & Rib.	2350

BONUS SOFTWARE SECTION!

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 window, 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 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 48K & ROM APPLESOFT.......... $35

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. User's 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 of files can be printed, if desired. Additional modules coming are a STATISTICS INTERFACE, CHECKBOOK, MAILING LIST & DATA-ENTRY.
REQUIRES 48K & ROM APPLESOFT............ $40

APPLE LITERATURE DATABASE: allows rapid retrieval (via keywords) of references from total APPLE literature thru 1980, on 5.25" disk. Each entry in the data base consists of the article, author-name, periodical-name, date of issue, & page nos. The database is intended to support large magazine files which would require lengthy manual searching to recover information. Annual updates will be available.
REQUIRES 48K, ROM APPLESOFT............ $60

WORDPOWER: is a simple, powerful, low cost, line-oriented word-processor program. It offers a fast machine language FIND & REPLACE. Text can be listed to screen or printer, with or without line-numbers. Lower-case adaptors are supported. You can merge files, move groups of lines, and easily add, change, or delete lines. WORDPOWER can be used to create and maintain EXEC files. It can also be used as a rapid, unstructured, information-storage and retrieval system via its rapid search capabilities.
REQUIRES: 48K, ROM APPLESOFT............ $50

LABELMAKER: allows users to quickly create address labels. A given label may be generated in any quantity from 1 to 32767. Space is allowed on labels for a personal and company name, but the space is automatically closed up if only a personal name is entered. Space is also allowed for foreign countries. The program can also generate labels for price-tags, part numbers and mail-messages such as "RUSH", "FRAGILE", etc. A self-incrementing feature allows theatre-tickets to be produced, with a date, and numbers running from a000 to z999. An editor is provided for editing labels prior to printing. All labels may be saved to disk for instant recall.
REQUIRES 48K & ROM APPLESOFT............ $35

TO ORDER: Use phone or mail. We accept VISA, MASTERCARD, COD's, personal checks & money orders. Add 4% for credit card. Customer pays handling on COD orders, Foreign orders must be in American Dollars & include 10% for handling. Connecticut residents add 7.5% sales tax.

BANKAMERICARD VISA

master charge

CONNECTICUT INFORMATION SYSTEMS CO.
218 Huntington Road, Bridgeport, CT 06608 (203) 579-0472

Listing 1 *(Continued)*

```
6120  VTAB 23: HTAB 12: PRINT "[E
    SC] TO END"
6130  VTAB 24: PRINT  TAB( 8);"[S
    PACE] TO CONTINUE ";
6140  PRINT "[ ]";: HTAB 29: GET
    ZZ$: IF ZZ$ =  CHR$ (27) OR


    ZZ$ =  CHR$ (3) THEN  TEXT :
     HOME : END
6150  IF ZZ$ =  CHR$ (32) THEN
     RETURN

6160  CALL  - 868: CALL  - 1008: GOTO


     6140: REM

65535  REM *********************
65535  REM *                   *
65535  REM *    CREDIT BOX     *
65535  REM *     CREATOR       *
65535  REM *                   *
65535  REM * S. MOSSBERG, M.D. *
65535  REM * 50 TALCOTT RD.    *
65535  REM * PORT CHESTER      *
65535  REM * NEW YORK, 10573   *
65535  REM *                   *
65535  REM * (914) 937-6400    *
65535  REM *                   *
65535  REM *********************
```

Listing 2

```
10  REM CREDIT BOX CREATOR
       EXEC FILE CREATE

20 D$ =  CHR$ (4):F$ = "CREDIT BO
    X CREATOR"

30  PRINT D$"OPEN"F$

40  PRINT D$"WRITE"F$

50  PRINT "POKE 1912,PEEK (103):P
    OKE 1913,PEEK (104):POKE 191
    4,PEEK (175):POKE 1915,PEEK
    (176)"

60  PRINT "POKE 103,PEEK (175):PO
    KE 104,PEEK (176) + 1:POKE (
     PEEK (103) +  PEEK (104) *
    256 - 1),0"

70  PRINT "RUN CRBC"

80  PRINT D$"CLOSE"
```

AT LAST...
...FOR
OSI

A **menu** driven
data base management
system for
multi-data base
applications

Data base
management system
$200.00

plus

Accounting
package
$150.00

Stock portfolio
$150.00

Stock financial
statement analysis
$250.00

for C8P/C3 systems
under OS65U

Genesis Information Systems, Inc.
P.O. Box 3001 • Duluth, MN • 55803
Phone 218-724-3944

ED-SCI STATISTICS

FOR THE PROFESSIONAL A COMPLETE STATISTICS AND DATA MANAGEMENT PACKAGE

Data Entry and Filing
- By Variable Name and Case Number
- One-Time Data Entry
- Easy and Rapid Editing
- Data Entry Worksheets

Data File Manipulation
- Add New Variables
- Add or Delete Case Values
- Create SUBFILES By User Defined SEARCH & SELECT Criteria
- Merge Files

Statistical Calculations
- Mean, Std. Dev., Std. Error
- Coefficient of Variation
- Frequency Distribution
- Unpaired t-Test
- Paired t-Test
- Mann-Whitney U Test
- Wilcoxon Paired Sample Test
- Chi-Square Test
- Linear Regression
- Correlation
- One-Way ANOVA with the Newman-Keuls Test
- Hard Copy of Data & Results

Statistical Calculations can be made on **VISICALC* (DIF)** and **DATADEX* FILES.** Graphic Plotting of all ***ED-SCI STATISTICS*** Data Files can be done with **APPLE PLOT.***

Only $95.00 brings you the ***ED-SCI STATISTICS*** instruction manual, the Master Program Disk, and a Back-Up Disk.

See ***ED-SCI STATISTICS*** at your local Apple Computer store. Dealer inquiries invited. For information please phone or write:

Ed-Sci Development
460 Beacon St. San Francisco, CA 94131 (415) 282-7020

ED-SCI STATISTICS requires an Apple II with the Applesoft or Language Card, or an Apple II+, 48K memory, and at least one disk drive with DOS 3.3 (16 sector).

*Apple is a registered trademark of Apple Computer Inc.
VisCalc is a registered trademark of Personal Software Inc.
DATADEX is a registered trademark of Sonoma Softworks.

GOSUB'S MX80/70 FRICTION FEED KIT
Only $49.95 ppd

- Use Single Sheet Paper
- Use Inexpensive Roll Paper
- Use Your Own Letterhead
- Does Not Affect Pin-Feed Use

Printers

Epson MX80.....$515 ppd   Epson MX100.....$825 ppd
Microtek's BYTEWRITER-1 .... $299 ppd

The Printer Stand **$29.95**

SAVE MONEY AND SPACE: Holds almost any printer that uses 9½"-size paper or forms. Made from ¼" acrylic, allows 3" of paper or forms to be stacked under printer. Great for use with bottom load printers! Fits MX80/70, Microline 80's, BASE 2-850, Bytewriter-1 and many more!

Software for the VIC 20 and PET

Math Huddler and Monster Maze	$14.95
Household Finance Parts 1 and 2	$34.95
Seawolf, Bounce Out and VIC Trap	$24.95
VIC 20 Biorhythm Compatibility	$ 9.95
VIC Lemonade	$14.95
AMOK	$18.95
ALIEN BLITZ	$18.95
VIC Music and Joypainting	$ 9.95

PET	Apple	Atari	TRS-80*
Nukewar	*Tanktics*		*B1-Bomber*
	Midway Campaign		*Planet Miners*

All just $14.95 each!

(Kansas residents add 3% sales tax)

GOSUB
P.O. Box 275
Wichita, KS 67201
(316) 265-9992

GOSUB Int'l, Inc.
501 E. Pawnee, Suite 430
Wichita, KS 67211
(316) 265-9858

**(TRS-80 is a trademark of Tandy Corp.)*

NOW TWO LOCATIONS

SAVE TIME • SAVE SHIPPING

Computers for people.™

800™ $699

410 Recorder	$59.00
810 Disc Drive	$444.00
822 Printer	$359.00
825 Printer	$629.00
830 Modem	$159.00
820 Printer	$269.00
850 Interface	$159.00
New DOS 2 System	$21.00
CX30 Paddle	$18.00
CX40 Joy Stick	$18.00
CX853 16K RAM	$89.00
Microtek 16K RAM	$75.00
Microtek 32K RAM	$159.00
One year extended warranty	**$50.00**

ATARI 400

16K.... $329
32K.... $478
48K.... $555

ATARI SOFTWARE

CX404 Word Processor	$119.00
CX404 PILOT	$68.00
CX413 Microsoft Basic	$68.00
CX4101 Invitation To Programing I	$17.00
CX4102 Kingdom	$13.00
CX4103 Statistics	$17.00
CX4104 Mialing List	$17.00
CX4105 Blackjack	$13.00
CX4106 Invitation to Programing 2	$20.00
CX4107 Biorythm	$13.00
CX4108 Hangman	$13.00
CX4109 Graph It	$17.00
CX4110 Touch Typing	$20.00
CX4008 SPACE INVADERS	$32.00
CX4112 States & Capitals	$13.00
CX4114 European Countries & Capitals	$13.00
CX4115 Mortgage & Loan Analysis	$13.00
CX4116 Personal Fitness Program	$59.00
CX4117 Invitation To Programing 3	$20.00
CX4118-20 Conversational Languages (ea.)	$45.00
CX4121 Energy Czar	$13.00
CXL4001 Educational Master	$21.00
CX6001-17 Talk & Teach Series (ea.)	$23.00
CX8106 Bond Analysis	$20.00
CX8107 Stock Analysis	$20.00
CX8101 Stock Charting	$20.00
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-Toe	$24.00
CLS4011 STAR RAIDERS	$39.00
CXL4012 MISSLE COMMAND	$32.00
CXL4013 ASTEROIDS	$32.00
CXL4015 TeleLink	$20.00
Visicalc	$149.00
Letter Perfect (Word Processor)	$109.00
Source	$89.00
CX481	$75.00
CX482	$109.00
CX483	$54.00
CX484	$319.00

PRINTERS

Centronics 739-1	$649.00
Diablo 630 Special	$1799.00
Epson	
MX70	$359.00
MX80	$469.00
MX80FT	Call
MX100	Call
NEC	
8023	$639.00
7730	Call
7720	Call
7710	Call
Okidata	
82A	$499.00
83A	$769.00
84	$1129.00
Citoh Starwriter	
25 CPS-P	$1329.00
45 CPS-P	$1699.00
Paper Tiger	
445G	$699.00
460G	$899.00
560G	$1129.00
Talley	
8024-7	$1399.00
8024.L	$1629.00
Xerox 820	
System I 5½"	$2450.00
System II 8"	$2950.00
CPM 5¼	$169.00
Word Processing	$429.00
Super Calc	$269.00

computer mail order saves

RAM SALE!

Atari	
Microtek 16K	$64.00
32K	$129.00
Intec 32K	$139.00
48K	$219.00
Ramcrom 128K	$519.00
Apple	
Microtek 16K	$94.00
32K	$129.00
Commodore	
64K upgrade	$389.00
Hewlett Packard	
16K upgrade	$249

TI-99/4A $379

32K Expansion	$329.00
PHP Printer Solid State	$319.00
PHC 004 TI-99/4 Home Computer	$399.00
PHP 1600 Telephone Coupler	$169.00
PHP 1700 RS-232 Accessories Interface	$169.00
PHP 1800 Disk Drive Controller	$239.00
PHP 1850 Disk Memory Drive	$389.00
PHP 2200 Memory Expansion (32K RAM)	$319.00
PHA 2100 R.F. Modulator	$43.00
PHP 1100 Wired Remote Controllers(Pair)	$31.00

Texas Instruments

PHM 3006 Home Financial Decisions	$26.00
PHM 3013 Personal Record Keeping	$43.00
PHD 5001 Mailing List	$60.00
PHD 5021 Checkbook Manager	$18.00
PHM 3008 Video Chess	$60.00
PHM 3010 Physical Fitness	$26.00
PHM 3009 Football	$26.00
PHM 3018 Video Games I	$26.00
PHM 3024 Indoor Soccer	$26.00
PHM 3025 Mind Challengers	$22.00
PHM 3031 The Attack	$35.00
PHM 3032 Blasto	$22.00
PHM 3033 Blackjack and Poker	$22.00
PHM 3034 Hustle	$22.00
PHM,3036 Zero Zap	$18.00
PHM 3037 Hangman	$18.00
PHM 3038 Connect Four	$18.00
PHM 3039 Yahtzee	$22.00
PHM 3017 Terminal Emulator I	$39.00
PHM 3026 Extended Basic	$88.00
PHM 3035 Terminal Emulator II	$45.00

HOW TO ORDER: Phone orders invited or send check or money order and receive free shipping in the continental United States. PA residents add 6% sales tax.

computer mail order west

800-648-3351

IN NEVADA, CALL (702) 588-5654
P.O. BOX 6689, STATE LINE, NEVADA 89449

TO SAVE YOU MORE!

COSTS • SAVE SALES TAX

you more money & time!

HEWLETT PACKARD

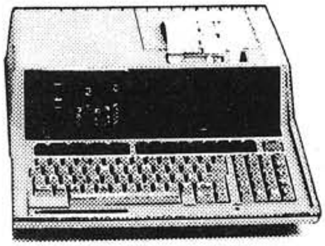
NEW! HP•125	$2999.00
HP•83	$1699.00
HP•85 16K Memory Module	$249.00
5¼" Dual Master Disc Drive	$2129.00
Graphics Plotter (7225B)	$2079.00

NOW IN STOCK!

The new HP41CV Calculator

$259

Call for HP Software Prices & Information.
Call for Calculator prices.



HP•85 $2479

CBM 8032

$1069

commodore

8032	$1069.00
4032	$969.00
4016	$769.00
8096	$1569.00
Super Pet	$1599.00
2031	$529.00
8050	$1299.00
4022	$599.00
C2N	$63.00
Word Pro 4 Plus	$299.00
Word Pro 3 Plus	$199.00

SOFTWARE

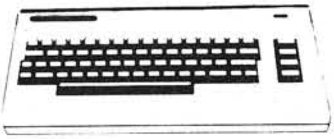
WordPro3 Plus	$299.00
WordPro4 Plus	$329.00
Commodore Tax Package	$399.00
Visicalc	$149.00
BPI General Ledger	$329.00
OZZ Information System	$289.00
Dow Jones Portfolio	$129.00
Pascal	$239.00
Legal Time Accounting	$449.00
Word Craft 80	$289.00
Power	$89.00
Socket-2-Me	$20.00
Jinsam	$Call
MAGIC	$ Call
The Manager	$209.00
Softrom	$129.00

Terminals

Televideo	
910	$579.00
912C	$699.00
920C	$749.00
950	$939.00
Call for computers	
Zenith 719	$749.00
Adds	$549.00

Monitors

Amdex 12" B&W	$149.00
12" Green	$169.00
13" Color	$349.00
Sanyo 12" B&W	$259.00
12" Green	$269.00
13" Color	$449.00
TI 10" Color	$349.00
Modems	
Novation Auto	$239.00
D Cat	$169.00
Cat	$159.00
HAyes	
Smart	$239.00
Pioneer Lazer Disk	$599.00
BSR X-10 Systems	
PK 500	$84.00
LM 501	$16.00
AM611	$17.00
AM286	$17.00



VIC 20 $259 COMPLETE

Vic-TV Modual	$19.00
Vic Cassette	$69.00
Vic 6 Pack Program	$44.00
VIC1530 Commodore Datassette	$69.00
VIC1540 Disk Drive	$499.00
VIC1515 VIC Graphic Printer	$399.00
VIC1210 3K Memory Expander	$32.00
VIC1110 8K Memory Expander	$53.00
VIC1011 RS232C Terminal Interface	$43.00
VIC1112 VIC IEEE-488 Interface	$86.00
VIC1211 VIC 20 Super Expander	$53.00
VIC1212 Programmers Aid Cartridge	$45.00
VIC1213 VICMON Machine Language Monitor	$45.00
VIC1901 VIC AVENGERS	$23.00
VIC1904 SUPERSLOT	$23.00
VIC1906 SUPER ALIEN	$19.00
VIC1907 SUPER LANDER	$23.00
VIC1908 DRAW POKER	$23.00
VIC1909 MIDNIGHT DRIVE	$23.00
VT106A Recreation Pack A	$44.00
VT107A Home Calculation Pack A	$44.00
VT164 Programmable Character/Gramegraphics	$12.00
VT232 VICTerm I Terminal Emulator	$9.00

Add 3% for VISA or MC. Equipment subject to price change and availability without notice.

computer mail order east

800-233-8950

IN PA. CALL (717) 327-9575
501 E. THIRD ST., WILLIAMSPORT, PA 17701

Lazer isn't afraid to compare!

(because we have the best lower case system available.)

Despite the fact that we were one of the first manufacturers to produce lower case equipment for the Apple II, Lazer MicroSystems products are still the state-of-the-art. Beside the obvious price/performance advantage we have over the competition, our products are expandable. Lazer is constantly introducting new products including our Lower Case + Plus II, Character Set + Plus (that adds 2 additional character sets to the Lower Case + Plus), and our new "Double Vision + Plus" for owners of Computer STOP's Double Vision 80-column board.

Lazer's products are compatable with more word processors than anybody else's. Our Lower Case + Plus is compatable with Easywriter and unmodified Apple writers. None of our competitors below can make that claim. In fact, BASIS' board isn't even compatable with Programma's PIE! The following chart lists Lazer's superiority over the competing units.

(LC+ = Lower Case + Plus, LC+II = Lower Case + Plus II, KB+ = Keyboard + Plus)

Feature	Paymar LCA-1	LCA-2	VIDEX	BASIS	VISTA	LC+	LC+II	KB+/ LC+II	KB+/ LC+	KB+
True ASCII upper/lower case display	Y	Y	Y	Y	N	Y	Y	Y	Y	N
Inverse Lower Case	N	N	rev 7 only	N	—	Y	N	N	Y	—
Font Size	5 x 7	5 x 7	5 x 8	5 x 8	—	5x7, 7x8	5 x 7	5 x 7	5x7, 7x8	—
# of on-board character sets	1	1	1	1	—	up to 4 (2 std)	1	1	up to 4	—
Pseudo-descenders	Y	Y	N	N	—	Y	Y	Y	Y	—
True descenders	N	N	Y	Y	—	optional	N	N	optional	—
Optional fonts avail. (ROM, disk)	N	N	N	Y	—	Y	N	N	Y	—
2716-compatible character generator compatable with fonts created by HIRES character generators	N	N	N	N	—	Y	N	N	Y	—
On-board graphics character set	N	N	N	N	—	Y	N	N	Y	—
Software provided on diskette	$5 extra		N	N	—	Y	Y	Y	Y	Y
Single board works with all Apples	N	N	N	N	Y	Y	N	N	Y	Y
Expandable System	N	N	N	N	N	Y	Y	Y	Y	Y
Extensive user Documentation	N	N	Y	N	N	Y	Y	Y	Y	Y
High quality PC board	N	—	Y	Y	Y	Y	—	Y	Y	Y
Reset key disable	N	N	Y	Y	N	N	N	Y	Y	Y
Shift key mod	N	N	Y	Y	N	N	N	Y	Y	Y
All 128 characters available from keyboard	—	—	N	N	—	—	—	Y	Y	Y
Type ahead buffer	N	N	N	N	Y	N	N	Y	Y	Y
# of characters in buffer	—	—	—	—	40	—	—	64	64	64
Ability to clear or turn off buffer	—	—	—	—	N	—	—	Y	Y	Y.
PRICE	59.95	49.95	129.95	125.00	49.95	64.95	29.95	129.90	164.90	99.95

Don't settle for anything less than the Lower Case + Plus, Lower Case + Plus II, or Keyboard + Plus. If your dealer doesn't stock our products give us a call, we'll try to connect you with a dealer in your area.

Before you buy a lower case system for the Apple II, call or write for our free booklet "Keyboard Enhances/Buffers and Lower Case Adapters: From The Inside Out." This booklet explains the advantages and disadvantages of using a lower case adapter in your Apple II.

Lazer products are carried by reputable dealers all across the world. If your dealer cannot provide you with a demonstration of our equipment and tries to sell you an inferior lower case adapter, give us a call, we'll give you the location of a dealer that can show you our equipment. Remember, an inteligent purchase cannot be made if you do not compare before buying.

Lazer's Products are recommended for use with DB Master by Stoneware.

State of the art performance, software compatability, and exceptional value make Lazer's products the best there are!

Lower Case + Plus	$64.95
Keyboard + Plus	$99.95
Lower Case + Plus II	$29.95
Character Set + Plus	$24.95
Double Vision + Plus	$39.95
Anix 1.0	$49.95
Lazer Pascal	$39.95

Visicalc Users! Now you can have lower case on your Visicalc Screen, *only* from Lazer.

Lazer
MICRO SYSTEMS INC.
1791-G Capital
Corona, CA 91720
(714) 735-1041

Visicalc - TM of Personal Software
DB Master - TM of Stoneware
Apple II - TM of Apple Computer, Inc.

RUNZMENU

This routine removes the nagging problem of rebooting the Apple DOS system master disk upon a language-card system reset. RUNZMENU re-enters DOS using the reset vector, and keeps both Applesoft and Integer BASIC active. Further, it allows the user the option of running a menu program automatically. With RUNZMENU, it is no longer necessary to make all disks system masters in order to support both BASICs and turnkey operation.

Frank Shyjka
112 North Ardmore
Columbus, Ohio 43209

The frequent loss of the BASIC on the language card was causing a problem. My children were unhappy with our Apple. Oh, they loved the games and did the educational lessons, but the addition of the language card caused them to enjoy and use the computer less. Instead of appreciating the increased power of the language system, they avoided the inconvenience of the monster called system master.

Previously, they had jumped from game to game with a simple PR#6. No more! The other, often necessary, language disappeared. With the language card they had to reboot the monster and then catalog, etc. By making each diskette a system master, we wasted too much space and lost the advantage of 16 sectors. Computers should do things quickly, quietly, and simply.

I decided to develop a program to tell Apple that the other language was still in the card. I decided on a machine language routine which would let me choose between simply mimicking the pre-language card reset, or running my Hello/menu program. The routine permits selection of a program with a single keystroke.

After much thought I decided to use the user reset vector at 03F2.03F3. My requirements for ease dictated that the program should always be resident. Obviously, it had to be loaded with DOS and survive without harm to itself or the operating system. I wanted to leave all DOS commands fully functional. Don Worth and Pieter Lechner's book, *Beneath Apple DOS*, was quite valuable. It listed two safe areas, the locations $BA69.BA95, with $2C bytes, and $BCDF.BCFF, with $20. The final program used all the available bytes.

The program is not complicated and has only one poor programming technique: it contains self-modifying code. Self-modifying code is undesirable because any interruption of the program could cause unexpected results upon re-entry. That problem concerns $BC6D. To prevent any error, that byte is reset with each entry, in the initialization section of code (at $BCDF-BCFD). It is reloaded with the correct starting location, $8C (at $BCFG).

The program is designed to do four things. First, it re-enters DOS *via* $03D0 by using the reset key and user reset vector, and keeps both languages active. Then, it automatically inputs the command, (RUNZMENU?). (ZMENU? is my menu program.) Next, after a keyboard input of a single letter, it either runs the menu program or re-enters DOS softly. And finally, it restores the Apple to its normal configuration to accept keyboard input. A disassembled listing with annotations is given below and step-by-step instructions for making your own RESET MASTER are listed at the end of the article.

The routine begins by saving the current input vectors.

```
BCDF-   AD 55 AA   LDA $AA55
BCE2-   8D FE BC   STA $BCFE
BCE5-   AD 56 AA   LDA $AA56
BCE8-   8D FF BC   STA $BCFF
```

Then we load a new input vector into KSWL,H at 0038.0039.

```
BCEB-   A9 69      LDA #$69
BCED-   85 38      STA  $38
BCEF-   A9 BA      LDA #$BA
BCF1-   85 39      STA  $39
```

Next we call the DOS subroutine that redirects its own input. Attempting to POKE directly into this vector ($AA55-$AA56) will cause the Apple to reply strangely.

```
BCF3-   20 51 A8   JSR $A851
```

In addition, we must correct the problem of self-modifying code by resetting the starting pointer.

```
BCF6-   A9 8C      LDA #$8C
BCF8-   8D 6D BA   STA $BA6D
```

With the initialization completed, we can jump to the DOS soft entry, confident that we will not receive a BASIC prompt that requires keyboard input. We will permit Apple to read its own input.

```
BCFB-   4C D0 03   JMP $03D0
```

Instead of looking to the keyboard, *via* the monitor, for an input character, we have directed it to $BA69.

```
BA69-   EE 6D BA   INC $BA6D
BA6C-   8D 8C BC   LDA $BC8C
```

Each return for another character moves the pointer upward from $BC8D until it loads a $00. The pointer starts at BC8D because it is incremented from BC8C before the first loading.

```
BA6F-   D0 18      BEQ $BA89
```

The branch is to another RTS later in the routine to save one byte. By loading the entire command, letter by letter, we will arrive at $00 which signifies the end of the input table.

Fortunately, $BA96 is always a 00 because it is part of the read translate table. Having loaded the final 00, we do not branch. The screen now has printed at the bottom: > RUNZMENU?

To find out the answer, we will jump to the read key subroutine in the monitor.

```
BA71-      20 1B FD      JSR $FD1B
```

When the key is pressed, the subroutine will return with the ASCII value in the accumulator. No matter what the value is, we must restore the input hooks back to the keyboard. While we are doing that we will save the accumulator value on the stack until $BA82.

```
BA74-         48              PHA
```

Now we must restore the input hooks.

Warning: pressing RESET again before the hooks are restored will cause a lockut of the keyboard and require rebooting DOS. This normally is not a problem unless you habitually press the RESET key repeatedly for extra effect.

```
BA75-     AD FE BC     LDA $BCFE
BA78-     85 38          STA $38
BA7A-     AD FF BC     LDA $BCFF
BA7D-     85 39          STA $39
```

With KSWL,H reloaded, we call DOS to move the input vectors to its satisfaction.

```
BA7F-      20 51 A8      JSR $A851
```

Now that we have finished housekeeping we can find out which key the operator pressed.

```
BA82-         68              PLA
```

If the operator had wanted simply to reset and not run the menu program, he would have started to type NO. We would only read one letter: 'N'. (The routine is so fast that it would be finished before the 'O' could have been entered.) The compare instruction will let us find out.

```
BA83-       C9 CE        CMP #$CE
```

The 'N' key equals ASCII $CE. If the value matches, we will branch to $BA8A.

```
BA8A-      4C 0D 03      JMP $03D0
```

That will take us into DOS softly with the RAM BASIC still in the language card — just like before the language card was installed! Also, the BASIC program being used is still present and active.

Let us assume that any other key was pressed. This routine doesn't care if a 'Y' were pressed, or the (RETURN) key, or any other key. We were only looking for a negative response. We can now load a return and tell Apple to execute the command that we automatically entered.

```
BA87-      A9 8D          LDA #$8D
BC89-      60                  RTS
```

The program is called ZMENU? for two reasons. First, during use, the question is obvious to an inexperienced user. And, during a printout of my master catalog, all the hello programs appear at the end. Actually, it doesn't matter how difficult the name is to type because you don't have to type it anymore!

I was very pleased with myself. I tried BA69G from the monitor and the routine worked perfectly. I initialized a disk. I booted it. The reset key wouldn't work! Entering the monitor, the program worked from $BCDFG. Yes, BCDFG ran my ZMENU? but reset wouldn't.

After some searching, I found that DOS sets the user reset to its own input point with each boot. I was determined to avoid BLOADing two bytes or POKEing values, so I changed DOS itself! The complete procedure is listed below in a step-by-step format.

Begin by inserting your Apple-provided system master diskette and boot. Then, type >NEW. If your resident language is Applesoft, > LOAD HELLO. If your resident language is Integer, > LOAD APPLESOFT. Next, type the following lines exactly as listed.

```
>CALL-151
 *BCDF: AD 55 AA 8D FE BC AD 56 AA
        8D FF BC A9 69 85 38 A9 BA
        85 39 20 51 A8 A9 8D 8D 6D
        BA 4C D0 03
 *BA69: EE 6D BA AD 96 BA D0 18 20
        1B FD 48 AD FE BC 85 38 AD
        FF BC 85 39 20 51 A8 68 C9
        CE FO 03 A9 8D 60 4C D0 03
 *BA8D: D2 D5 CE DA CD C5 CE D5 FB
 *9E31: 74 9E
 *9E3C: 73 9E
 *9E73: DF BC 19
 *3D0G
>INIT RESET MASTER
```

This will result in your neat, new system master diskette having a hello program called RESET MASTER. This will load the other BASIC into the language card when booting. To really make the new diskette convenient, I altered the program RESET MASTER by putting an extra line in it. The extra line causes the menu program to be run at the end of the booting process. For example,

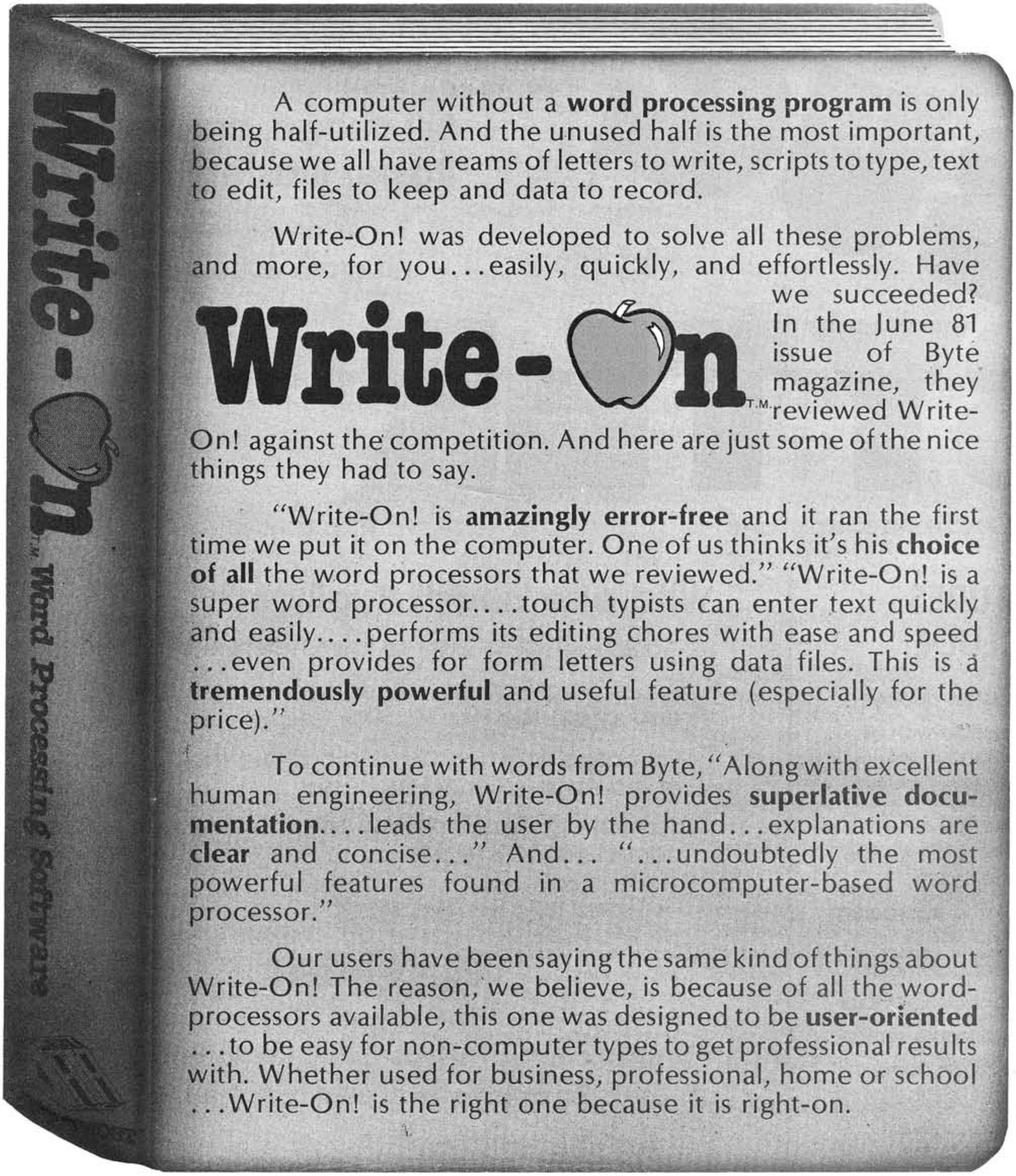
```
> 240 PRINT D$; "RUNZMENU?"
```

After initializing your new diskette,

```
>BRUN FID
```

and transfer the other version of BASIC and your menu program to the diskette. Your menu program (usually your hello-type program) must be saved as MENU?. (The question mark is part of the program name.) Diskettes not having that program will cause the routine to return the file not found error and the BASIC language prompt.

My children are pleased with the handiness of our Apple again.


Write-On
T.M. Word Processing Software
Write-On T.M.

Shape Manipulate

This Applesoft program lets you create a shape table, delete shapes, add shapes from other tables, or change the maximum number of shapes in a table.

John R. Raines
2170 Wellesley Ave.
St. Paul, Minnesota 55105

The ability of the Apple's BASICs to draw high-resolution shapes is useful in animation or picture-drawing programs. The improvement in "human factors" engineering[1], which shape tables permit, is tremendous. John Figueras' shape-creating program[2] makes the use of these shapes much more practical. Peter Cook combined Figueras' three programs with a menu, and also made several changes[3, 4].

Although these programs contributed to my work, this program allows much more flexibility in manipulating shape tables than either of its predecessors. The hardware required for this program is 48K RAM and one disk drive.

Program Description

This program is an extension of Figueras' work. It allows you to INITIALIZE a table, CREATE shapes, DELETE shapes from an existing table, ADD shapes from a second table, change the maximum NUMBER of shapes in a table, REVIEW the table, and SAVE the resulting table. The biggest change in Figueras' design of the program is that the various functions are combined in one program. You select the function(s) that you want to use by typing one-letter options. The various options can be used in nearly any order. The resulting flexibility is a sizable improvement. You can abort any command in progress and return to the option list by typing CTRL-0 in place of any input string. The code that deals with the option list is after line 10000.

The CREATE section is basically Figueras' code, but has been adapted to allow a 50 × 50 grid. The larger grid allows much bigger shapes. As you recall, Applesoft will increase the size of shapes according to the value of N in "SCALE=N." The result is often unsatisfactory; a diagonal line may become jagged, curves cannot be simulated with any sizeable SCALE factor. The larger grid helps, but does not correct the underlying problem.

Several other small changes to the CREATE section have also been made. When you have finished your shape, it is displayed with SCALE=1 and SCALE=2 before it asks if you want to SAVE it. Both are XDRAW outputs — funny things happen if you plot the same point twice in your shape definition and then XDRAW it. The enlarged version lets you assess jaggedness or "holes" that may appear in your shape when it is enlarged. Many of the problems with shapes, which are revealed in this display, can be eliminated by more careful shape definition. By avoiding enlargement and XDRAW in the programs you write to use the shape tables created here, you can completely circumvent these problems. It is always easier to use SCALE=1 and DRAW than to try to CREATE shapes that work well with SCALE=2 and XDRAW. Shapes can be erased with HCOLOR=0 and DRAWing over them in this approach.

If you are creating several shapes in one session, the option of not erasing the previous shape may be useful. This shows you how shapes will fit together when they are drawn later. The CREATE section in this progam, like Figueras' but unlike Cook's program, assumes that the first shape in shape table 1 is the cursor. This is set up by INITIALIZE. You can DELETE the cursor later, but if you do, CREATE will not work properly on that table. The CREATE section is from line 10 through line 1399.

The INITIALIZE section just clears out shape table 1 and puts the cursor definition into it. This code is virtually unchanged from Figueras' program. This section begins at line 20000.

There are two load sections: one for each of the two shape tables. The second shape table is useful only for the ADD shape function. The options to load a table are "1" or "2." Note that DELETE, ADD, SAVE, REVIEW, NUMBER, and CREATE are commands that only make sense after you have either initialized or loaded. These sections check to be sure that the appropriate tables do exist and give an error message if you have forgotten this step. Unlike Figueras' program, the location of the shape table buffers and their length are fixed in this program. The length is always 2048 locations. The load sections are between lines 22000 and 23010.

The DELETE section, all new, removes a shape from table 1. When this option is selected it asks which shape you want to DELETE, displays it in two sizes and asks if that is the correct shape. If you say "N", it asks for another shape number. If you say "Y", then it removes the shape, moves the rest of the shapes to fill in the gap and corrects the index. The "move" is done by using a routine in the Apple monitor.

When written in Applesoft, the move operation took many seconds. The BASIC code to perform the move is in line 21060. The actual move code is located after line 48000, in a subroutine. The subroutine I use is general, and uses parameters called FROM, DESTination and FINISh (the latter is the last location from which a byte is moved). The monitor move subroutine is listed (with the rest of the monitor ROM) in the *Apple II Reference Manual.*

The ADD option is for adding a shape in table 2 to table 1. The routine asks what shape you want, displays it, and asks if that's correct. If you say "N", it asks for another shape number. If you say "Y", it adds the indicated shape to the end of table 1 and updates the index. The monitor move routine is used here, too (see "DELETE" above). The ADD section is about line 24000.

The REVIEW command shows you the shapes in table 1. Fewer shapes are shown on each frame than in either Cook's or Figueras' programs, because the shapes are potentially 50 × 50 here, rather than 15 × 15. When one screenful has been displayed, the program waits for you to hit return before displaying the next screenful. Shapes are displayed in numeric order, from left to right and top to bottom.

The NUMBER command lets you change the maximum number of shapes in shape table 1. Shape table 2 is used for temporary storage during this operation and is left empty at the end. The new maximum may be either more or less than the old maximum, but must be at least one more than the number of shapes currently in the table.

About Line 1

If you don't know the following trick, and have a 48K Apple and use high-resolution graphics (with or without shapes), then this is the best tidbit in the article. With "POKE 103,1:POKE 104,64:POKE 16384,0" before loading your program, you load it above the first high-resolution graphics page. There may be a lot more room there than below it. Below the first high-resolution graphics page, your program has from 2048 to 8191 — just 6K. Above it, with a 48K Apple and DOS, your program has from 16384 to 38400 — nearly 22K bytes.

I have not succeeded in getting the variables and strings into the space below the first high-resolution graphics page, so the actual increase is a bit less than this. Before I learned this trick, I was chaining frequently to get programs to fit into the little 6K segments available. I was nearly fed up with high resolution graphics because of this problem.

Figure 1 shows the core layout for this program. Variations on this technique are discussed in references 5 and 6.

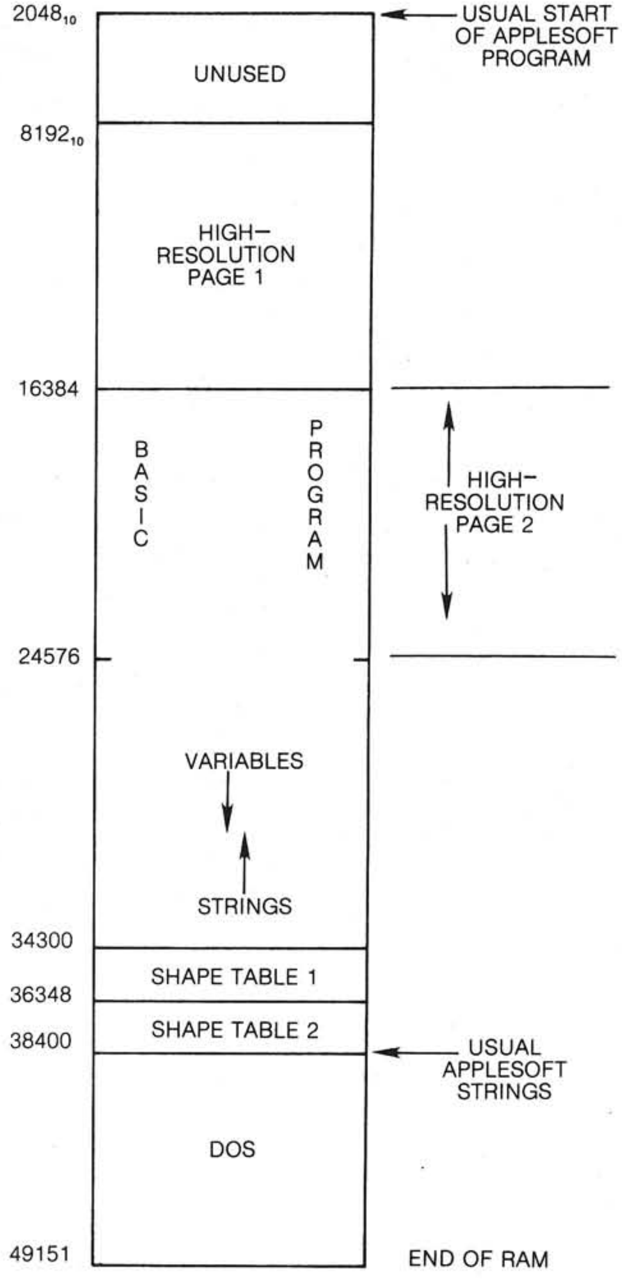
Control-0

The CTRL-0 key returns you from whatever command you are in to the "option" display, wherever this program accepts keyboard input. This is implemented mainly at line 30000, a subroutine called by any part of the program which would normally do an "input" statement. Line 700 handles CTRL-0 for the cursor move section of the CREATE command (where input is obtained *via* a GET). Frequent use of CTRL-0 should be discouraged, however, since the stack is not cleaned up and might eventually fill.

The Shape Table Buffers

The buffers for the shape tables are in fixed locations in this program, unlike Figueras' program. This saves some typing and has not caused me any problems. The buffers are just 2K in size and are located in the 4K of RAM

Figure 1



just below DOS (see figure 1). The location and size of the buffers is defined in line 5, and it should be possible to move them or change their sizes just by changing this line (although I have not done so).

ASVE is the start of shape table 1. BS is the buffer size. A2 is the start of shape table 2. The program will warn you if you expand table 1 so that it overwrites table 2. If this happens, you should be able to SAVE table 1, but will be unable to use table 2. Conceivably the buffers could be expanded to 3K by putting them in the unused RAM between 2048 and 8191.

To load and use one of the shape tables created with this program, you proceed just as with either Figueras' or Cook's programs. First you BLOAD *filename,Alocation* (the "A" parameter may be needed more often with my program, but really should be made explicit with any binary file for clarity). Next you need to tell Applesoft where the table is, by POKEing the same location into locations 232 and 233. One example of this is line 26010.

I think the Applesoft manual and Figueras' article, together, constitute an adequate description of the shape table, and I will not repeat that description here. The most common error in writing functions such as I have written here, is to forget about the special case of the last index entry. This entry usually points to the first byte after the last shape (to the place where an undefined shape will begin). When the table is full (when N = MAX), there is no pointer to a "next" entry. The second most common error is to forget that the index entry is an offset, not an absolute pointer to the shape definition. I have left a debugging subroutine in place at 39000. You can just leave it out if you like, but it is useful in the following fashion:

1. Go through the part of the command sequence you are sure works.
2. Do one command you are suspicious of.
3. When the "option" display appears, type CTRL-C, return.
4. Type in a line such as C = 34300 : GOSUB 39000 — this will dump the first 50 bytes from shape table 1 onto the TV. Examine these to see if you identify an error.
5. Type "CONT".
6. GOTO step 2.

This program does not incorporate the changes to Figueras' programs discussed by Cook. Depending on your specific needs, you may want to use some of Cook's ideas, such as allowing blank shapes. It would seem worthwhile to read the reference articles from MICRO before you start typing any of the programs into your machine.

References

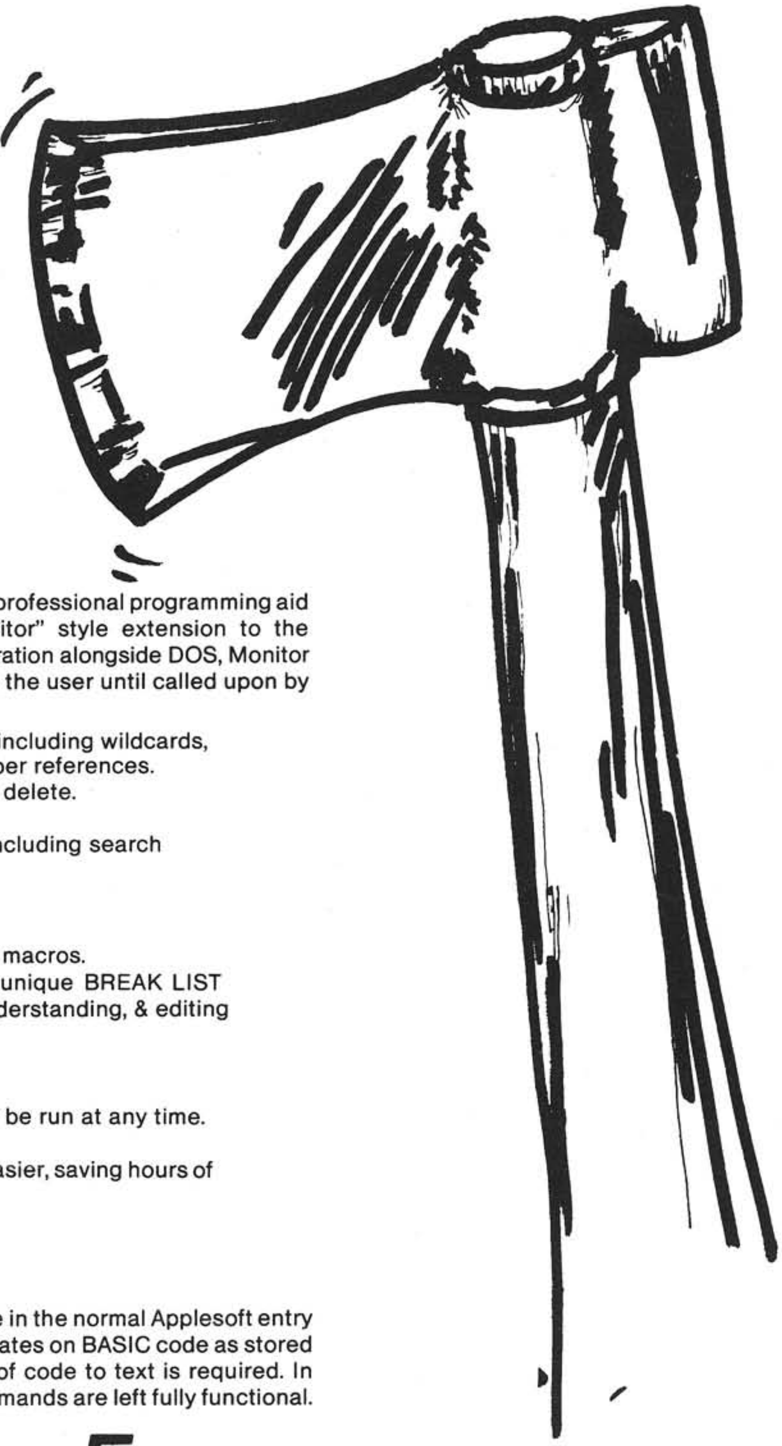
1. Shneiderman, Ben, *Software Psychology; Human Factors in Computer and Information Systems.* Winthrop Publishers, Inc., Cambridge, MA, 1980.
2. Figueras, John, "How to do a Shape Table Easily and Correctly," MICRO 19:11, reprinted in *MICRO/Apple I*, p. 78.
3. Cook, Peter, "Creating Shape Tables, Improved," MICRO 28:7.
4. Cook, Peter, "Microbes and Updates," MICRO 31:76.
5. Guild, George S., "Applesoft Program Relocation," MICRO 19:19.
6. Kluepfel, Charles, "Applesoft Program Splitter Modifications," *CALL —A.P.P.L.E.*, October 1980, pg. 45.

```
1  IF  PEEK (1C4) <  > 64 THEN  PCKE 16384,C: PCKE 1C3,1: PCKE 1C4,64: PRINT
      CHR$ (4);"RUN SHAPE MANIPULATE": REM IF WE WERE LCADED EELCW HI-RES
      PAGE1 THEN RE-LCAD; PRCGRAM DCESN'T FIT IN 6K
5  TEXT : SPEED= 255: HCME : PRINT "SHAPE MANIPULATE": HIMEM: 34299:DS =
      CHR$ (4):BS = 2C48:ASVE = 343CC:A2 = ASVE + BS: REM ASVE & A2 ARE S
     HAPE TABLE EUFFER LCCATICNS,BS IS EUFFER SIZE
6  DEF  FN PK(X) =  PEEK (X) + 256 *  PEEK (X + 1)
9  VTAB 22: GCTC 1CCCC
10  PRINT  TAB( 1C);"SHAPE CREATE"
3C  PRINT  TAB( 5);"BY J. FIGUERAS, RCCHESTER N.Y."
6C  IF NAME$ = "" THEN  PRINT "NEED TO INIT CR LCAD FIRST": GCTC 1CCCC
120  REM   GET MAX NC. CF SHAPES (SPEC'D AT INIT)
13C MAX = ( FN PK(ASVE + 2) - 2) / 2
15C  REM   GET NC. SHAPES IN TABLE
16C N =  PEEK (ASVE):FLAG = 1
17C  REM   CET FILE LENCTH
18C INDEX =  PEEK (ASVE + 2 * N + 2) + 256 *  PEEK (ASVE + 2 * N + 3)
19C  REM   CCMPUTE ADDRESS CF NEXT FREE BYTE
20C ADDR = ASVE + INDEX
22C  IF MAX > N THEN 260
23C  PRINT "SHAPE TABLE FULL."
240  GOTC 1000C
25C  REM   SETUP INTERNNAL POINTERS TC TABLE
26C  PCKE 232,ASVE - 256 *  INT (ASV / 256): PCKE 233, INT (ASVE / 256)
27C  REM   UPDATE SHAPE CTR
28C N = N + 1: PCKE ASVE,N
29C  REM  DISPLAY PLCTTING GRID. INIT CTR, CYCLE
300  HCOLCR= 3: SCALE= 1: RCT= C:CYCLE = 0: IF HG = 1 THEN  PRINT "ERASE
     SCREEN (Y CR N) ?";: GCSUB 30000: IF AN$ = "Y" THEN  HGR
310  IF HG = 0 THEN  HGR :HG = 1
320  FCR X = C TC 15C STEP 3: HPLOT X,C TC X,15C: NEXT
33C  FCR Y = 0 TC 150 STEP 3: HPLCT 0,Y TC 15C,Y: NEXT
350  PRINT : PRINT : PRINT : PRINT : PRINT "ENTER STARTING GRID CCORDS."
370  PRINT "X ";: GCSUB 30000:X = 3 * AN - 2
38C  PRINT "Y ";: GCSUB 300CC:Y = 3 * AN - 2
390  DRAW 1 AT X,Y:XS = X:YS = Y
41C  PRINT : PRINT : PRINT : PRINT
420  PRINT "MCVE PLCT CURSCR WITH KEYS": PRINT " I J K M    P=PLCT   Q=QU
     IT"
```

(Continued)

```
460 KEY$ = "":KSVE$ = "": GOTO 570
470  REM  FLAG RE-ENABLES CURSOR AFTER A PLOT DISABLE
480  IF FLAG = 1 THEN 520
490  REM  ERASE PREVIOUS CURSOR
500  XDRAW 1 AT X1,Y1
510  REM  PLOT NEW CURSOR
520 X1 = X:Y1 = Y:FLAG = 0
530  XDRAW 1 AT X,Y
540  REM  SAVE LAST TWO KEYSTOKES. KI$ IS NEEDED FOR ERASE ROUTINE
550 KI$ = KSVE$:KSVE$ = KEY$
570  GET KEY$
580  REM  GO TO SIEVE TO GET 3-BIT PLOT VECTOR FROM KEY$ AND KSVE$
590  IF KEY$ = "I" THEN SYMBOL = 0:Y = Y - 3: GOTO 760
610  IF KEY$ = "K" THEN SYMBOL = 1:X = X + 3: GOTO 760
630  IF KEY$ = "M" THEN SYMBOL = 2:Y = Y + 3: GOTO 760
650  IF KEY$ = "J" THEN SYMBOL = 3:X = X - 3: GOTO 760
670  IF KEY$ = "P" THEN FLAG = 1: GOSUB 1000: GOTO 530
690  IF KEY$ = "Q" THEN 1090
700  IF KEY$ =  CHR$ (15) THEN 10000: REM  CTRL/O
710  IF KEY$ <  > "E" THEN 570
720  HCOLOR= 0:FLAG = 0: GOSUB 1000
730  REM  SET UP PRE-PLOT STATUS
740 KSVE$ = KI$: HCOLOR= 3: GOTO 500
750  REM  ADJUST 3-BIT VECTOR FOR PLOT
760  IF KSVE$ = "P" THEN SYMBOL = SYMBOL + 4
780 CYCLE = CYCLE + 1
790  IF CYCLE = 1 THEN BYTE = SYMBOL: GOTO 480
810  IF CYCLE <  > 2 THEN 900
820 BYTE = BYTE + 8 * SYMBOL: IF BYTE > 7 THEN 480: REM  GUARD AGAINST EN
     D OF SHAPE FLAG!
860 BYTE = BYTE + 8: POKE ADDR,BYTE:ADDR = ADDR + 1
870  REM  ENTER UP MOVE AND DUMMY LEFT MOVE IN NEW BYTE
880 BYTE = 24:CYCLE = 2: GOTO 480
890  REM  IF 3RD 3-BIT IS A MOVE ONLY, FINISH BYTE.  ELSE LOAD BYTE INTO
     TABLE AND STORE 3-BIT VECTOR IN NEXT BYTE.
900  IF SYMBOL > 3 THEN 930
910 BYTE = BYTE + 64 * SYMBOL
930  POKE ADDR,BYTE:ADDR = ADDR + 1
940  REM  STORE 3-BIT VECTOR IN NEXT BYTE IF NEEDED.
950  IF SYMBOL = 0 OR SYMBOL > 3 THEN 980
960  REM  PREPARE FOR NEXT BYTE.  GET NEXT 3-BIT VECTOR
970 CYCLE = 0: GOTO 460
980 CYCLE = 1:BYTE = SYMBOL: GOTO 480
1000  REM
1010  REM
1020  FOR Y2 = Y - 1 TO Y + 1: HPLOT X - 1,Y2 TO X + 1,Y2: NEXT
1030  REM  TURN OFF CURSOR IN PLOTTED SQUARE
1040  IF X = XS AND Y = YS THEN  RETURN
1050  XDRAW 1 AT X,Y: RETURN
1060  REM  PREPARE BYTE FOR QUIT
1070  REM  CLOSE OUT BYTE FOR MOVE-ONLY
1080  IF KSVE$ <  > "P" THEN 1150
1090  REM  USE PLOT-THEN-UP VECTOR TO END
1100  IF CYCLE = 2 THEN  POKE ADDR,BYTE:ADDR = ADDR + 1
1120  IF CYCLE = 1 THEN BYTE = BYTE + 32: GOTO 1150
1140 BYTE = 4
1150  POKE ADDR,BYTE:ADDR = ADDR + 1:AN = ADDR: GOSUB 31000
1160  REM  ADD RECORD MARK.  DISPLAY NEW SHAPE
1170  POKE ADDR,0:ADDR = ADDR + 1: XDRAW N AT 200,25: SCALE= 3: XDRAW N AT
     200,100: SCALE= 1
1180  PRINT : PRINT "SAVE SHAPE (Y OR N) ";: GOSUB 30000:KI$ = AN$
1190  IF KI$ = "Y" THEN 1220
1200 N = N - 1: GOTO 180
1210  REM  GET INDEX FOR NEXT FREE BYTE
1220 N = N + 1:ADDR = ADDR - ASVE
1230  IF N < MAX THEN 1270
1240  PRINT : PRINT "TABLE FULL WITH THIS SHAPE "
1250  IF N > MAX THEN 1310
1260  REM  STORE INDEX IN DIRECTORY
1270  POKE ASVE + 2 * N,ADDR - 256 *  INT (ADDR / 256): POKE ASVE + 2 * N
     + 1, INT (ADDR / 256)
1290  PRINT : PRINT "DONE WITH CREATE (Y OR N) ";: GOSUB 30000:KI$ = AN$
1300  IF KI$ = "N" THEN 160
1310  GOTO 10000
10000 HG = 0: PRINT : PRINT "CREAT,INIT,DEL,LOAD1,L2,ADD,SAVE,REV,NO.";: PRINT
     "OPTION (C,I,D,1,2,A,S,R,N)";: GOSUB 30000:AL$ = "CID12ASRN":I = 1
10020  IF AN$ =  MID$ (AL$,1,1) OR I >  LEN (AL$) THEN 10040
10030 I = I + 1: GOTO 10020
10040  ON I GOTO 10,20000,21000,22000,23000,24000,25000,26000,27000: PRINT
     "PLEASE TYPE C,I,D,1,2,A,S,R OR N": GOTO 10000
20000  REM  SHAPE TABLE INIT FROM MICRO 19:19 DEC 1979, J. FIGUERAS, ROCH
     EST N.Y
20020 NAME$ = " ": REM  SEE LINE 60
20030 ADDR = ASVE
20040  PRINT "NO. SHAPES TO BE ALLOWED ";: GOSUB 30000:N = AN
20060  FOR I = 0 TO 2 * N + 1: POKE ADDR + I,0: NEXT
20080  REM   CALC INDEX TO CURSOR
```

Advanced X-tended Editor



The Advanced X-tended Editor (AXE) is a professional programming aid which provides the user with a "text-editor" style extension to the standard Applesoft* operation system. Operation alongside DOS, Monitor and Applesoft, AXE remains transparent to the user until called upon by one of over **thirty** commands.

- Global search & replacement, including wildcards, selected changes, & line number references.
- Full character insert, gobble & delete.
- Full statement insert & delete
- Enhanced cursor movement including search ahead & position.
- Two packed list edit modes.
- Recall editing modes.
- User programmable keyboard macros.
- Four LIST formats, including unique BREAK LIST format for easier reading, understanding, & editing of code.
- Auto line-numbering.
- Lower case character entry.
- Resident BASIC program may be run at any time.
- Many more features.
- Develop programs quicker & easier, saving hours of programming cost.
- Requirements:
  Apple II/II+,* Applesoft,
  DOS 3.3, 48K Ram

Commands are easy, logical, and operate in the normal Applesoft entry mode, or in AXE's editing modes. AXE operates on BASIC code as stored in memory by Applesoft. No conversions of code to text is required. In addition, all Apple II DOS and Monitor commands are left fully functional.

Advanced X-tended Editor

*Apple II/II+, and Applesoft are registered trademarks of Apple Computer, Inc.

Available at your local dealer
An invaluable tool at only $69.95

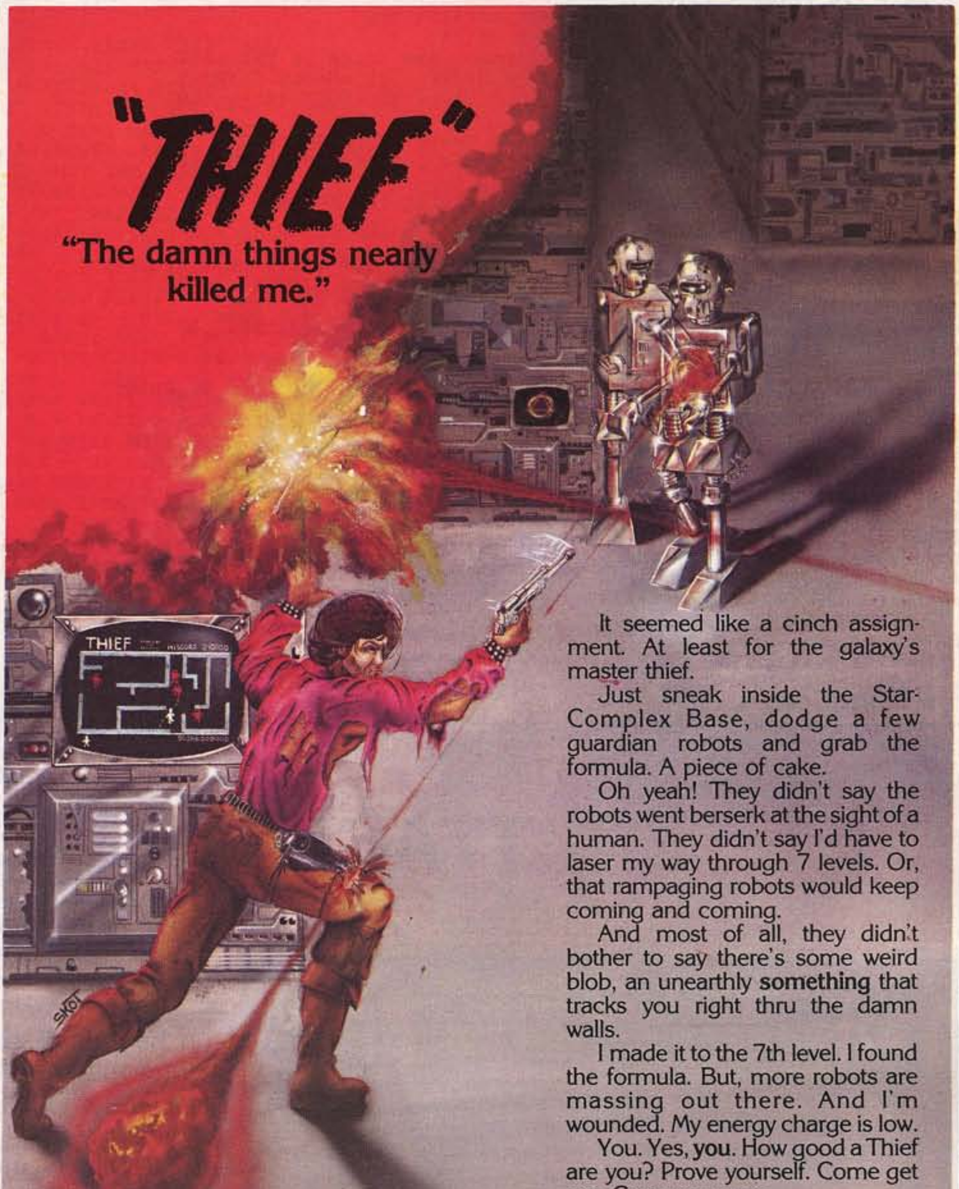
Versa Computing, Inc.

3541 Old Conejo Road, Suite 104
Newbury Park, CA. 91320 **(805)498-1956**

```
20090 N = 2 * N + 2
20100  REM   PUT CURSOR INTO DIRECTORY
20110  POKE ADDR + 2,N - 256 *  INT (N / 256): POKE ADDR + 3, INT (N / 256)
20130  REM  CALC INITIAL ADDR TO CURSOR
20140 INIT = ADDR + N: RESTORE
20150  REM  ENTER CURSOR SHAPE VECTORS
20160  DATA 62,36,45,54,04,00
20170  FOR I = 0 TO 5: READ A: POKE INIT + I,A: NEXT
20190  REM GET INDEX TO NEXT SHAPE
20200 N = N + 6: POKE ADDR + 4,N - 256 *  INT (N / 256): POKE ADDR + 5, INT
      (N / 256)
20250  POKE ADDR,1: REM  UPDATE SHAPE CTR
20270  GOTO 10000
21000  REM  DELETE A SHAPE FROM TABLE #1
21010 N =  PEEK (ASVE): PRINT "SHAPE NUMBER TO DELETE?";: GOSUB 30000:SH =
      AN: HGR : HCOLOR= 3: SCALE= 1: ROT= 0: POKE 232,ASVE - 256 *  INT (A
      SVE / 256): POKE 233, INT (ASVE / 256): IF SH > N THEN  PRINT "HIGHE
      ST SHAPE # IS ";N: GOTO 21010
21015  DRAW SH AT 80,80: SCALE= 2: DRAW SH AT 190,80: SCALE= 1
21020  PRINT "IS THAT THE SHAPE TO BE DELETED (Y OR N) ?";: GOSUB 30000:K
      I$ = AN$: IF (KI$ <  > "Y") AND (KI$ <  > "N") THEN 21020
21030  IF KI$ = "N" THEN 21010
21040 LS =  FN PK(ASVE + 2 * SH):LN =  FN PK(ASVE + 2 * SH + 2):MAX = ( FN
      PK(ASVE + 2) - 2) / 2
21045  IF MAX > N + 1 THEN MAX = N + 1
21050 SIZE = LN - LS: FOR I = ASVE + 2 * SH TO ASVE + 2 * MAX - 2 STEP 2:
      X =  FN PK(I + 2) - SIZE: POKE I,X - 256 *  INT (X / 256): POKE I +
      1, INT (X / 256): NEXT : POKE ASVE + 2 * MAX,0: POKE ASVE + S * MAX +
      1,0: REM  ADJUST SHAPE DIRECTORY
21060  REM    FOR I = ASVE + LN TO ASVE+BS-1  : POKE I - SIZE, PEEK (I):
      NEXT : REMMOVE SHAPES--THIS WORKS BUT IS AWFULLY SLOW.
21065 FROM = ASVE + LN:DEST = ASVE + LS:FINIS = ASVE + BS - 1: GOSUB 4800
      0: REM USE MACHINE MOVE SUBROUTINE
21070  POKE ASVE,N - 1: GOTO 10000
22000  REM  SHAPE TABLE #1 LOAD
22010  PRINT "SHAPE TABLE NAME :";: GOSUB 30000:NAME$ = AN$: PRINT D$;"BL
      OAD";NAME$;",A";ASVE: GOTO 10000
23000  REM  SHAPE TABLE #2 LOAD
23010  PRINT "SHAPE TABLE NAME :";: GOSUB 30000:N2$ = AN$: PRINT D$;"BLOA
      D";N2$;",A";A2: GOTO 10000
24000  REM   ADD SHAPE FROM TABLE2 TO TABLE1
24010  IF NAME$ = "" THEN  PRINT "NO TABLE #1. INIT OR LOAD FIRST": GOTO
      10000
24020  IF N2$ = "" THEN  PRINT "NO TABLE #2.  USE OPTION '2' TO LOAD IT":
       GOTO 10000
24030 N2 =  PEEK (A2): PRINT "SHAPE NUMBER TO ADD?";: GOSUB 30000:SH = AN
      : HGR : HCOLOR= 3: SCALE= 1: ROT= 0: POKE 232,A2 - 256 *  INT (A2 /
      256): POKE 233, INT (A2 / 256): IF SH > N2 THEN  PRINT "HIGHEST SHAP
      E IN TABLE2 IS ";N2: GOTO 24030
24040  DRAW SH AT 80,80: SCALE= 2: DRAW SH AT 190,80: SCALE= 1
24050  PRINT "IS THAT THE RIGHT SHAPE (Y OR N)?";: GOSUB 30000:KI$ = AN$:
       IF (KI$ <  > "Y") AND (KI$ <  > "N") THEN 24050
24060  IF KI$ = "N" THEN 24030
24070 MAX = ( FN PK(ASVE + 2) - 2) / 2:N =  PEEK (ASVE): IF N >  = MAX THEN
       PRINT "SHAPE TABLE1 FULL": GOTO 10000
24100 LS =  FN PK(A2 + 2 * SH):LAST = ASVE + (2 * N) + 2:LN =  FN PK(LAST
      ):FINIS = A2 +  FN PK(A2 + 2 * SH + 2) - 1
24120 FROM = A2 + LS:DEST = ASVE + LN: GOSUB 48000: REM  USE MACHINE MOVE
       FOR SHAPE
24140 SIZE = FINIS - FROM + 1:TM = LN + SIZE: POKE ASVE,N + 1:AN = TM + A
      SVE: GOSUB 31000: IF N + 1 <  > MAX THEN  POKE LAST + 2,TM - 256 *  INT
      (TM / 256): POKE LAST + 3, INT (TM / 256): REM    FIX DIRECTORY
24160  GOTO 10000
25000  REM  SAVE SHAPE TABLE #1
25005  IF NAME$ = "" THEN  PRINT "NEED TO INIT OR LOAD FIRST": GOTO 10000

25010  PRINT "SHAPE TABLE NAME :";: GOSUB 30000:NAME$ = AN$:N =  PEEK (AS
      VE): REM    NO. SHAPES IN TABLE
25020 LAST = ASVE + 2 * N + 2:: REM INDEX ENTRY FOR "NEXT" SHAPE
25030 ADDR =  PEEK (LAST) + 256 *  PEEK (LAST + 1): PRINT D$;"BSAVE";NAME
      $;",A";ASVE;",L";ADDR: GOTO 10000
26000  IF NAME$ = "" THEN  PRINT "NEED INIT OR LOAD FIRST": GOTO 10000
26010 N =  PEEK (ASVE): POKE 232,ASVE - 256 *  INT (ASVE / 256): POKE 233
      , INT (ASVE / 256): HGR : HCOLOR= 3: SCALE= 1: ROT= 0
26020 X = 25:Y = 25:I = 1
26030  DRAW I AT X,Y:X = X + 60: IF X > 254 THEN X = 25:Y = Y + 60: IF Y >
      124 THEN Y = 25: IF I <  > N THEN  PRINT "HIT RETURN TO GO ON.";: GOSUB
      30000: HGR
26040 I = I + 1: IF 1 > N THEN 10000
26050  GOTO 26030
27000  REM  "N" -- CHANGE MAX NO. SHAPES ALLOWED IN TABLE 1.  DESTROYS TA
      BLE 2 BY USING IT AS TEMPORARY STORAGE.
27010 N =  PEEK (ASVE): IF NAME$ = "" THEN  PRINT "NEED TO INIT OR LOAD F
      IRST.": GOTO 10000
27020 OLDMAX = ( FN PK(ASVE + 2) - 2) / 2: PRINT "NEW MAXIMUM NUMBER OF S
      HAPES :";: GOSUB 30000:NOWMAX = AN: IF N + 1 > NOWMAX THEN  PRINT "T
      ABLE ALREADY HAS ";N;" SHAPES.": GOTO 27020
```

```
27030 FROM = ASVE:DEST = A2:FINIS = ASVE + BS - 1: GOSUB 48000:N2$ = "": REM
     COPY TABLE1 TO TABLE2
27040 S1 = NOWMAX * 2 + 2:TM =  FN PK(A2 + 2) - S1: REM  S1 IS INDEX ENTR
     Y FOR NEW CURSOR.  TM=(INDEX OF OLD)-(INDEX OF NEW)
27050  FOR I = 1 TO N + 1:P2 =  FN PK(A2 + I * 2):NP = P2 - TM: POKE ASVE
      + I * 2,NP - 256 *  INT (NP / 256): POKE ASVE + I * 2 + 1, INT (NP /
     256): NEXT : REM CREATE THE NEW INDEX FOR TABLE 1, CORRECTING THE OL
     D ENTRY (P2) BY THE OFFSET (TM).
27060  IF N + 1 < NOWMAX THEN  FOR I = N + 2 TO NOWMAX: POKE ASVE + I * 2
     ,0: POKE ASVE + I * 2 + 1,0: NEXT : REM FILL IN EMPTY INDEX SLOTS WI
     TH ZEROES
27070 FROM = A2 +  FN PK(A2 + 2):DEST = ASVE +  FN PK(ASVE + 2):FINIS = A
     2 + BS - TM: GOSUB 48000: REM  MOVE SHAPE DEFINITIONS
27080  IF N <  > OLDMAX THEN 10000: REM  INDEX IS OK.
27090 ADDR = ASVE + N * 2:P2 = ASVE +  FN PK(ADDR): REM  ADDR IS THE LAST
      INDEX ENTRY; IT IS OK BUT THE NEXT ONE IS WRONG.  IT SHOULD POINT T
     O WHERE THE NEXT SHAPE BEGINS BUT SINCE THE TABLE WAS FULL IT IS (TH
     E FIRST 2 BYTES OF SHAPE1) -TM
27091  REM  SHAPE1 IS OK, IT WAS COPIED INTO THE TABLE SOMEWHERE ELSE.
27100  IF  PEEK (P2) <  > 0 THEN P2 = P2 + 1: GOTO 27100: REM FIND THE EN
     D OF THE LAST SHAPE
27110 P2 = P2 + 1 - ASVE: POKE ADDR + 2,P2 - 256 *  INT (P2 / 256): POKE
     ADDR + 3, INT (P2 / 256)
27120  GOTO 10000
30000  INPUT AN$:AN =  VAL (AN$): IF ( LEFT$ (AN$,1) =  CHR$ (15)) OR ( RIGHT$
     (AN$,1) =  CHR$ (15)) THEN 10000: REM CTRL/O RETURNS TO LINE 10000
30010  RETURN
31000  IF AN >  = ASVE + BS THEN  PRINT "2K BUFFER HAS BEEN EXCEEDED.  DO
      NOT": PRINT "USE TABLE 2 UNTIL YOU SAVE TABLE 1.":N2$ = ""
31010  RETURN
39000  FOR I = 0 TO 0 + 50 STEP 10: PRINT "AD=";I;"  ";: FOR J = 0 TO 9: PRINT
      PEEK (I + J);" ";: NEXT : PRINT : NEXT
39010  RETURN
48000  POKE 60,FROM - 256 *  INT (FROM / 256): POKE 61, INT (FROM / 256):
      POKE 66,DEST - 256 *  INT (DEST / 256): POKE 67, INT (DEST / 256)
48010  POKE 62,FINIS - 256 *  INT (FINIS / 256): POKE 63, INT (FINIS / 25
     6): REM   NOW PARAMETERS FOR MOVE ROUTINE ALL SET EXCEPT "Y" HARDWAR
     E REGISTER
48020  POKE 768,152: POKE 769,72: POKE 770,160: POKE 771,0: POKE 772,32: POKE
     773,44: POKE 774,254: POKE 775,104: POKE 776,168: POKE 777,96: REM
     TYA:PHA:LDY#0:JSR FE2C:PLA:TAY:RTS -- SAVE Y, ZERO IT, CALL MOVE, RE
     STORE Y, RETURN
48030  CALL 768: RETURN
63999 D$ =  CHR$ (4): PRINT D$;"OPEN LISTING": PRINT D$;"WRITE LISTING": POKE
     33,30: LIST : PRINT D$;"CLOSE LISTING": TEXT : REM  LIST THIS PROGRA
     M ONTO DISK WITH "RUN 63999"
```

MICRO™



It seemed like a cinch assignment. At least for the galaxy's master thief.

Just sneak inside the Star-Complex Base, dodge a few guardian robots and grab the formula. A piece of cake.

Oh yeah! They didn't say the robots went berserk at the sight of a human. They didn't say I'd have to laser my way through 7 levels. Or, that rampaging robots would keep coming and coming.

And most of all, they didn't bother to say there's some weird blob, an unearthly **something** that tracks you right thru the damn walls.

I made it to the 7th level. I found the formula. But, more robots are massing out there. And I'm wounded. My energy charge is low.

You. Yes, **you**. How good a Thief are you? Prove yourself. Come get me. Come save me.

$29.95 for Apple II*. At computer stores, or from DATAMOST.

VISA/MASTERCHARGE accepted. $1.00 shipping/handling charge. (CA residents add 6% tax)

DATAMOST
19273 Kenya St., Northridge, CA 91326, (213) 366-7160

*Apple II is a trademark of Apple Computer, Inc.

MICRO

From Here to Atari

Jim Capparell
297 Missouri St.
San Francisco, California 94107

This month we will look at the DOS II file structure and the floppy disk system. We shall investigate what the I FORMAT option of the DOS II menu does and look at the physical characteristics of a mini-floppy.

Included are two programs which print out the directory files of a disk. Program 1 is written in BASIC and provides the ability to list the disk directory without having to access the DOS menu first. Program 2 also lists a disk's directory, and its distinction is the use of the FORTH language. (This FORTH is from Quality Software.)

A floppy disk is nothing but flexible mylar coated with a substance that will hold a magnetic charge. The floppy itself is protected by an envelope whose interior is designed to clean the disk surface as it spins. A slot is cut in the envelope allowing the read/write head access to the magnetic surface. The other two noticeable physical characteristics of the disk envelope are the write protect notch on the left edge and the index hole near the hub.

When the write protect notch is covered, the disk hardware is prevented from writing data to disk, affording some protection from inadvertent erasure. The index hole is used by the hardware to find the start of the first sector on a track (not used by Atari).

The format of Atari disks is known as soft sectored. Software provides the sector marks rather than the index holes. This formatting is performed whenever the I option of the DOS II menu is selected. At format time the disk surface is divided into 18 pie-shaped wedges. The beginning of every wedge has a preamble or header written which identifies the particular sector by number. This header is followed by the actual data, which is then followed by a gap.

Program 1

```
10 REM ***                  ***
20 REM USE THIS PROGRAM TO RETRIEVE A DIRECTORY LISTING
25 REM WITHOUT EXITING TO DOS MENU
30 REM INCLUDE AS A SUBROUTINE AT START OF EVERY EDITING SESSION
100 GOSUB 30010
120 STOP
30010 DIM FILENAME$(48)
30020 OPEN #1,6,0,"D:*.*":REM OPEN DIRECTORY FILE
30030 TRAP 30100
30040 INPUT #1,FILENAME$
30050 ? FILENAME$
30060 GOTO 30040
30100 CLOSE #1
30110 RETURN
```

Program 2

```
SCR # 101
 0 ( TYPE LST TO LIST FILENAMES )
 1 : SPC SPACES ;
 2 : DIR ( DIR ENTRY - POS BUFADR)
 3  1 - 8 /MOD 301 + BLOCK ;
 4 : HEAD CR ." TOTAL" 2 SPC
 5  ." START" 2 SPC ." FILENAME"
 6  CR ." SECT" 3 SPC ." SECT" 3
 7  SPC CR CR ;
 8
 9 : DIRNAM ( ADR - ..) DUP 16 +
10  SWAP 5 + DO I C@ EMIT LOOP ;
11 : GETLST DIR SWAP DROP HEAD DUP
12  128 + SWAP DO I 1+ @ . 4 SPC
13  I 3 + @ . 4 SPC I DIRNAM CR
14  16 +LOOP ;
15 : LST 64 1 DO I GETLST 8 +LOOP ;
```

Table 1

810 disk drive
18 sectors/track
40 tracks/disk
720 sectors/disk (40 * 18)
128 bytes/sector
92160 bytes/disk surface *

* This is nominal total. When using DOS II files it must be reduced by 13 sectors used in DOS file structure (see table 2) and by an additional three bytes for each of remaining 707 sectors (file number, forward pointer, byte count).

13 * 128 = 1664
3 * 707 = 2121
Total = 3785

Total capacity when using DOS II format is 92160 − 3785 = 88375 bytes/disk.

Table 2

DOS II sector allocation
3 sectors used for boot
1 sector used for VTOC
8 sectors used for file directory
1 sector unused due to numbering discrepancy between FMS and disk controller
13 sectors total

In addition to the sector division, each disk is also arranged in 40 concentric circles known as tracks. It is upon these tracks that data is recorded within a sector. As noted, every track is divided into 18 sectors. Refer to table 1 and table 2 for further information.

The formatting process (1) lays out sector arrangement by number on 40 tracks; (2) writes Atari DOS file structure on disk; (3) initializes every sector of disk to zero.

The Atari DOS expects certain information at specific sector positions on the disk surface. It is this information which allows the File Management System to recover a file by name. The sectoring is performed by a ROM in the disk drive upon command from the File Management System (FMS). This sectoring arrangement is a rather arbitrary decision. The hardware doesn't care where the sectors are put on the disk, as long as 18 exist on each track, and each sector has its preamble or header, identifying itself by number.

(Continued on page 81)

FOCUS

(Continued from page 79)

Drives purchased prior to fall 1981 wrote the sectors sequentially (i.e., 1, 2, 3, 4...). This method wastes time. For example, suppose sector 1 were to be read and then a small amount of processing were done prior to sector 2 being read. By the time the CPU were ready, sector 2 would have rotated past the read/write head, forcing the system to wait another revolution to read it. This is easily corrected by ordering or interleaving the sectors differently. This, in fact, is what the new drives do, and as a result save about 30% on file load time. Those of us with the old drives probably will be able to purchase the new ROMs as a retrofit by the time you read this.

When the formatting process is complete, 719 sectors have been initialized and allocated as follows:

Sector 1 - 3	boot record
Sector 4 - 359	user data
Sector 360	Volume Table of Contents (VTOC)
Sector 361 - 368	file directory
Sector 369 - 719	user data

Sector 360, the VTOC byte allocation is as follows:

Byte 0
directory type set to 0

Byte 1 - 2
maximum sector number (low, high byte)

Byte 3 - 4
number of sectors available (low, high byte)

Byte 5 - 9
unused

Byte 10 - 99
bit map, one bit per sector, bit set when sector in use

7						0
	1	2	3	5	6	7

Byte 10 of VTOC

8	9	10	

Byte 11 of VTOC

Sectors 361 through 368 are reserved for the file directory. Each directory entry is 16 bytes long, and allocated as follows:

Byte 0
flag byte $40 = in use, $60 = file locked, $80 = file deleted

Byte 1 - 2
total sectors in file (low, high byte)

Byte 3 - 4
starting sector of file (low, high byte)

Byte 5 - 12
file name (eight characters)

Byte 13 - 15
file name extension (three characters)

The last important piece of information is the layout of a sector in a user file.

Byte 0 - 124
user data/program

Byte 125
bits 2 - 7 file #
bits 0 - 1 are high bits of next byte

Byte 126
forward pointer (10 bits including bits 0 and 1 from byte 125)

Byte 127
bits 0 - 7 byte count used in sector

The file number is used to verify file integrity and contains the value of the directory position of that file. The forward pointer contains the disk sector number's 10-bit value of the next sector in the file. The pointer is equal to zero in the last sector of the file, and the byte count is the number of data bytes in the sector.

We'll continue our discussion of DOS II file structure and the floppy disk system again next month.

"Using Atari's Countdown Timer" *(Continued from page 40)*

When CDTM1 decrements to zero, the subroutine pointed to by CDTMA1 will be indirectly executed. If this subroutine is short in nature, its action will not appreciably slow down the currently executing program. If the countdown timer service routine also resets the countdown timer, then the user may implement their own background routines — all executing independently of the current "foreground" BASIC program. Thus the Atari Computer System may be set up to periodically execute an assembly language subroutine.

A potential problem exists in initiating a countdown timer with a value greater than 255 (one byte). Since 16-bit quantities cannot be manipulated by the 6502 processor directly, a VBLANK interrupt could occur while one timer byte is initialized and the other byte is not yet set. (If the initialization is done by BASIC POKEs, the chance is greater since BASIC is slower than machine language.) The programs in this article avoid this problem by limiting the countdown value to one byte, 255 or less. Other special cases are possible. Page 106 of the *Operating System User's Manual* outlines some general techniques to handle this problem.

To demonstrate the capability of CDTM1, listings 3 and 4 show a program that uses a background routine to change the basic voice #0. The timer completion routine uses the random number generator in the Atari operating system to select the next note frequency and timer value. An interesting point is that the countdown timers are controlled by the operating system, not the BASIC language cartridge. Because of this, the background timer processing continues after pressing the BREAK key. A system RESET is required to stop the timer processing.

With a few interfaces to the built-in joystick 6520 PIA ports and paddle inputs, background monitoring functions could be performed while a foreground BASIC program provides the man/machine interface, analyzes the data collected in the background, controls output interfaces based on the background monitoring, ...

MICRObits

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

TRS-80 Color Computer Game

Play "Brickaway" on any configuration of the Color Computer. A high-speed arcade game in 6809 machine language with realistic sound and brilliant colors. Only $7.00 postpaid. Requires joysticks, sold on cassette.

Britt Monk, CDP
P.O. Box 802
Elyria, OH 44036

6800/6809 Software

Includes compatible single-user, multi-user and network-operating systems, compilers, accounting and word processing packages. Free catalog.

Software Dynamics
2111 W. Crescent, Sta. G
Anaheim, CA 92801

Computer Hardware Cases

Protect your hardware from hard knocks. FIBERBILT stocks an extensive line of storage, carrying, and shipping cases for virtually every piece of hardware on the market. Call or write for details. Sales rep. inquiries invited.

FIBERBILT
601 West 26th Street
New York, NY 10001
(212) 675-5820

Apple-6809 Cross Assembler

Turn your S-C Assembler Version 4.0 into a cross-assembler for 6809 programs. Interfaces to Stellation MILL. Disk includes both source and object code for the self-installing conversion patches, sample 6809 program, instructions. Only $20.00 from:

S-C Software Corporation
P.O. Box 280300
Dallas, TX 75228
(214) 324-2050

Small Computer Program Index

Bi-monthly index to published computer programs. Over 2000 programs included in the first year. All subjects: education, games, business, mathematics, graphics, science, engineering, text handling and utilities. Subscription $35 per year. Further information from:

ALLM Books
21 Beechcroft Road
Bushey, Herts WD2 2JU
England

Accounting Software by SBCS

For the Apple II. General Ledger, Farm Ledger, Crop Ledger, Livestock Ledger, Accounts Receivable. Efficient, easy to use. Detailed documentation (credited towards purchase). FREE updates. Write for more information!

Small Business Computer Systems
4140 Greenwood
Lincoln, NE 68504
(402) 467-1878

PROM Programming Service

OSI users, put your Extended Monitor or other utilities in PROM. Your PROM (2708/2716/2732) erased and programmed for $5 each, if your program is on cassette (KC 300b). Blank PROMS available. Write for details.

Alpha III Electronics
285 N. Main Street
Crystal Lake, IL 60014

HTS Timestack KIM/AIM/OSI Products

Timestack Operating Systems implements programmable time-based controller. Software manual — $20. User manual — $5. Real-time clock hardware/software/interface manual — $10. General Expansion Board adds 8K of RAM, EPROM, and PORTS. Construction manual — $10. Specify system. Information — $1.

Hunter Technical Services
P.O. Box 359
Elm Grove, WI 53122

1981 Tax Programs — PET/CBM

1 — Federal 1040, 1040A, Sched. A, B, D, G. 2 — Federal/CA State and Sched. A & B. 3 - Schedules C, D, G, 2210, 2441, 5695. More. $150.00 each.

Don Henderson
9330 Bolsa Ave., Dept. M
Westminster, CA 92683

Librarian/Contour Mapping

Filing and keyword retrieval system for books, articles, excerpts, etc. Produces automatic bibliographies. Initial data base included. $69.95. Contour Mapping - high quality contouring system for printer, plotter or CRT. Data and grid transformations, trend surfaces, residuals. Data manager. Geological and geographical software for Apple and other systems.

Geosystems, Inc.
802 E. Grand River
Williamston, MI 48895
(517) 655-3726

OSI Software

Universe (plays like *Scramble* and *Cobra Copter*) is a super smooth machine code arcade game played by keyboard or joystick. $14.95. *Edit All* includes a full screen editor, scroll window output, full cursor control and more. $19.95. For C1P, C2-4 computers.

DMP Systems
319 Hampton Blvd.
Rochelle, NY 14612

A FORTHword

What is FORTH?

Most people have, at least, heard of FORTH. And most probably know that it uses Reverse Polish Notation and is a stack-oriented language. Many have heard that it is a "threaded" language, although some probably don't know what that means. It is a most peculiar language, but it is also one of the most powerful and the most flexible languages available.

Unlike Pascal and many other computer languages, FORTH was designed from the ground up by Charles H. Moore. Instead of starting with a grand scheme for the language, about ten years ago he started with the most basic aspects of the language, and gradually added features until it became the very powerful language it is today.

Moore arrived at the name because it was considered a fourth generation language. The system being used, however, could only take 5-letter names!

The basic element of FORTH is the "word." Even colons, commas, and semicolons are considered words. If it isn't a number, it must be a word! Writing a FORTH program is simply a matter of defining words, starting with the simplest and increasing in complexity, until the main program can be written in just a few lines with predefined words. As each word is encountered in the program, it is looked up in the dictionary and its definition executed. Every FORTH implementation starts out with a vocabulary of the most essential words. The programmer uses these to define his own. Frequently-used words may be permanently added to form a customized FORTH to suit particular needs. Even the standard words may be redefined, but only the most recent definition is used. It is even possible to define whole classes of words using the powerful > BUILDS... DOES > structure.

If you start at the highest level of operation of a program, you can trace the flow of control to the definitions of the words used, and from there to successively lower levels of definitions. Eventually the flow can be traced to words defined in the machine language of the host processor. These threads of control from the highest to the lowest level result in the term "threaded" being applied to FORTH.

There are actually two stacks: the main stack and the return stack. The main stack is used for nearly everything. As with a deck of playing cards, numbers or addresses may be added to the top of the stack or removed as needed. If you put three things on the stack, you can be sure you'll get those three items back, only in reverse order. Most operations act on the top two entries, removing them and replacing them with the result.

Reverse Polish Notation (RPN or Post-fix notation) is the method used to code FORTH programs. It makes use of the stack much more convenient and eliminates the need for parentheses. To add two numbers using the normal arithmetic notation you would enter: 5 + 3 = . In RPN, you would enter: 5 3 +. A more complicated expression, (7 + 5)*(3 − 1), would appear as 7 5 + 3 1 − * in RPN.

FORTH certainly qualifies as a "structured" language, in that the flow of control is usually very clear. There is no GOTO statement, and line numbers are used only by the editor to create a source program. Words in FORTH are actually procedures and functions in disguise. A number of powerful control structures are supported by FORTH, including DO...LOOP, IF...ELSE...THEN, and BEGIN...WHILE...REPEAT. Variables are used sparingly because the stack serves most storage needs.

Extensibility is another word frequently applied to FORTH, and FORTH literature is heavy with articles on various FORTH extension packages. If a word you need isn't in the vocabulary, all you need do is define it, and it is handled with equal priority. String handling is not a part of either fig-FORTH or FORTH-79. That is left to the user, since it is a pretty easy matter, and there are so many different ways to implement string functions. Floating point numbers are not supported in either standard, but again, words can be added for these functions.

Virtual memory is a feature of FORTH that receives little attention. A full implementation of FORTH involves one or more disk drives. The disks are formatted in 1024-byte blocks. These blocks are called screens because each block holds enough for 15 lines of 64 characters. These are handled through buffers, and if changes are made in a screen, the copy on the disk is automatically updated. It is also possible to write programs that occupy many screens (more than the actual memory of the computer) without any special provisions.

Advantages and Disadvantages

The advantages of FORTH are many. The FORTH system occupies little space in the computer, and it can be stripped down even further for particularly small systems or for dedicated applications. Programs can be very tightly coded so that they occupy little space, and the speed is invariably faster than the comparable BASIC programs. Most definitions can be tried out interactively before they are actually included in a program. With the built-in assembler, time-critical portions of the program can be written without switching to another assembler program. And finally, FORTH is one of the most portable languages available. This is due both to the structure of the language and to the willingness of all to set and adhere to standards.

There are also a few disadvantages. It takes quite a while to get used to RPN and stack manipulation, and FORTH code can be very difficult to read. Generally this is overcome by using very short word definitions and by using comments liberally. The disk has no directory — everything is accessible by screen number only — so it may take some effort to find a particular definition.

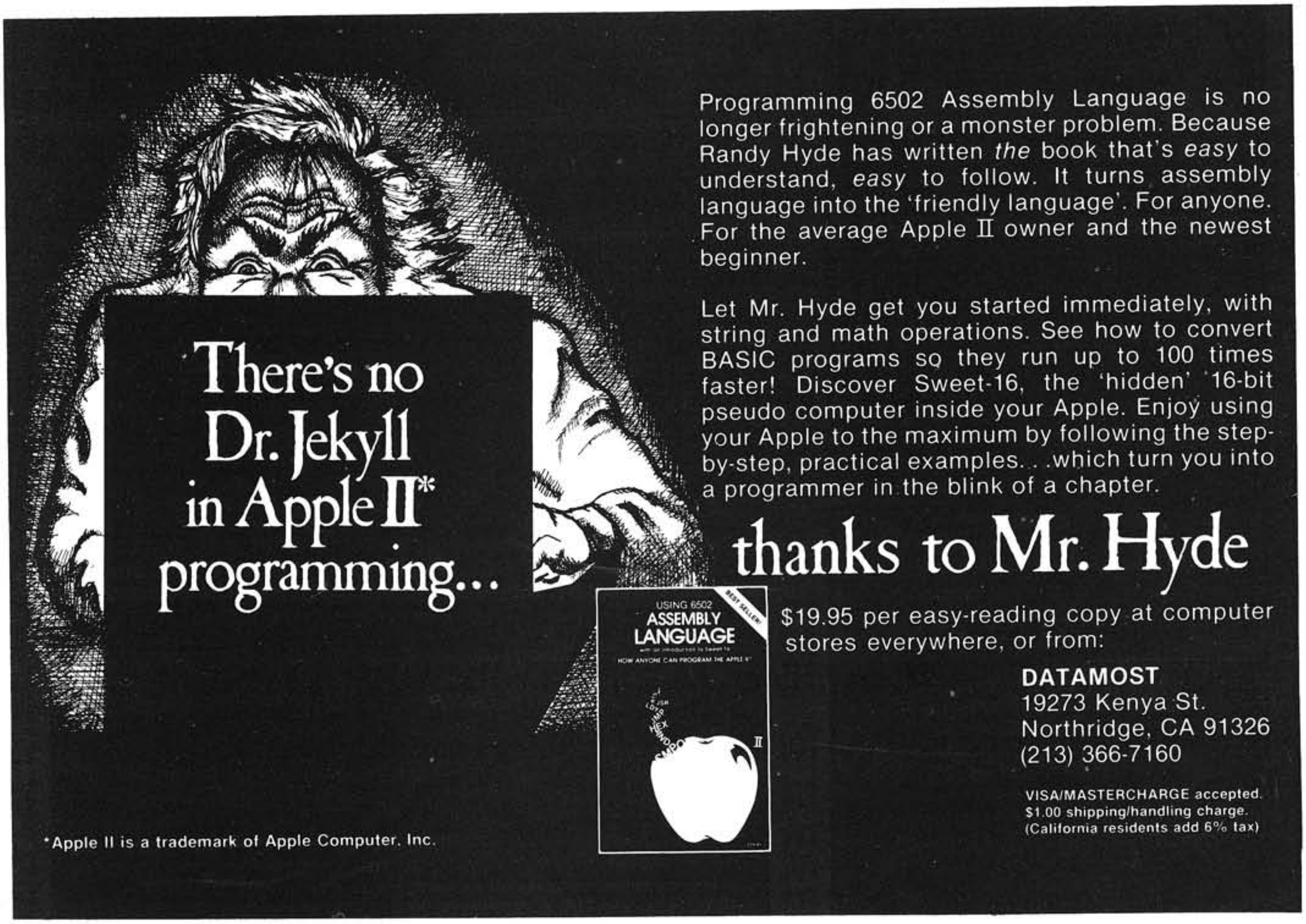
In this Issue

If you know nothing about FORTH, a good place to start is Nick Vrtis's "LIFE in FORTH and BASIC." The popular game of LIFE is presented so that the FORTH program can be compared directly to a BASIC program that follows the same flow. Mark Bernstein's "Stepper Motor Control" is an example of the "process-control" group of applications to which FORTH is particularly well-suited. The third article by Raymond Weisling demonstrates the extensibility of FORTH by adding a CASE statement, a FORTH decompiler, and Apple disk commands. We have listed in our "FORTH Resource List" (p. 108) only those companies which support 6502 and 6800 family implementations.

The best way to learn about FORTH, of course, is to use it, and with prices as low as $20, there is little to stand in the way.

Loren Wright

MICRO



There's no Dr. Jekyll in Apple II* programming...

Programming 6502 Assembly Language is no longer frightening or a monster problem. Because Randy Hyde has written *the* book that's *easy* to understand, *easy* to follow. It turns assembly language into the 'friendly language'. For anyone. For the average Apple II owner and the newest beginner.

Let Mr. Hyde get you started immediately, with string and math operations. See how to convert BASIC programs so they run up to 100 times faster! Discover Sweet-16, the 'hidden' 16-bit pseudo computer inside your Apple. Enjoy using your Apple to the maximum by following the step-by-step, practical examples. . .which turn you into a programmer in the blink of a chapter.

thanks to Mr. Hyde

$19.95 per easy-reading copy at computer stores everywhere, or from:

DATAMOST
19273 Kenya St.
Northridge, CA 91326
(213) 366-7160

VISA/MASTERCHARGE accepted.
$1.00 shipping/handling charge.
(California residents add 6% tax)

*Apple II is a trademark of Apple Computer, Inc.

WHY DO UNNECESSARY SURGERY ON YOUR APPLE?

Sooner or later, you're going to need a 16K memory-expansion for your Apple. When you do, we suggest you buy it on the card that doesn't require poking about on the motherboard — nor removing a RAM chip, installing a strap, etc.

The Ramex 16 RAM Board just plugs in. It's simple, reliable, and does its own memory refresh, with no additional connections.

Run Pascal, Fortran, FP, INT and other alternate languages, 56K CPM with a Z80 Softcard, increase usable memory for Visicalc by 16K. The possibilities are endless. Do it with the finest, closed-track engraved, epoxy sealed, 16K board available — the Ramex 16. And do it without unnecessary surgery on your Apple.

In spite of its quality, the Ramex 16 costs less than most other expansion boards — just $139.95. And it comes with a *one year* limited warranty, instead of the usual 90 days.

Get the Ramex 16 from your local dealer, or order direct. Visa and Mastercard holders call toll-free, 1-800-835-2246.

Look, ma, no straps!

OMEGA MICROWARE, INC.
222 SO. RIVERSIDE PLAZA
CHICAGO, IL 60606
312-648-1944

Apple and Applesoft are registered trademarks of Apple Computer, Inc. Pascal is a registered trademark of the Regents of the U. of C., San Diego. Visicalc is a registered trademark of Personal Software. CPM is a registered trademark of Digital Research, Inc. Z80 is a registered trademark of Zilos, Inc. Softcard is a registered trademark of Microsoft.

Using FORTH with the 6502

Here are three applications designed to help the newcomer to FORTH better use the power and flexibility inherent in this revolutionary language. These applications extend the power of FORTH by adding a CASE structure, a decompiler, and disk commands to the language.

Raymond Weisling
Jalan Citropuran No. 23
Surakarta, Jawa Tengah,
Indonesia

Until recently, FORTH has been available only to a small number of users. But in the past year the interest in this unusual language has grown tremendously, perhaps now boasting more proponents than other languages like C, APL, etc. It has enjoyed this growth because it offers an excellent compromise between the typical trade-offs of speed, memory efficiency, flexibility, cross-system transportability, and terseness, which other languages only partly achieve at the expense of other elements.

But FORTH's real beauty lies in its extensibility, something no other common computer language offers. (Human languages are all expandable, changing to fit the needs of its users. Imagine a human language that did not permit new words to be added, its users forced to describe new concepts with a fixed vocabulary long outdated.)

The ability to add new operators or commands, or even subroutine-like structures, to a language makes it appealing to all of us, who at one time or another have faced the frustration of some stubborn aspect of a rigid language. Indeed, programming in FORTH is always a process of expanding the language, as there is no real difference between the language itself and the "program."

This article describes several applications of FORTH using the Apple II and the Cap'n Software Version 1.7 FORTH. But since several other Apple II versions have been released by different vendors, and since FORTH boasts good transportability, the Apple II applications given could be easily implemented on other systems with little alteration. The applications shown here are for a CASE structure, an effective FORTH decompiler, and a partial implementation of Apple DOS. Readers who are unfamiliar with the basic principles of this language are directed to the bibliography given at the end of this article.

This article is aimed at those who already have FORTH running or who are seriously contemplating purchase (or even homebrewing — yes, it is possible) of FORTH. It offers useful tools to smooth the transition from other more conventional programming languages, and illustrates some of the powerful aspects of FORTH in a microcomputer environment.

A Case Structure

One of the current deficiencies of FORTH is the lack of a CASE structure. I felt that a CASE structure would improve programming efficiency and produce source code closer to human thinking habits than an alternative sequence of IF THEN statements, which have certainly been liberally sprinkled over the BASIC fare. When the FORTH Interest Group (FIG) ran a CASE contest, I selected one of the published entries for my use. A CASE structure should be able to use a value to make an n-way branch in the program flow, like the ON ... GOTO of BASIC, but without the restriction of contiguous input values. Listings for SCR # 80 and 81 show my selection, which is used in defining the remaining applications.

This CASE structure allows any single value to cause execution of branches in a "tree" if the condition for that branch is true. A value is placed on the stack before the BEGIN-CASES word, and it is compared with a value (or two for RANGE-CASE) that is placed on the stack before the CASE word. This usually is a literal value, but it could be any word yielding a value.

If the condition is true, the words between the CASE and END-CASE are executed. If more than one condition is true, only the first one is executed. The ELSE-CASE form is a default execution if none of the preceding cases were true. By logically arranging the order of these statements we can generate almost any kind of conditional structure (see SCR # 82 for an example of this kind of use).

SCR # 80 shows the actual run-time words used in this CASE structure, written in 6502 assembler code for the FORTH resident assembler. The original reference also lists a FORTH equivalent for this machine-dependent code, but it is slower and consumes more memory. As my other uses require maximum speed, I opted for the machine-code run-time words. SCR # 81 shows the compile-time words associated with the CASE structure. These are really expansions to the FORTH compiler, as they compile the high-level words CASE, END-CASE, BEGIN-CASES, ELSE-CASE, and END-CASES into correctly structured references to the machine-code run-time words. Some compiler error checking is done to assure proper pairing of the CASE words.

Credit for this CASE structure must go to R.D. Perry and the FORTH Interest Group (FIG). For more details on this structure or other submitted CASE structures, see FORTH DIMENSIONS, Vol. II, Number 3 (Sept/Oct 1980).

```
SCR # 80

( CASE STRUCTURE - PART 1    80A028110)

        ( CASE 6502-CODE RUN-TIME WORDS)
HEX

CODE N=BRANCH   INX. INX. FE ,X LDA,
 BOT CMP,    0=
 IF,  FF ,X LDA,  BOT 1+ CMP,    0=
   IF, INX,  INX,  ' 0BRANCH 8 + JMP,
   THEN,
 THEN,  ' BRANCH JMP,
END-CODE

CODE NRANGE=BRANCH   INX, INX, INX, INX,
 SEC, FC ,X LDA, BOT SBC, FD ,X LDA,
 BOT 1+ SBC,     0< NOT
 IF, SEC, BOT LDA, FE ,X SBC,
   BOT 1+ LDA,  FF ,X SBC,    0< NOT
   IF, INX, INX,  ' 0BRANCH 8 + JMP,
   THEN,
 THEN,   ' BRANCH JMP,
END-CODE

  -->
```

SCR # 80: **CASE run-time words defined in 6502 assembly language for the FORTH resident assembler.**

```
SCR # 81

( CASE - PART 2                    81A018110)

: BEGIN-CASES ?COMP 0 4 ;      IMMEDIATE

: CASE ?COMP 4 ?PAIRS COMPILE N=BRANCH
  HERE 0 , 5 ;                  IMMEDIATE

: RANGE-CASE ?COMP 4 ?PAIRS COMPILE
  NRANGE=BRANCH HERE 0 , 5 ;  IMMEDIATE

: ELSE-CASE ?COMP 4 ?PAIRS COMPILE DROP
  0 5 ;                         IMMEDIATE

: END-CASE ?COMP 5 ?PAIRS COMPILE BRANCH
  DUP IF HERE 2+ OVER - SWAP ! ELSE DROP
  THEN HERE SWAP , 4 ;          IMMEDIATE

: END-CASES ?COMP 4 ?PAIRS DUP 0= 0= 1
  ?PAIRS COMPILE DROP BEGIN DUP WHILE
  DUP @ SWAP HERE OVER - SWAP ! REPEAT
  DROP ;                        IMMEDIATE

;S SEE FORTH DIMENSIONS, VOL 2, NO 3,
   SEPT/OCT 1980; R.D. PERRY, P. 78
```

SCR # 81: **CASE compiler extensions compile references to the machine code run-time words into new definitions.**

A Decompiler for FORTH

With almost any program we encounter there is always some degree of curiosity about its inner workings. FORTH is no exception, and since we are charged with expanding the language (i.e., programming in it), we should do it in style. One excellent way to learn style is to study other programmers' examples. As FORTH is a fully structured language from end to end, virtually any part of it can be viewed and understood, given a suitable inspection tool. Even the core or nucleus, written in assembler code for the host processor, is in neat little blocks with rather uniform protocol "hooks."

I wanted to know more about these inner workings, so I developed this decompiler. But it has more than entertainment value. Building the DOS vocabulary required some digging at the FORTH disk processing words. Here, this tool paid for its own development time. In fact, I had originally planned to keep this decompiler on disk, compiling it into the FORTH dictionary when needed. But it quickly became evident that it should be permanently added to my FORTH dictionary so that it is present after booting the language.

The decompiler itself, shown on SCR # 83 to 85, is invoked by the word LK:, followed by the word to be decompiled. If the word is a colon definition, the individual words making up the definition are printed. Literals are printed in the current number base. If the word being decompiled is a machine code definition, the Apple II monitor disassembler is called to display a fixed number of lines in hexadecimal notation. Other words, like constants and variables, and the run-time part of new defining words using the < BUILDS DOES > structure, are properly identified.

The word LK: first makes the parameter field address of the next word in the input stream, the word we are to decompile. The code field is created from the parameter field, and is tested against a list of literal addresses for the action appropriate to the type of word being decompiled. If this address matches, the branch is taken and other words are executed or messages are output. The default branch is for code definitions, as its code field always contains a value two greater than its code field address. The word DASM invokes the monitor disassembler. (On non-Apple systems you could write a FORTH disassembler, or simply print a message like "CODE DEFINITION".) Note that these are specific addresses for my vendor's version and other address values have to be found experimentally for other implementations. The toolbox we are filling here includes the word CFQ to help derive these addresses. CFQ has no other association with the decompiler, and need not be included once its work is done.

The word LKD handles the case of the run-time DOES > definition. The result is the DOES > part of the parent-defining word, as well as the address and contents of the following memory address in the member word we are decompiling. Since it is impossible to know the function of a < BUILDS DOES > definition, further decompiling is not done. These results may be used for more investigation if desired. LKU handles the user-variable type of word. The literal 830, which points to the user variable area, may not be the same for other versions; this terminal sequence will yield the correct value:

```
HEX 0 USER JUNK JUNK .
FORGET JUNK DECIMAL
(return)
```

The words LKC and LKTST handle the relatively complex task of decompiling colon definitions, the bulk of the FORTH language. LKC consists of display formatting words, and operations necessary to step through the word until the end is reached. Hitting any key will abort the processing loop. LKC does the name printing using the word ID. . LKTST tests each code field address in the definition to see if the next one or two bytes contain an explicit literal, or a literal for one of the branching control structures compiled from the IF-ELSE-THEN or BEGIN-UNTIL family of branching words. Our recent additions, the run-time words for our CASE structure, are also included. Note that the CASE literals for these cannot be predicted. Therefore, the sequence

```
[ ' N = BRANCH CFA ] LITERAL
```

is used to temporarily exit the compiler for purposes of computing the code field address of the word following the tick ('). This address is placed on the stack and is used by LITERAL to make

```
SCR # 82

( CASE EXAMPLE                               82A018127)

: TEST                                         ( N ---)
 BEGIN-CASES
  0       CASE ." ZERO"                      END-CASE
  1       CASE ." ONE"                       END-CASE
  2       CASE ." TWO"                       END-CASE
  3       CASE ." THREE"                     END-CASE
 -2       CASE ." MINUS TWO"                 END-CASE
 BASE @ CASE ." SAME AS BASE"                END-CASE
 4 12 RANGE-CASE ." 4 TO 12"                 END-CASE
 0< 1     CASE ." NEGATIVE"                  END-CASE
   ELSE-CASE ." TOO BIG"                     END-CASE
 END-CASES CR ;


: TESTLOOP CR 17 -5
 DO
  I  8 .R 3 SPACES
  I TEST
 LOOP ;
```

SCR # 82: **Example of the CASE construct used for selecting terminal messages. Note that anything which places a value onto the stack may precede the word CASE ; here we have literal values, a fetch from the system variable for number base, and a logical operator to detect any negative number. The word TESTLOOP places this test into a loop for illustration.**

Listing 1: **An example of the decompiler output for three different colon definitions.**

```
HEX LK: DASM
LIT 3A ! LIT FE61 CALL CR ;S

 OK
LK: LKTST
LIT 89E N=BRANCH 10 DUP @ U. 2+ 0
BRANCH 9A LIT 8D8 N=BRANCH 10 DUP
C@ . 1+ 0 BRANCH 84 LIT 953
N=BRANCH 8 LKLIT BRANCH 76 LIT 983
N=BRANCH 8 LKLIT BRANCH 68 LIT 932
N=BRANCH 8 LKLIT BRANCH 5A LIT 913
N=BRANCH 8 LKLIT BRANCH 4C LIT 1474
N=BRANCH E DUP C@ LKEMIT 0 BRANCH 38
LIT C37 N=BRANCH A DROP 1 BRANCH 28
LIT 3DB6 N=BRANCH 8 LKLIT BRANCH 1A
LIT 3DDE N=BRANCH 8 LKLIT BRANCH C
DROP 0 BRANCH 4 DROP ;S

 OK
LK: MIN
OVER OVER > 0BRANCH 4 SWAP DROP ;S

 OK
DECIMAL  OK
```

(obviously) a literal. The other values are fixed for my system (but may be different for others), and can be handled similarly, using the same syntax form. Otherwise the CFQ tool can be used as before.

If one of the CASE branch tests is valid, the literal associated is printed and the address pointer is incremented the appropriate amount (+ 2 for all except the byte literal CLIT). Then a 0 or 1 flag is set to show if the end of the definition has been reached (only ;S signals an end). The case of a text string literal is handled by the branch with LKEMIT . The ELSE-CASE branch receives the most traffic, as most other words used in the definition are trapped

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.**

FORTH-79

Ver. 2 For your APPLE II/II+

The complete professional software system, that meets ALL provisions of the FORTH—79 Standard (adopted Oct. 1980). Compare the many advanced features of FORTH—79 with the FORTH you are now using, or plan to buy!

FEATURES	**OURS**	**OTHERS**
79-Standard system gives source portability.	YES	______
Professionally written tutorial & user manual	200 PG.	______
Screen editor with user-definable controls.	YES	______
Macro-assembler with local labels.	YES	______
Virtual memory.	YES	______
Both 13 & 16-sector format.	YES	______
Multiple disk drives.	YES	______
Double-number Standard & String extensions.	YES	______
Upper/lower case keyboard input.	YES	______
LO-Res graphics.	YES	______
Z-80 CP/M Ver. 2.x & Northstar also available	YES	______
Affordable!	$99.95	______
Low cost enhancement options:		
Floating-point mathematics.	YES	______
HI-Res turtle-graphics.	YES	______
Data Base Management System.	YES	______
Apple DOS/FORTH—79 file transfer.	YES	______
Modem file transfer utilities.	YES	______

FORTH—79 V.2 (requires 48K & 1 disk drive)	$ 99.95
ENHANCEMENT PACKAGES FOR V.2:	
Option 1 - floating point & HI-Res turtle-graphics.	$ 49.95
Option 2 - DBMS, DOS/FORTH conv. & modem.	$ 49.95
COMBINATION PACKAGE	$179.95
(Postage Paid: CA res. add 6% tax: COD accepted)	

MicroMotion
12077 Wilshire Blvd. # 506
L.A., CA 90025 (213) 821-4340
Specify APPLE, CP/M or Northstar
Dealer inquiries invited.

here. The word LKLIT only saves us from filling LKTST with redundant code. I chose to print out the branching literals as signed numbers, but pure literals are printed as unsigned numbers. You may prefer all of them to be printed as signed numbers. If so, the (LIT) line could also use the LKLIT word to replace the longer DUP @ U. 2+ 0 sequence.

You can see that the CASE structure is very flexible and quite valuable. The alternative, IF THEN structures, would look less obvious even though they might compile into the same resultant code. This is what languages are all about — making the job of the programmer (or writer, poet, composer...) easier by offering useful structures for judicious selection.

The decompiler is easy and straightforward to use. Note that some defined words compile into something quite different from the source, especially the branching and looping structures. The most familiar is our CASE structure, which compiles into machine code primitives. Therefore, the decompiler output will not look like the original code, but understanding this simple transformation will help shape a better understanding of how FORTH performs its own work. I have included a sample (listing 1) of some words decompiled by LK: for purposes of illustration. The disassembler word DASM may, of course, be used individually, and it saves having to enter the monitor. Just be sure that the starting address is on the stack.

Apple DOS 3.2 in FORTH

Another problem I encountered when using FORTH came from the fact that FORTH uses its own simple (but fast and effective) set of disk operations, quite incompatible with the Apple DOS. I prefer these to the standard DOS, as they are much more flexible. But I still had a few BASIC programs that make binary files, and I needed to use these in a FORTH application. I also have a PROM programmer running in FORTH, and my conventional assembler only makes DOS files. So clearly there was a need for an effective "hybrid critter" to be created.

Not all of the DOS functions were needed, but the groundwork is done if anyone wants to expand my DOS operators. I only needed a CATALOG report and the BLOAD command. Loading other types of files is not much different (but maybe pointless for Applesoft or Integer BASIC files). Writing

```
SCR # 83

( DECOMPILER - PART 1          83C038127)
HEX
: LKEMIT    ( ADR CHARCOUNT --- NXTADR)
 1+ 2DUP            ( PREPARE FOR OUTPUT)
 TYPE   22 EMIT SPACE       ( OF STRING)
 +  ;       ( POINT TO NEXT WORD IN DEF)

: LKLIT  DUP @ . 2+ 0 ;

: LKTST  BEGIN-CASES     ( CFA LIT TEST)
89E CASE DUP @ U. 2+ 0 END-CASE  ( LIT)
8D8 CASE DUP C@ . 1+ 0 END-CASE ( CLIT)
953 CASE LKLIT         END-CASE ( LOOP)
983 CASE LKLIT         END-CASE ( +LOO)
932 CASE LKLIT         END-CASE  ( 0BR)
913 CASE LKLIT         END-CASE   ( BR)
1474 CASE DUP C@ LKEMIT 0 END-CASE
C37  CASE DROP       1 END-CASE  ( ;S)
 [ ' N=BRANCH CFA ]         LITERAL
     CASE LKLIT        END-CASE ( N=BR)
 [ ' NRANGE=BRANCH CFA ]   LITERAL
     CASE LKLIT        END-CASE ( R=BR)
ELSE-CASE 0 END-CASE        ( OTHERS)
END-CASES ;             -->

SCR # 84

( DECOMPILER - PART 2          84C028127)

       ( APPLE II MONITOR DISASSEMBLER)
: DASM                      ( ADR ---)
 3A ! FE61 CALL CR ;

         ( DECOMPILE COLON DEFINITION)
: LKC
  BEGIN  DUP DUP >R  ( DO EACH PF ITEM)
   @   2+ NFA DUP @ 1F AND 6 +
   OUT @  +  C/L @   >
   IF      CR  0  OUT !
   THEN    ID. 2+ R>  @
   LKTST ?TERMINAL OR  ( SPECIAL FIELD)
  UNTIL CR ;


: LKU    C@ 830 + @ . ; ( PRINT USER V)

: LKD DUP                   ( PFA ---)
 @ LKC ." PFA+2 = " 2+ DUP U. 5 SPACES
 @     ." @ = "   U. ;

 -->

SCR # 85

( DECOMPILER - PART 3          85C028126)

: LK:  <<COMPILE>> '  ( WORD IN-STREAM)
80 OUT !           ( FOR DISPLAY FORMAT)
DUP CFA @
BEGIN-CASES
  EBE CASE                  LKC   END-CASE
  F05 CASE ." CONSTANT " @ .   END-CASE
  F21 CASE ." VARIABLE " @ .   END-CASE
  FA2 CASE ." USER-VAR " LKU   END-CASE
 13DA CASE ." DO-DOES> " LKD   END-CASE
 ELSE-CASE                 DASM  END-CASE
END-CASES CR ;

 ( CFQ PRINTS THE CFA & CF OF ANY WORD)

: CFQ                 ( WORD IN-STREAM)
 -FIND 0= 0 ?ERROR DROP
 [COMPILE]  LITERAL
 CFA   DUP ." CFA " U.
 5 SPACES @ ." CF " U. ;

DECIMAL
```

SCR # 83 to 85: **A FORTH decompiler handles colon definitions, constants, variables, machine code, and defining words.**

```
SCR # 86

( DOS 3.2 VOCABULARY              86A028126)

VOCABULARY DOS            DOS DEFINITIONS

HEX

      ( &&IO IS SECTOR READ-WRITE WORD)
: &&IO    ( ADR CSECT F --- ) ( 1=R 0=W)
 MINUS 2+ B7F4 C!          ( THE R/W FLAG)
 SWAP B7F0 !        ( THE ADDRESS STUFFED)
 0D /MOD B7EC C! B7ED C!  ( SECT/TRACK)
 &RWTS  DROP ;    ( DROP ERROR RETURNED)

DECIMAL

: BYTE-TABLE <BUILDS ALLOT DOES> + ;

256 BYTE-TABLE SECTOR-BUF

: RSECT                    ( CSECT ---)
 0 SECTOR-BUF SWAP  ( MOVE 1 SECTOR TO)
 1 &&IO ;                ( SECTOR BUFFER)

 -->
```

***SCR # 86:* The disk I/O words and a sector buffer are the foundation for an Apple II DOS utility vocabulary.**

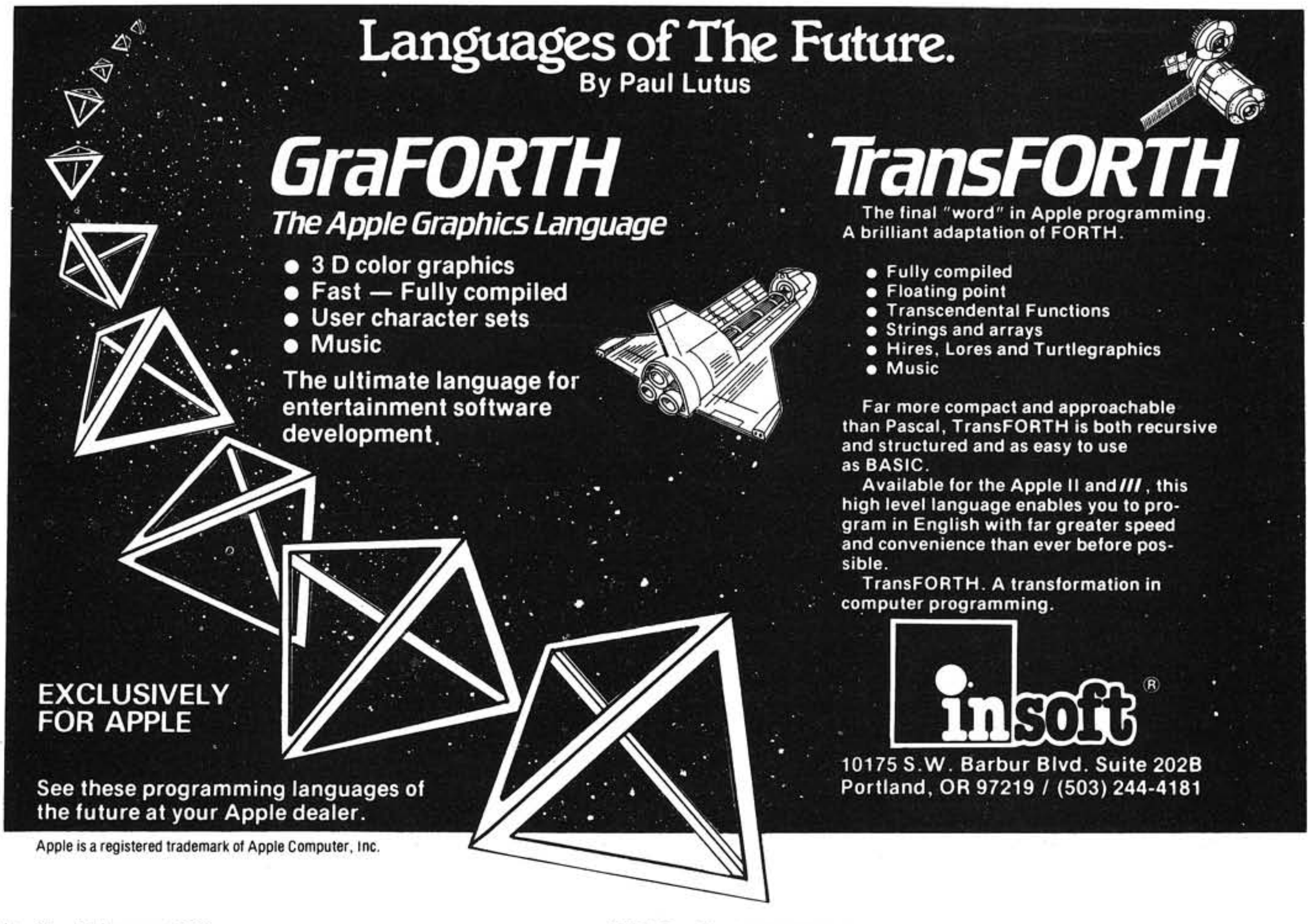
```
SCR # 87

( DOS - PART 2                    87A028126)

: ?DOS 221 RSECT        ( TEST IF DOS3.2)
   1 SECTOR-BUF C@ 17 =
   2 SECTOR-BUF C@ 12 = AND 0=
  IF ." NOT DOS 3.2 DISK" 7 EMIT QUIT
  THEN ;

: DOS-FTYPE DUP 8 >          ( TYPE ---)
 IF 42  EMIT 15 AND     ( IF ITS LOCKED)
 ELSE SPACE
 THEN                  ( PRINT FILETYPE)
 BEGIN-CASES
  0 CASE 84 END-CASE     ( FILETYPE-TO-)
  1 CASE 73 END-CASE       ( ASCII CASE)
  2 CASE 65 END-CASE       ( CONVERSION)
  4 CASE 66 END-CASE
 END-CASES EMIT ;      ( PRINT FILETYPE)

: TSLINK           ( BUFFPTR --- CSECT )
 SECTOR-BUF DUP C@ 13 * ( BUF TRACK*13)
 SWAP 1+ C@ + ;             ( ADD SECTOR)

 -->
```

***SCR # 87 to 88:* The DOS words for the CATALOG command include a test to be sure that a DOS disk is in the drive.**

DOS files is more complex, as there are a lot of housekeeping details associated with sector allocation.

The DOS words are all defined in a separate vocabulary, so that when they are not needed they cannot be accidentally invoked. The compiler source is shown in SCR # 86 to 91. Let's take a look at some of the details we need to take care of.

The FORTH system I have employs a set of disk-handling words. Among these is a primative called &RWTS , which handles the read and write operations to defined tracks and sectors. A FORTH definition, &I/O , sets up the operations and invokes &RWTS . But &RWTS returns an error message if we try to read *some* DOS sectors, because a FORTH initialized disk is expected. We just make our own version of &I/O , called &&IO , which discards



this error message. (See SCR # 86.) Two messages are known to be returned by &RWTS : one for the above-mentioned error, and another for a write to a write-protected disk. Since my words do not write, I simply ignore the returned messages. &&IO also allows us to use only one number for the track and sector, which is the "contiguous sector" number, shown as "csect" in the comments. It is simply the track number multiplied by the number of sectors per track plus the sector number (13T + S for DOS 3.2), so it ranges from 0 to 454. This is easier than trying to untangle two similar numbers.

Next we see a byte table-defining word used to make a sector buffer, called SECTOR-BUF , which returns an absolute memory address when the relative byte number is given. The word RSECT will load this buffer from the supplied sector number (csect).

SCR # 87 has a disk-type test word, ?DOS , which checks the DOS VTOC sector for some DOS-related information, so that accidental use on a FORTH disk will not send you off on a wild sector hunt, possibly without an ending other than hitting Reset. Other error-checking words are included, and all thread back to the word QUIT, which aborts the DOS command. Also on this screen is a test and report by DOS-FTYPE , of DOS file types, including whether or not the file is locked. The word TSLINK takes a pointer into the sector buffer where a track/sector link resides, and returns the contiguous sector number of that link. On SCR # 88 is the director (or catalog) scan word SCAT for the current directory sector in the sector buffer, which checks to be sure the entry isn't empty or deleted. If it is active, SCAT reports the file type, sector length, and filename.

The word we will use to see the full catalog is the same as in DOS, the familiar CATALOG . This word loads the first directory sector into the sector buffer, uses SCAT to make a report, and then links to the next directory sector, repeating until the link is a null. The volume number is not displayed or tested.

The definitions on SCR # 89 and 90 do the directory searching for a user-supplied filename, and can be used for expansion where similar directory searching is required. SCNSCT searches the current directory sector in the buffer for a match to what FILENAME retrieved from the input stream. The word -TEXT is a non-standard

```
SCR # 88

( DOS - PART 3                   88A038127)


     ( ONE SECTOR OF CATALOG DISPLAYED)
: SCAT 222 11                        ( ---)
 DO  I SECTOR-BUF C@  ( GET FIRST BYTE)
  DUP 255 = SWAP 0= OR 0=     ( ACTIVE?)
  IF CR I 2+ SECTOR-BUF C@ DOS-FTYPE
     I 33 + SECTOR-BUF C@ 4 .R 2 SPACES
     I  3 + SECTOR-BUF 30 TYPE
  THEN  ?WAIT 35      ( PAUSE? THEN LOOP)
 +LOOP ;                ( 35 BYTES/ENTRY)


             ( DISPLAY COMPLETE CATALOG)
: CATALOG ?DOS  ( TEST IF DOS 3.2 DISK)
 233              ( FIRST SECTOR OF DIR)
 BEGIN
  RSECT SCAT          ( GET & DISPLAY CAT)
  1 TSLINK -DUP 0= ( MAKE LINK & CHECK)
 UNTIL CR ;         ( REPEAT IF NOT NULL)

 -->

SCR # 89

( DOS - PART 4                   89A028126)

      ( SCAN BUFFER FOR FILENAME MATCH)
: SCNSCT 222 11 ( KEYADR CNT --- N1 N2)
 DO
  2DUP
  I 3 + SECTOR-BUF ( POINT TO FILENAME)
  -TEXT 0=             ( COMPARE STRINGS)
  IF  2DROP I -1 LEAVE               ( HIT)
  THEN  35             ( MISS, TRY AGAIN)
 +LOOP ;  ( IF HIT:  N1=T/S PTR N2= -1)
          ( IF MISS: N1=KEYADR  N2=CNT)

      ( GET FILENAME FROM INPUT STREAM)
: FILENAME                ( --- ADR COUNT)
 13 WORD HERE 1+  ( GET FILENAME & ADR)
 HERE C@               ( AND ITS LENGTH)
 OVER + 30 BLANKS  ( TRAIL WITH BLANKS)
 30 DUP 1+ 0
 DO I HERE + DUP C@   ( FORCE HIGH BIT)
  128 OR SWAP C!                  ( HIGH)
 LOOP ;            ( COUNT ALWAYS = 30 )

 -->

SCR # 90

( DOS - PART 5                   90A048127)

: NOFERR CR ." FILE NOT FOUND"
 7 EMIT SP! QUIT ;

: FTERR  CR ." FILE TYPE MISMATCH"
 7 EMIT SP! QUIT ;

      ( SCAN FULL CATALOG FOR FILENAME)

: CATHUNT                     ( --- BUFFPTR)
 FILENAME               ( INPUT FILENAME)
 ?DOS 233      ( CHK DISK; 1ST CAT SECT)
 BEGIN   RSECT SCNSCT  ( RD, SCAN SECT)
  DUP 0< >R              ( TEST HIT/MISS)
  1 TSLINK DUP 0=    ( GET LINK & CHECK)
  R> OR            ( IF NULL LINK OR HIT)
 UNTIL          ( BE DONE, ELSE REPEAT)
 DROP 0< 0=           ( WHAT IS RESULT?)
 IF NOFERR
 THEN ;                   ( EXIT IF FOUND)
 ( POINTING TO CATALOG ENTRY IN BUFFER)
 -->
```

***SCR # 88 to 90:* These words locate a supplied filename in the directory and then point to the track/sector list link.**

```
SCR # 91

( DOS - PART 6                      91A048127)

: BLOAD                      ( LOADADR --- )
 >R ?STACK R>      ( STACK HAS ANYTHING?)
 CATHUNT DUP SECTOR-BUF ( SCAN CATALOG)
 2+ C@ 7 AND 4 = 0=      ( TEST FILETYPE)
 IF FTERR                 ( NOT B COMPLAIN)
 THEN TSLINK RSECT         ( GET T/S LIST)
 PAD 256 12  ( LOOP TO GET EACH SECTOR)
 DO DUP      ( SAVE SECTORS IN PAD AREA)
  I TSLINK DUP 0=  ( IS IT A NULL T/S?)
  IF DROP 2DROP LEAVE      ( REACHED END)
  ELSE 1 &&IO 256 +      ( DATA INTO PAD)
  THEN 2   ( LOOP TO CHECK NEXT SECTOR)
 +LOOP                       ( UNTIL DONE)
 PAD 2+ @          ( GET TRUE FILE LENGTH)
 PAD 4 + ROT ROT  ( SETUP TO MOVE DATA)
 CMOVE ;                ( MOVE, THEN EXIT)
          ( BE SURE LOADADR IS PRESENT)

                ( END OF DOS VOCABULARY)

FORTH DEFINITIONS     ( RESTORE FORTH AS)
                   ( CURRENT AND CONTEXT)
```

SCR # 91: **The BLOAD operation is done with one definition. Similar load operations could be created for other filetypes using the previously defined words like CATHUNT.**

```
SCR # 92

( EXTRA WORDS FOR DOS OR ??  92A018127)
                        ( STRING COMPARE)
: -TEXT          ( ADR1 COUNT ADR2 --- F)
 0 ROT 0         ( PREPARE DO PARAMETERS)
 DO  DROP        ( DROP PRIOR RESULT = 0)
  OVER I + C@ OVER I + C@ - ( $COMPARE)
  IF   LEAVE       ( IF <> CAN QUIT LOOP)
  THEN                 ( WITH DIFFERENCE)
 LOOP                 ( IF 0 TRY NEXT CHAR)
 DUP           ( TEST RESULT & MAKE +/-1)
 IF    DUP ABS /       ( FROM DIFFERENCE)
 THEN  ROT ROT 2DROP ;       ( DROP ADRS)

HEX                ( IF KEY HIT, WAIT FOR)
 CODE ?WAIT                 ( ANOTHER ONE,)
   C000 BIT, 0<      ( ALLOWING A PAUSE)
   IF,    C010 BIT,
     BEGIN,  C000 BIT,   0<
     UNTIL,
   THEN,  C010 BIT,
   NEXT JMP,
 END-CODE      ( ?WAIT FOR APPLE II ONLY)
DECIMAL
```

SCR # 92: **A string-comparing word and a start-stop display word for DOS or any other application. Some systems may already have one or both of these.**

```
SCR # 93

( NON-DESTRUCT STACK DISPLAY 93A018127)

DECIMAL

: S??                        ( ?? --- ??)
 SP@ DUP 189 < ( GET CURRENT STACK ADR)
 IF   CR           ( IF NOT EMPTY, PRINT)
   2 - 187          ( ONE ITEM AT A TIME)
   DO  I @ 8 .R                ( IN LOOP)
     -2                       ( NEXT ADR)
   +LOOP
 ELSE DROP                  ( EMPTY STACK)
 THEN CR ;           ( EXIT WITH CR FIRST)
```

SCR # 93: **A debugging tool permits display of the stack contents without altering the stack; the most recent values appear on the right.**

6502

FORTH

- performs **FLOATING POINT MATH**
- adheres to the FORTH-79 STANDARD
- contains a 6502 MACRO ASSEMBLER
- handles STRINGS much like BASIC
- includes VIRTUAL MEMORY MANAGER
- is ROMABLE
- costs under $100
- is available for KIM-1, AIM 65 and SYM-1
- manual has extensive FORTH TUTORIAL

For more information and an ordering form

send a S.A.S.E. to: Eric C. Rehnke
1067 Jadestone Lane
Corona, CA 91720
(714) 371-4548

TRS-80 **COLOR COMPUTER SOFTWARE**

FORTH FOR THE TRS-80 COLOR COMPUTER DISK SYSTEM

Trying to get control of your Color Computer?? Tired of translating HEX to decimal?? Tired of remembering where the VDG and SAM are and how to program them?? Want to write machine language code with assembly language mnemonics instead of POKES??
Want to write programs in half the time?? Want to write lots of small pieces of code that you can put together in seconds to do BIG JOBS??? Want a language that is at least 5 to 10 times faster than BASIC??? Want to learn everything there is to know about FORTH, with the best manual on the market, including lots of examples of FORTH applications, and detailed explanations of how everything works??

CC FORTH IS THE ANSWER!!

Includes Editor, 6809 Assembler, String Functions, Disk Data File Operations and Much Much More! **$99.95**

THE COLOR TOOLKIT

Utility and Diagnostic Disk Programs by Dick Bartholomew

The COLOR Toolkit is a set of Disk diagnostics and Disk utilities for the TRS-80 Color Computer Disk System. Dick Bartholomew, well known for his utility programs for FLEX systems, has created a package of invaluable tools for the serious programmer. These include: Reading FLEX disks, Writing FLEX disks, Repairing Radio Shack disks, Extended directory, and many, many more.

PRICE ONLY $49.95 on RS disk.

U.S.A. ADD $2.00 FOR STANDARD UPS SHIPPING & HANDLING
FOREIGN ORDERS ADD 10% SURFACE, 20% AIRMAIL

FRANK HOGG LABORATORY
130 MIDTOWN PLAZA, SYRACUSE, NY 13210 (315) 474-7856

string-comparison word in the Cap'n Software release. If you don't have a similar string-compare word, you can use my high-level definition shown in SCR # 92, which is a bit slower than the machine-code one in my system. It takes the address of string A, the character count, and the string B address, and returns a single flag which is − 1 if string A is less, + 1 if greater, and 0 if equal. This screen also shows the ?WAIT word that allows us to stop-start long catalogs. If you compile any of these words, you might want to place them in the main FORTH vocabulary so they are accessible to other applications you add later.

These words are used by CATHUNT to actually search the entire directory for a filename match. The result returned is the sector buffer pointer to the matched directory entry left on the stack. This pointer actually points to the track/sector list link. The next higher word definition (BLOAD in this case) merely has to load that sector into the sector buffer and start to process this list. If the search is unsuccessful, an error message-quit sequence aborts the whole operation, so there is no return from a CATHUNT without our desired prey.

What's left? SCR # 91 has our user-invoked word BLOAD . It looks blacker than other screens, because of heavy commenting. It does a simple check for at least one stack entry of any value so an empty stack (no load address supplied) will not cause problems with the final positioning of the data by the CMOVE word. An additional file type test is made to be sure that the found directory entry is indeed a binary file. Then the track/sector list is loaded into the buffer RSECT . The actual file transfer is to the free area above the FORTH dictionary, beginning at PAD . This extra buffering is necessary since the first sector has four non-data bytes specifying load address and file length, and since this true length may not be a multiple of one sector (256 bytes). When the whole file has been read, the file length data is placed on the stack along with the address of our PAD and the originally-supplied load address. CMOVE moves it into place.

You or your program must take care in supplying this address, as CMOVE is happy to move this data anywhere in memory, including over FORTH itself. For this reason, the DOS default load of a binary file to its original load address was not implemented. The use of the BLOAD word (or CATHUNT and any expansion words using it) involves specifying a filename with or without spaces, followed by a carriage return, 30 characters maximum (the DOS length limit). For example:

```
HEX 6000 BLOAD JUNKFILE
  XYZ (return)
```

Doing a VLIST after this DOS system is compiled into the FORTH dictionary will reveal only one word: DOS, itself. This is the name of a new vocabulary, and to use it you must type DOS first. How are we doing on memory? The DOS vocabulary as shown here consumes a tidy 1154 bytes including the sector buffer. We can't claim to have even half of the real DOS capability, but the foundation is laid. However, the Apple DOS 3.2 version uses over 9K of memory. My development time was only at eight to twelve hours.

Bonus Offering

As I am debugging a FORTH program, I often want to see the stack contents after a certain word is manually invoked from the keyboard. The . (dot) print method is not suitable, as it both empties the stack and displays the contents in reverse order to the convention of FORTH documentation, where the top of the stack is on the right. I have written a very short non-destructive stack display word, S?? , for my system. The literals are for the Cap'n Software version.

The stack address for different systems can be found by typing SP! SP@ . and noting the result. The 6502 fig-FORTH model uses a zero page stack that grows downward. Other releases for the 6502 probably are based on this model with perhaps a different starting point. Simply alter the literals in the listing in SCR # 93 and improve your debugging efficiency.

Bibliography

1. *Using FORTH,* by FORTH Inc., Hermosa Beach, CA (also available from FIG).
2. *BYTE,* August 1980, a special FORTH issue.
3. *FORTH DIMENSIONS,* the official publication of the FORTH Interest Group (FIG), P.O. Box 1105, San Carlos, CA 94070. *Note:* All publications from FIG are in the public domain.
4. fig-FORTH Installation Manual and assembly language source listing for 6502 and Apple II.
5. *Threaded Interpretive Languages,* R.G. Loeliger (1981), BYTE Books, Peterborough, NH.
6. *Systems Guide to fig-FORTH,* C.H. Ting, available from the author at 156 14th Ave., San Mateo, CA 94402.

Raymond Weisling has worked with digital-analog music synthesis systems and industrial process control. Current activities include programming in FORTH, building kinetic sculptures, teaching digital circuit design and English, and enjoying immersion in the culture of Java.

FORTH in a Nutshell: FORTH is a member of a family of computer languages known as Indirect Threaded Interpretive Languages. The most important characteristic of these languages is that they consist of subroutine-like modules which contain lists of addresses of previously defined modules; none of these address lists is executable. A relatively small number of elementary function machine language routines form the foundation (sometimes called the nucleus or kernal) of these address lists, as all of the lists eventually thread back to these executable code modules. An extremely simple address interpreter processes the address lists.

FORTH-type languages usually contain their own disk operating system, a text editor for source code preparation, a machine language assembler and the language compiler itself. They are all memory-resident, usually use only 10-12 kilobytes, and are well-structured and self-consistent in form.

It is important to see the hierarchal nature of these lists. Each group of addresses, representing sequences of earlier-defined functions, can be represented with only one address in any number of subsequent lists. This resembles a dictionary where new words can be defined from existing, previously-defined, words. FORTH calls this grouping of lists a dictionary, and it is what causes the language to be expandable. The expansion can be to any part of the language already present, such as the editor, compiler, run-time functions, or disk system, or it can be an entirely new application program. For this reason, there is no difference between the body of code called the "language" and the body called the "program," distinctions that characterize almost all other computer languages.

Other features include the (often-criticized) postfix notation (or Reverse Polish Notation — RPN) which greatly simplifies the processing of arithmetic and logic, and a parameter stack for passing results from one defined function to another, eliminating frequent use of variables and thus improving speed.

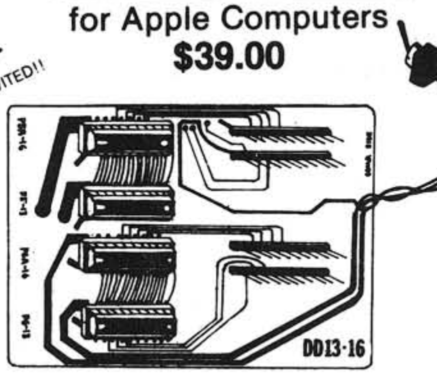
NEW

DEALER INQUIRIES INVITED!!

DOUBLE DOS PLUS

for Apple Computers

$39.00



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.

WHY YOU NEED THE INSPECTOR.

If you're serious about programming, you need to set all your utilities together in one place—*inside* your Apple. The Inspector comes on an Eprom that simply plugs into the D8 socket, or on a disk ready to merge with Integer Basic for automatic loading on boot. Either way, it stays at your fingertips, ready to call without disturbing your current program.

The Inspector puts you in total control of both memory and disks. You can search forward and backwards, edit, read nibbles, map disk space, dump the screen to a printer, examine every secret of your Apple. Use The Inspector to repair blown disks, undelete files, input "illegal" commands, read and alter files, locate strings in memory or on disk. The uses are endless. The manual, alone, is an education. And it's always *there* when you need it.

You need the most powerful disk and memory utility available for your Apple. You need the Inspector.

See your local dealer, or order direct for just $49.95. Mastercard and Visa holders order toll-free, 1-800-835-2246.

OMEGA MICROWARE, INC.
222 SO. RIVERSIDE PLAZA
CHICAGO, IL 60606
312-648-1944

Apple is a registered trademark of Apple Computer, Inc.

Stepper Motor Control: A FORTH Approach

Stepper motors translate digital commands to motion, bridging the gap between computers and robots. A flexible command language, written in FORTH, translates natural, English-like commands to precisely controlled movement.

Mark Bernstein
Department of Chemistry
Harvard University
12 Oxford Street
Cambridge, Massachusetts 02138

If you want your computer to travel around your house, to assemble parts in your factory, to guide your wheelchair or steer your telescope, your computer will need to control a motor. Motors translate electronic commands to physical action. They are "digital-to-mechanical" converters. Let a computer control motors, and it becomes a robot.

Recently, our lab needed a computer-controlled motor for our laser system. We built a carriage which carries a set of laser mirrors, and which rides on a linear track (figure 1). The computer moves the carriage back and forth along the track, measuring the laser's output and its effect on a chemical sample as the carriage moves.

The Stepper Motor

Stepper motors are convenient devices for many computer-controlled chores. Unlike most motors, which revolve continuously, stepper motors move in steps or jumps, one step at a time. The motor's step size is accurately fixed by the motor's construction, with step sizes ranging from 90° to 0.005° or less.

Stepper motors can imitate regular motors by stepping at a constant rate. Unlike other motors, though, they can easily move a fixed distance and then stop immediately without complicated brakes. Steppers can reverse direction at any time, and a stepper motor can easily turn five steps forward, two steps backward, and then start moving forward again.

Steppers do have limitations though. They are generally restricted to moderate speeds and moderate loads. They cannot handle jobs that demand high torques. Unlike other motors, they consume power even when they aren't moving. Small steppers, in particular, tend to get hot.

Still, stepper motors are ideal for all sorts of personal computer tasks. Most personal tasks don't require tremendous speeds or great power, but precise movement and simple, accurate control. These are the steppers' forte.

The integrated hardware and software system described here can control many different types of motors, ranging from miniature versions the size of a quarter to heavy-duty, high-torque models. This hardware tool is controlled using a special control language, an extension to fig-FORTH, which permits simple, convenient programming. The control language "knows" the characteristics of the motor, so the user need not be concerned with such details. Instead, programs use human units and natural commands, like

```
MOVE 5 INCHES FORWARD.
```

The Motor Interface

The heart of our stepping motor interface is the SAA1027 motor driver (figure 2). This IC, manufactured by AIRPAX/North American Phillips and Signetics, can control a wide variety of stepping motors.

The computer provides two simple control signals. To advance the motor one step, the computer transmits a brief negative-going pulse on the (normally high) *step* line. The *direction* control signal selects clockwise or counterclockwise rotation. If this line

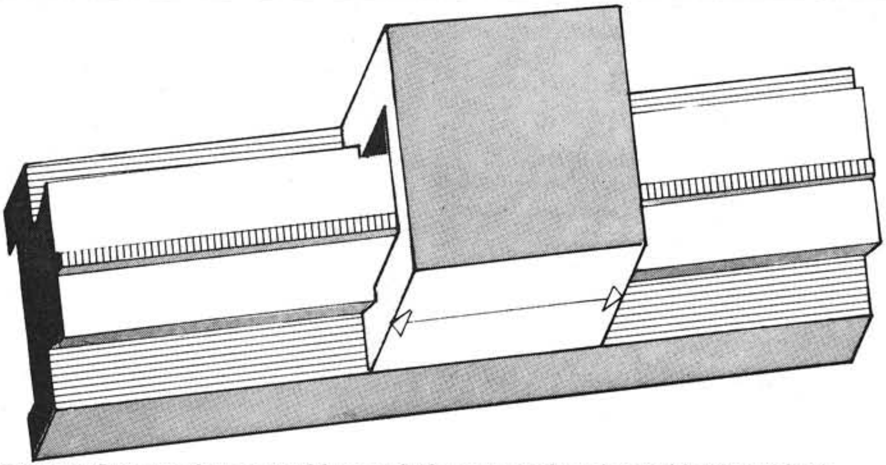


Figure 1: **Our stepping motor drives a platform or carriage (center) back and forth along a 2' track. The motor (hidden behind the carriage) drives a pinion which engages a rack gear embedded in the track. Laser mirrors are mounted on top of the platform, and can be positioned precisely under the computer's control.**

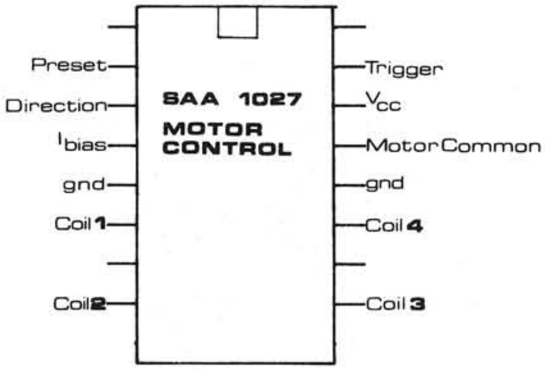


***Figure 2:* The SAA1027 stepper motor controller. This device accepts step and direction commands from the computer, and drives a stepper motor connected to the 4 coil outputs.**

is low, the motor advances in the clockwise direction and if the line is high, the motor turns counterclockwise.

The chip's four outputs, Coil1-Coil4, can be connected directly to a small unipolar stepping motor. Steppers are manufactured in both unipolar (8-wire) and bipolar (4-wire) varieties. The SAA1027 is intended for use with 12V unipolar motors. Adapter circuits can be built for use with bipolar motors, or motors requiring high voltages or large currents.

Both the stepping motor and the SAA1027 IC require a 12V power supply (figure 3). We use an opto-isolator to translate the computer's TTL output signals to the motor's 12V logic levels. The opto-isolators also prevent motor-generated inductive spikes from damaging the computer.

Software Requirements

This interface places few demands on the computer. The computer controls the motor by generating two simple signals.

> The *direction* signal selects either clockwise or counterclockwise rotation.
>
> The *step* signal when pulsed low, commands the motor to advance one step in the selected direction.

No critical timing parameters need be met, except that the motor's maximum speed should not be exceeded. The top speed is a function of the motor's power and torque capacity, and of the load it is asked to move. High speeds can be obtained for light loads, or by using heavy-duty motors. If the motor is stepped too quickly, it may skip or stall.

Of course, we could easily write programs in BASIC or assembler to control such a simple device. For example, we could use a statement like

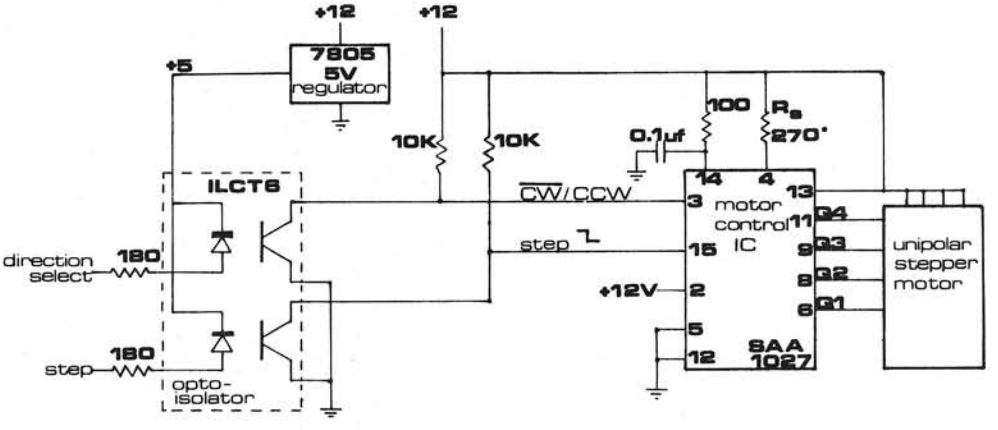
```
POKE  59459 + 15,PEEK(59456
   + 15) OR 64
```

to tell the motor to move clockwise. But FORTH permits us to *extend* the language to make the syntax easier to read, write, and debug. We replace a forest of PEEKs and POKEs with simple statements like "MOVE 1 INCH BACK".

The Motor Commands

All motor-control commands begin with the keyword MOVE. MOVE has no immediate effect, but tells FORTH that motor-control commands follow. Next, we must specify the distance the motor should move. Our motor controls a carriage which travels back and forth along a track (figure 1), so we permit motor motion commands to use either angular or linear units. Examples of legal commands are:

10 INCHES	move 10 inches
1 INCH	move 1 inch
7 CM	move 7 centimeters
70 MM	move 70 millimeters
213 STEPS	move 213 motor steps
90 DEGS	advance motor 90 degrees
3 REVS	advance motor 3 revolutions



*$R_s = 3 \times R_{motor}$

***Figure 3:* The stepper motor interface circuit. Opto-isolators protect the computer's output port and translate TTL logic levels to the 12V logic required by the motor control chip. Small motors can be driven directly by the control IC; larger motors may require external pass transistors on the coil outputs to handle the heavier currents.**

We may optionally specify a motor speed by invoking the commands

FAST	fastest
CRUISE	moderate
SLOW	slow
CRAWL	very slow

The motor speed may also be specified by setting the variable MOTOR-RATE. If no motor speed command is issued, the previous speed remains in effect. If no speed commands are issued at all, a default speed (normally "slow") is assumed.

Finally, the commands FORWARD and BACK perform the actual movement. FORWARD turns the motor clockwise; BACK turns the motor counterclockwise. A distance *must* be specified before FORWARD or BACK can be invoked.

Implementing the Control Language

Listings 1-4 contain a FORTH implementation of this stepping-motor control language. This language extension requires less than fifty lines of FORTH, containing no assembly-language or CODE definitions. To preserve flexibility and assist new implementors, many CONSTANTS and VARIABLES have been explicitly defined.

The first screen of the implementation (listing 1) is devoted to declaring and defining constants and variables. Most of the values are installation-dependent addresses, and may require changes for different computers.

Our system uses a 6522 VIA parallel port (the PET "user port") at address $E840. The motor controls are connected as follows:

direction control	PA6
step control	PA7

Data should be written to the address called M-PORT. The corresponding VIA data-direction register is called M-DIR. The words FWD/REV and < STEP > are bit masks for the direction-control bit and the step-control bit, respectively.

The variable MOTOR-RATE determines the minimum interval between steps. High values correspond to *long* intervals between steps, and so produce *slow* motor rotation. For our motor (1.5^{0}/step) and gearing-ratio (about 0.0031 in./step) typical values of MOTOR-RATE lie between 25 and 5000.

```
HEX
VOCABULARY MOTOR IMMEDIATE
MOTOR DEFINITIONS

( ADDRESS OF MOTOR I/O PORT )
E84F CONSTANT M-PORT

( ADDRESS OF MOTOR DATA-DIRECTION REG )
E843 CONSTANT M-DIR

( MASK FOR MOTOR DIRECTION BIT )
40 CONSTANT FWD/REV

( MASK FOR MOTOR STEP  BIT )
80 CONSTANT <STEP>

DECIMAL 100 VARIABLE MOTOR-RATE

-->
```

Listing 1: **Screen 80 — Stepper Motor, Constants**

```
HEX

( INITIALIZE THE MOTOR VIA PORT )

: (MOTOR) M-DIR @
   FWD/REV OR <STEP> OR
   M-DIR ! ;

( MOTOR SPEED COMMANDS )
DECIMAL

: FAST   20    MOTOR-RATE ! ;
: CRUISE 40    MOTOR-RATE ! ;
: SLOW   100   MOTOR-RATE ! ;
: CRAWL  1000  MOTOR-RATE ! ;

-->
```

Listing 2: **Screen 81 — Initialization, Speed Control**

Listing 2 includes commands for initializing the motor port and for setting motor speeds. The command (MOTOR) ensures that the I/O port is properly configured to *output* data on the direction-control and step-control lines. The speed commands which follow store convenient values into the variable MOTOR-RATE. The actual numbers corresponding to FAST and SLOW depend on the application, and may be altered to suit special needs or different motors. For example, our spectroscopy work requires a very slow CRAWL, but testing and debugging demand that the carriage move fast enough to notice!

Listing 3 contains the commands used to specify the distance the motor should move. Programmers may prefer to use many different units, perhaps even using several different units in one routine. For instance, we use degrees, inches and centimeters in various programs.

The distance specification commands convert all the various human units into motor steps, the unit the motor control understands. The values of the constants STEPS/INCH , STEPS/REV (steps per revolution) and STEPS/CM are determined by the step size and gearing ratio of the motor, and will vary from one type of equipment to another. The appropriate values can be determined from the design of the motor and gear system, or can be found by trial and error.

Notice that INCH and INCHES are both defined (and do the same thing), so that the ungrammatical commands like 1 INCHES are not required.

The word DEGS, which converts degrees to steps, deserves some comment. Fig-FORTH normally calculates arithmetic results using 16-bit signed binary numbers, so that numbers as large as + 32767 can be represented. Since the motor step size may be very small, STEPS/REV may be quite large. Moreover, FORTH truncates standard division results. Hence, it is important to approach the calculation

$$\text{steps} = \text{degrees x } \frac{\text{steps/revolution}}{360}$$

with some caution. FORTH's */MOD

```
DECIMAL

324 CONSTANT STEPS/INCH
240 CONSTANT STEPS/REV
128 CONSTANT STEPS/CM

: INCHES STEPS/INCH * ;
: INCH   INCHES ;

: REVS  STEPS/REV * ;
: REV   REVS ;

: CM STEPS/CM * ;
: MM CM 10 / ;

: DEGS STEPS/REV 360 */MOD SWAP DROP ;

-->
```

Listing 3: **Screen 82 — Motor · Unit Conversion**

performs the multiplication and division in one step, using a 32-bit intermediate buffer. In this way, we can avoid problems with either overflow or truncation.

Listing 4 contains the definitions of STEPS, FORWARD, and BACK, the instructions that actually control the motor. STEPS performs a long DO loop one time for each step of the motor. First, the < STEP > bit of M-PORT is toggled, causing the motor controller to advance the motor one step. Next, STEPS checks the "run/stop" key and calls ABORT (FORTH's version of STOP) if the run/stop key has been pressed. The command " 20 BEEP " makes our system emit a high-pitched beep, indicating that the computer has responded to the stop request.

Finally, provided that the run/stop key was *not* depressed, STEP runs through a delay loop MOTOR-RATE times before proceeding with the next step command.

FORWARD and BACK both leave the stack unchanged, and both call STEPS. FORWARD sets the *direction control* signal high, calling for clockwise motion, while BACK sets the *direction control* signal low, requesting counterclockwise motion. Both FORWARD and BACK end with a command to return the user to the normal FORTH vocabulary.

Finally, we come to the MOVE command (listing 5). MOVE tells FORTH that the following words are to be treated as motor-control commands.

The motor-control commands are all grouped in VOCABULARY MOTOR to avoid confusion with other commands which might share the same names. MOVE tells FORTH to enter the motor VOCABULARY, and calls (MOTOR) to initialize the motor control port. If commands are being entered directly from the keyboard, these actions are taken immediately. If MOVE is entered within a colon definition, these actions are *compiled* into the new definition, to be performed when that definition is invoked.

Multiple Motors

This implementation only drives one motor. Of course, some users might want to control several motors. To control several motor channels, we may assign ID numbers to each device. For example, we might write

```
: CARRIAGE 0 ;
: TRACTOR 1 ;
```

to assign the ID numbers 0 and 1 to the carriage and the tractor motors, respectively.

ID codes are specified immediately after MOVE:

```
MOVE TRACTOR 1 INCH BACK .
```

The words FORWARD, BACK, and STEPS would be rewritten to toggle different bits of different ports, depending on the device code specified. Of course, the actual addresses and bits would vary from installation to installation.

Bugs

This FORTH package handles motor control in a simple and pleasing way. The syntax is attractive, easily-learned and unintimidating. In fact, this sort of control language, resembling stylized English and controlling a big, easily-perceived object, makes a good introduction to programming for younger children.

The design and implementation are not, however, without flaws. Since the

Listing 4

```
HEX

( "10 STEPS" MOVES MOTOR 10 STEPS )

: STEPS
   0 DO
      M-PORT C@ DUP
      <STEP> FF XOR AND
         M-PORT C!  ( PULSE LOW )
      <STEP> OR
         M-PORT C!  ( PULSE HIGH )
      ?TERMINAL    ( STOP KEY PRESSED?
      IF 20 BEEP ABORT ENDIF
      MOTOR-RATE @ ( DELAY LOOP )
         0 DO LOOP
   LOOP ;

-->
```

Screen 83 — Motor · Steps

```
HEX

: FORWARD
   M-PORT C@
   FWD/REV OR
   M-PORT C!
   STEPS [COMPILE] FORTH ;

: BACK
   M-PORT C@
   FWD/REV  0FF XOR  AND
   M-PORT C!
   STEPS [COMPILE] FORTH ;

-->
```

Screen 84: **Forward, Back**

```
FORTH DEFINITIONS

: MOVE [COMPILE] MOTOR
   STATE @ ( COMPILING ? )
   IF
      MOTOR COMPILE (MOTOR)
   ELSE
      MOTOR (MOTOR)
   ENDIF ;
           IMMEDIATE


DECIMAL ;S
```

Listing 5: **Screen 85 — Move**

motor step size is fixed, a request for motion of exactly "1 inch" or "87 degrees" can't always be obeyed exactly. For example, since our motor rotates 1.5 degrees per step, a request for

```
2 DEGS
```

actually causes a motion of only 1.5 degrees. In this implementation, distances are always *truncated*. Proper rounding might be preferable, but has been omitted because it would obscure the simplicity of the unit conversion commands.

Requests for negative distances are not handled correctly. Ideally, FORWARD should refer requests for negative distances to BACK, and *vice versa*. But to keep things simple, we simply forbid negative displacements.

There are presently no provisions for confirming that requested motion is actually occurring, or for detecting spurious movements. Controls could be implemented in STEPS should the application warrant. Attempts to achieve precise position control in any system without some form of feedback to the computer are likely to be fraught with frustration and peril, but again we wanted to present a simple, basic design.

Finally, the control language is, for some, too verbose. A more succinct form for experienced users would be welcome, especially if it preserved the rather pleasant, natural tone of the version presented here.

Parts and Information

The SAA1027 stepper motor control chip is manufactured by the Cheshire division of Airpax, and by Signetics. Single unit prices now run about $15, but substantial price breaks are offered for larger orders.

The Airpax stepper motor catalog lists a large line of inexpensive stepper motors. It also includes an excellent and detailed discussion of stepper motor design and application methods, including a useful compendium of load and torque relationships for common configurations.

Haydon Switch and Instruments, Inc. (Waterbury, CT) makes a series of small and very inexpensive steppers. These small, cheap motors have become quite popular in our labs.

Mark Bernstein is a graduate student in chemistry at Harvard University. He uses microcomputers throughout his experiments, controlling and gathering data from powerful picosecond laser systems.

utility city

21 of our most-asked-for Apple Utilities
by Bert Kersey 48K Applesoft ROM

For YOUR Big Apple—Our 21 most-asked-for Apple Utilities on one big disk—**List Formatter** makes properly spaced & indented listings with printer page breaks. Each program statement is on a new line with If-Then's & Loops called out; a great de-bugger! **Catalog in any number of columns** & any page-width to CRT or printer. Automatically post the Run-Number & last-used Date in your programs. **Make any command invisible** in your listings; Access program lines in memory for garbage repair & "illegal" alteration; Quickly sort & store info on disk; Run any Applesoft file while another stays intact; Move lines within your programs; Renumber to 65535; Save inverse, INVISIBLE & trick file names; Convert dec to hex & binary, or INT to FP; Append programs; dump the text screen to ANY printer . . . More too: **21 Programs Total!**

PLUS . . . APPLE TIP BOOK NUMBER THREE!
40 PAGES of new tips, tricks & articles—"DOS Trickery," "Copy Stoppers," "Programming the Reset Key" & much more. Plus understandable explanations of how each U-City Program works—One of the best Apple learning tools on the market today!

ALL 21 PROGRAMS $29.50 +$1.50 Shipping
✓ **Utility City** on Applesoft Disk.
✓ 40-page Beagle Bros Apple Tip Book # 3
✓ Apple PEEKS, POKES & POINTERS Chart

alpha plot

Hi-Res Apple Graphics/Text Utility
by Bert Kersey & Jack Cassidy 48K REQUIRED

HI-RES DRAWING: Create hi-res pictures & charts, **appendable to your programs.** Keyboard or Paddle control; Optional Xdraw Cursor (see lines before you draw!); Any color mix or REVERSE (opposite of background); Circles, Boxes & Ellipses, filled or not. Bonus Programs too—**SCRUNCHER** stores hi-res in as little as 1/3 normal disk space. **SHIFTER** transfers any portion of the hi-res screen. Also superimpose hi-res images and convert Hi-Res to Lo-Res & back for fascinating abstracts!

HI-RES TEXT: Beautiful upper & lower case with Descenders; color or reverse; Positionable anywhere (NOT restricted by Htabs & Vtabs). Professional looking **PROPORTIONAL SPACING!** Adjustable Type Size, Leading (line spacing) & Kerning (letter spacing). Multi-directional typing; up, down, even backwards!

PLUS . . . APPLE TIP BOOK NUMBER FOUR!
40 PAGES of tips for making the most of your Apple's advanced graphics capabilities. Alpha instructions plus shape table tips, new titling & color tricks and fascinating animation experiments!

$39.50 +$1.50 Shipping
✓ **Alpha Plot** on Applesoft Disk.
✓ 40-page Beagle Bros Apple Tip Book # 4
✓ Apple PEEKS, POKES & POINTERS Chart.

NEW Apple® Utility Disks!
With Free PEEKS & POKES Wall Chart!

Beagle Bros MICRO SOFTWARE
4315 Sierra Vista
San Diego,
CA 92103
714-296-6400

(not available separately)

BONUS WITH EACH DISK!
POKE your Apple all night long with this handy 11 × 17 reference poster! The most useable PEEKS, POKES, POINTERS & CALLS scrounged up from every source imaginable! A great companion to our original Apple Command Chart.

WITH EACH ORDER, YOU WILL RECEIVE . . .
☐ **Dos Boss, Utility City** or **Alpha Plot** on Applesoft Disk
☐ A 40-page **Apple Tip Book** (each entirely different)
☐ An 11x17 Apple **PEEKS, POKES & POINTERS** Chart

dos boss
DISK COMMAND EDITOR
by Bert Kersey & Jack Cassidy

A classic utility you will ENJOY! **Rename Commands:** "Catalog" can be "Cat," etc. **Save-Protect your programs;** Unauthorized copy attempts produce "Not Copyable" message. 1-Key Program Selection from catalog (extremely handy!). Catalog Customizer—Change Disk Volume message to your title; Omit or alter file codes; Catalog by file type, etc. Rewrite Error Messages—"Syntax Error" can be "Oops!" or anything you want!

All of Dos Boss's change feaures may be appended to your programs, so that anyone using your disks on any Apple (booted or not) will be formatting DOS the way you designed it!

PLUS . . . APPLE TIP BOOK NUMBER TWO!
With valuable DOS information: "Poking Around in DOS," "Making Programs Unlistable," "Two-Sided Disk Tips," "Care of Your Drives," "Creative Peeking & Poking," "3.2 vs 3.3," etc.

ONLY $24. +$1.50 Shipping
✓ **Dos Boss** on Applesoft Disk.
✓ 40-page Beagle Bros Apple Tip Book # 2
✓ Apple PEEKS, POKES & POINTERS Chart
MINIMUM 32K REQUIRED

AT YOUR APPLE DEALER NOW. OR ORDER BY PHONE AT ANY HOUR... TOLL FREE!

Ask around about Beagle Bros Software.
Our programs are practical, friendly & user-oriented, and always come with an amazing amount of documentation & juicy Apple information. If your dealer doesn't have our stuff, get on his case about it, or order directly from us. We ship all disks & books WITHIN 48 HOURS.

24 hour TOLL FREE Order Desk:
MasterCard, VISA or COD orders, call:
Nationwide: **800-854-2003** ext. 827
California: **800-522-1500** ext. 827
Alaska/Hawaii: **800-854-2622** ext. 827
(Orders only please. Questions, phone 714-296-6400.1 COD orders, add $3.)

Or order by mail (use this coupon or separate sheet)

RUSH! The disks checked below
Plus The Tip Book and PEEKS & POKES Chart!
☐ **UTILITY CITY** ($31) ☐ **ALPHA PLOT** ($41) ☐ **DOS BOSS** ($25.50)
(Prices include First Class Shipping; Calif. residents add 6%.)
Specify:
NAME$= ______ ☐ **3.2** ☐ **3.3**
ADDR$= ______
CITY$= ______ **ZIP=** ______
VISA **VISA/MASTERCARD: Include Account No. & Expiration Date.** master charge
Mail U.S. check, money order or Visa/MC Numbers
to: **BEAGLE BROS, Dept. T**
4315 Sierra Vista,
San Diego, CA 92103
Telephone (714) 296-6400.
Please add $4.50 for shipping outside US or Canada.
COD orders, add $3. California residents, add 6%.

&Amper~Magic T.M.

by Bob Nacon

MACHINE LANGUAGE SPEED WHERE IT COUNTS...

IN YOUR PROGRAM!

Some routines on this disk are:

Binary file info
Delete array
Disassemble memory
Dump variables
Find substring
Get 2-byte values
Gosub to variable
Goto to variable
Hex memory dump
Input anything
Move memory
Multiple poke decimal
Multiple poke hex
Print w/o word break
Restore special data
Speed up Applesoft
Speed restore
Store 2-byte values
Swap variables

For the first time, Amper-Magic makes it easy for people who don't know machine language to use its power! Now you can attach slick, finished machine language routines to your Applesoft programs in seconds! And interface them by name, not by address!

You simply give each routine a name of your choice, perform the append procedure once at about 15 seconds per routine, and the machine language becomes a permanent part of your BASIC program. (Of course, you can remove it if you want to.)

Up to 255 relocatable machine language routines can be attached to a BASIC program and then called by name. We supply some 20 routines on this disk. More can be entered from magazines. And more library disks are in the works.

These routines and more can be attached and accessed easily. For example, to allow the typing of commas and colons in a response (not normally allowed in Applesoft), you just attach the Input Anything routine and put this line in your program:

```
xxx  PRINT "PLEASE ENTER THE DATE. "; : & INPUT,DATE$
```

&-MAGIC makes it Easy to be Fast & Flexible!

PRICE: $75

For reliability, Aurora Systems uses MAXELL diskettes only.

&-Magic and Amper-Magic are trademarks of Aurora Systems, Inc.
Applesoft is a trademark of Apple Computer, Inc.

aurora systems, inc.
2040 E. Washington Ave.
Madison, WI 53704
608 - 249 - 5875

Introducing:

Disk Library ®

The Software Management System

Disk Library is an elegant, user-oriented system for creating and maintaining a thorough, cross-referenced Index of all your disk-based programs and data files. It provides for **Automatic** entry into your Library file of the full catalog of any Apple* diskette. Disks formatted under other operating systems (such as Pascal and CP/M*) are easily entered from the keyboard. Written entirely in machine code, **Disk Library's** operation is both smooth and swift.

EASY TO OPERATE:

- Menu-driven • User-definable prompt defaults
- Single keystroke operation • Full featured **Editing**
- Super fast **Sorts** (1200 items in 4 sec.!)
- Works with all disks created under DOS 3.1, 3.2 and 3.3
- User definable **Program Types** (e.g., Business, Game, Utility) of up to 15 characters each can be assigned to each program entry with single keystroke or via block actions.
- On-screen and printed **Summaries,** by File Type (Inter, Applesoft, Binary, Text) and by Program Type.
- **Block Actions** (global editing/deleting)
- Instant **Searches. . .** find any item in 1/3 sec.
- **Append** new files to existing ones, in memory or on disk
- **Unique Feature:** User can redefine the Disk Volume Number displayed by the DOS Catalog Command
- Assign a unique Identifier and Title to each disk in your library
- Printed **Reports** are formatted for easy readibility

EASY TO LEARN:

- The 75 page, professionally prepared User's Guide is oriented to the needs of beginners and experts
- Incls.: Tutorials, Reference and Applications Sections, and Index

EASY TO ORDER:

Use your Visa or MasterCard
ANYDAY, ANYTIME

TOLL FREE: 1-800-331-1750 (For orders only)
Ask for operator 948, Dept. 400
(In Okla.): 1-800-722-3600 Operator 948

Or send check or money order to: **MODULAR MEDIA**
Florida residents add 4% sales tax.

PRICE: $39.95 (Special Introductory offer until Feb. 28, 1982) **$49.95** (Regular Price)

WHAT YOU NEED:

- A 48K Apple II or II+ with DOS 3.3
- A desire to get organized!

MODULAR MEDIA

11060 Paradela St., Miami, FL 33156 (305) 661-7310
Developers/Publishers of Innovative Software

* Apple, Apple II and Apple II+ are registered trademarks of Apple Computer, Inc.
CP/M is a registered trademark of Digital Research, Inc.

SENSIBLE SOFTWARE

The Utility People

Making Life With the Apple II A Bit Easier

DISK RECOVERY "THE SCANNER"

48K+, Disk II

Apple II/Apple II+

$30.00

Are you plagued with disk I/O errors?

Does the integrity and safety of your data concern you?

This program is a must for all Disk II owners!!

Just as "Apple Pascal" provides a "BAD BLOCK SCAN", DISK RECOVERY will do a complete scan of your diskettes' recording surfaces. 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 is completely rebuilt to accurately reflect BAD, USED, and FREE SECTORS, further insuring the integrity of the diskette. A complete report is generated advising the user of all corrections BEFORE the object disk is altered. A resulting "DISK MAP" is presented for your review. Flexibility of the program allows evaluation of NEWLY INITIALIZED DISKETTES as well as DISKETTES THAT ALREADY CONTAIN PROGRAMS AND/OR DATA! The SPEED of analysis is unsurpassed - only 18 seconds for a 16 sector diskette! DISK RECOVERY is supplied in the revolutionary QUICKLOADER format which allows easy reconfiguring to make it fully compatible with 13 sector and 16 sector diskettes as well as your SUPER-TEXT data diskettes! TREAT YOURSELF TO PEACE OF MIND KNOWING THAT YOUR PROGRAMS AND DATA ARE SECURE WITH...DISK RECOVERY.

SUPER DISK COPY III $30.00

48K & DISK II required, APPLE II or APPLE II PLUS

Practically a "mini-DOS" in itself, SUPER DISK COPY III has become the standard for APPLE disk utilities. SDC is a menu-driven program that allows manipulation of all types of files under DOS 3.1, 3.2, and 3.3. SDC is the only disk utility available for the APPLE that combines these features: 1) COPY single files (Integer, Applesoft, Binary, or Text), 2) COPY DOS, 3) COPY entire disk, 4) UNDELETE deleted files, 5) Optional "INIT" of copy disk, 6) REPLACE illegal characters in filenames, 7) FIX filesizes, 8) Use of DOS and directory sectors for up to 13 K additional storage, 9) ALPHABETIZE filenames in diskette catalog, 10) LOCK or UNLOCK files, 11) PLOT of disk usage, and 12) Optional rearrangement of files so that they occupy contiguous sectors for improved access times, SDC III now supports the APPLE wildcard character "=" in file specifications. SDC III makes the conversion to DOS 3.3 less painful (than MUFFIN) and also allows files to be transferred back to DOS 3.2 since both 13 and 16 sectored disks can be accessed at the same time.

MULTI-DISK CATALOG III $25.00

48K and DISK II drive required, APPLE II or APPLE II PLUS

MDC is a very fast, machine-language database program designed specifically for keeping track of the contents of your APPLE diskette library, MDC III requires only seconds to read FILENAMES, FILETYPES, FILESIZES, number of free sectors remaining on diskette, and actual volume number from each of your diskettes. Both sides of a diskette can be loaded and assigned to the same DISK ID#. MDC III supports use of a two-character CLASSIFICATION field that can be used to group games, utilities, and other types of related files together. MDC III supports a fast Shell-Metzner sort on any of the five database fields. A unique "LIST MASK" provides a powerful search capability for interrogating the database for specific information. A "FLIP DOS" command allows MDC III to read directories from DOS 3.1, 3.2, and 3.3 disks and to store the resulting database on either a 13 or 16 sectored disk.

SENSIBLE SOFTWARE
6619 Perham Drive Dept MO
West Bloomfield, Michigan 48033
313-399-8877

For a complete catalog, send
$1.00, refundable with your first purchase.
Visa and MasterCard welcome.
Please add $1.25 postage and handling per diskette.
APPLE is a registered trademark of APPLE Computer Company

APPLE • ALTOS • ATARI • MAXELL • DYSAN • EPSON • CCS • SHARP • CASIO • HP • VERBATIM • MEMOREX • SOROC • CORVUS • ADDS

MICRO

BUSINESS WORLD INC.
Information Line (213) 996-2252
TOLL FREE MAIL ORDER LINES
(800) 423-5886 Outside Calif.

COMPUTERS

apple computer

	List	Our Price	SAVE
Apple II-16K	$1330.00	$999.00	$331.00
Apple II-32K	1430.00	SPECIAL CALL	$
Apple II-48K	1530.00		$
Apple II-64K	1725.00		$
Apple III 128-K	3915.00	2997.00	918.00
ALTOS 8CS 8000-15	5990.00	4450.00	1540.00
SHARP-64K Z80 FULL KBD YX-3200			CALL
NEC PC 8001	1295.00	1050.00	245.00

ATARI PERSONAL COMPUTERS	List	Our Price	SAVE
400 16K Bytes of Ram	595.00	337.45	257.05
800 16K Bytes of Ram	$1080.00	739.00	332.00
410 Program Recorder	90.00	77.00	13.00
810 Disk Drive	600.00	457.00	143.00
825 Printer (80 col-Centronic 737)	999.95	769.00	230.00
820 Printer (40 col impact)	450.00	353.00	97.00
830 Acoustic Modem	200.00	155.00	45.00
850 Interface Module	220.00	192.00	48.00
Atari Visicalc	200.00	164.00	36.00
Atari 400-16K	595.00	327.00	304.00

HEWLETT PACKARD	List	Our Price	SAVE
HP-125 Microcomputer	3750.00	2990.00	760.00
HP-85 Microcomputer	3250.00	2475.00	775.00
HP-83 Microcomputer	2250.00	1777.00	473.00
16K Exp-Memory Module	295.00	255.00	40.00
Graphics Plotter 7225	2450.00	2075.00	375.00
Personality Mod. for 7225	750.00	665.00	85.00
2631B Impact/printer/hvy dty	3950.00	3250.00	700.00
Option 020 for 2631B	150.00	125.00	25.00
8 Drives to choose from 82902S	1300.00	1125.00	195.00
9895A 8" Dual Drive	6850.00	5500.00	1350.00
Graphics Tablet 9111A	2050.00	1678.00	374.00
HP-41 CV New 2.2 bytes mem	325.00	250.00	75.00
HP-41-C Calculator	250.00	185.00	65.00
Card Reader for 41CV/C	215.00	162.00	53.00
Printer for 41CV/C	385.00	289.00	101.00
Optical Wand for 41 CV/C	125.00	97.00	28.00
Quad Ram equals 4 Mem. Mods	95.00	81.00	14.00
Memory Modules for 41C		25.00	$
HP-97 Programble Printer	750.00	595.00	175.00
HP-67 Programble Calculator	375.00	295.00	80.00
HP-34C Programble Scientific	150.00	117.00	33.00
HP-38C Programble Bus. R/E	150.00	117.00	33.00
HP-32E Adv. Scientific	55.00	48.00	7.00
HP-37E Business Mgmt.	75.00	57.00	18.00

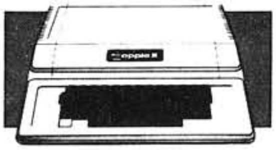
We carry a large inventory of Libraries, accessories and supplies.

PRINTERS

EPSON	List	Our Price	SAVE
MX 80 FT	745.00	549.00	196.00
MX 80 IMPACT	645.00	450.00	195.00
MX 70 IMPACT	500.00	390.00	110.00
MX 100	995.00	765.00	230.00
APPLE SILENTYPE	645.00	299.00	346.00
ANADEX 9501	1650.00	1299.00	351.00
NEC			
5510	3195.00	2445.00	750.00
5515	3295.00	2545.00	750.00
3510	2495.00	1795.00	700.00
3515	2545.00	1849.00	696.00
OKIDATA			
MICROLINE 80	545.00	395.00	150.00
MICROLINE 82	649.00	549.00	100.00
MICROLINE 83	1050.00	769.00	281.00
PAPER TIGER			
445G with Graphics	795.00	695.00	100.00
460G with Graphics	1,394.00	899.00	495.00
560G New full size	1,695.00	1,139.00	556.00
DIABLO (LETTER QUALITY)			
630 R102 bi-directional tractors	2,965.00	2,350.00	615.00
1640K109 keyboard tractors	4,000.00	2,899.00	1,100.05
630 RO Receive Only	2,710.00	2,250.00	460.00
1650K 136 keyboard tractors	4,000.00	3,100.00	900.00

SPECIAL OF THE MONTH

APPLE II PLUS
48K W/16K
EXPANSION BOARD **=64K**



Our price $1199^00

List price $1780.00
Our price $1199.00
Save $581.00

APPLE II STUDENT SYSTEM

- Apple II Plus System-48K RAM
- Disk II Floppy Disk & Interface (DOS 3.3)
- 12" Grn. Phs. Video Monitor

SAVE $655

Our Price $1795

List $2450.00

MONITORS

	List	Our Price	SAVE
NEC Grn. Phs. 12"	$285.00	239.99	$46.00
BMC Green Phs. 12"	275.00	229.00	46.00
Zenith 12"	159.00	119.00	30.00

SANYO MONITORS
High Resolution, Number 1 seller

	List	Our Price	SAVE
13" Color (new) high quality	$550.00	388.00	$162.00
12" Green Phosphorous	360.00	240.00	120.00
12" Black & White	340.00	217.00	123.00
15" Black & White	370.00	235.00	135.00
9" Black & White The Best Seller!	235.00	145.00	90.00

DISKETTES

SOLD IN BOXES OF 10
(Min. Purchase)
BUY $100 of the Following Diskettes

	List	Our Price	SAVE
DYSAN	PRICE PER DISKETTE		
104/1 5" SOFT SECTOR	$ 6.00	3.99	$ 2.01
104/1D = " DBL. DEN.-SOFT SEC.	6.40	4.60	1.80
3740/1 8" SOFT SECTOR	7.25	4.75	2.50
3740/1D 8" DBL. DEN.-SOFT SECTOR	10.75	6.90	3.85
MAXELL			
MD-1 5" SOFT SECTOR/DBL. DEN.	5.00	4.00	1.00
MD-2 = " SOFT SECTOR/DBL. SIDE/DBL. DEN.	7.00	4.90	2.10
FD-1 8" SOFT SEC./DBL. DEN.	6.50	4.50	2.00
FD-2 8" SOFT SEC./DBL. SIDE/DBL. DEN.	8.50	5.95	2.55

TERMINALS

TELEVIDEO	List	Our Price	SAVE
910	699.00	599.00	100.00
912C	950.00	699.00	251.00
920C	995.00	795.00	200.00
950C	1,195.00	949.00	246.00

DRIVES

CORVUS			
5 MBYTES	3,750.00	3.050.00	700.00
10 MBYTES	5,350.00	4,449.00	901.00
20 MBYTES	6,450.00	6,450.00	1,125.00

SOFTWARE

FOR APPLE II

	List	Our Price	SAVE
Apple Fortran	$ 200.00	$ 147.00 $	53.00
DOS 3.3 (16 Sector)	60.00	45.00	15.00
Apple PILOT (16 Sector)	150.00	119.00	31.00
Apple FORTRAN (requires A2B0006) (16 Sector)	200.00	159.00	41.00
Language System with Apple Pascal	495.00	399.00	96.00
BPI General Ledger System	395.00	319.00	76.00
BPI Inventory Package	395.00	319.00	76.00
Visidex	200.00	159.00	41.00
Visicalc	200.00	159.00	41.00
Desktop Plan II	200.00	159.00	41.00
Microlab Database System	150.00	119.00	31.00
Stoneware DB Master	229.00	189.00	40.00
Muse Supertext II	150.00	119.00	31.00
Softape Magic Window	99.00	72.00	27.00

TEXAS INSTRUMENTS

TI 99/4A Console New Improved	950.00	385.95	564.05
10" Color Monitor High Resolution	374.95	317.95	57.00
32K Memory Module	399.95	312.95	87.00
Speech Synthesizer	149.95	127.95	22.00
Disk Memory Drive	499.95	390.95	109.00
RF Modulator	49.95	42.50	7.45
Telephone Coupler (Modem)	224.95	185.95	39.00
Printer (Solid State)	399.95	315.95	84.00

MODEMS

NOVATION CAT MODEM	189.95	140.00	49.95
NOVATION D-CAT	199.00	150.00	49.00
NOVATION APPLE-CAT	349.00	310.00	39.00
HAYES MICROMODEM	379.00	295.00	84.00
HAYES SMARTMODEM	279.00	239.00	40.00

CALCULATORS

CASIO			
HR-10	49.95	39.00	10.95
HR-12	54.95	42.00	12.95
FR-100	79.95	59.00	20.95
FR-1210	129.95	99.00	30.95
PQ-20	29.95	23.00	6.95
LC-785	12.95	10.00	2.95
LC-3165	12.95	10.00	2.95
FX-68 Scientific	29.95	23.00	6.95
FX-81 Scientific	19.95	17.00	2.95
FX-3600P Scientific	39.95	29.95	10.00
FX-602P "Computer Talk" 88 Memories Programming Upper & Lower Case Dot Matrix	129.95	99.00	30.95
FX-702P Solves Problems with Alpha-Numeric Clarity, uses Basic Language	199.95	159.00	40.95

TELE. ANSW. DEVICES

PHONE MATE			
900	119.95	86.00	33.95
905 Remote	199.95	144.00	55.95
910	159.95	115.00	44.95
920	199.95	144.00	55.95
925 Remote	239.95	173.00	66.95
930 Remote	299.95	216.00	83.95
950 Remote	339.95	245.00	94.95
960 Remote	399.95	288.00	111.95

1 Year Extended WARRANTY
$99.00
INQUIRE

$ AVINGS
$ ERVICE
$ ELECTION
$ ATISFACTION

MEMBER BBB

MICRO BUSINESS WORLD
MAIL ORDER WAREHOUSE
18720 Oxnard, #108 Tarzana, CA 91356

Telex: 182852
Answer: MICKOTZNA

OUTSIDE CA CALL TOLL FREE 1 (800) 423-5886 IN CA (213) 996-2252

Name (Please Print) ______
Address ______
City ______ State ______ Zip ______

TOTAL ORDER $ ______
TAX IF APPLICABLE* ______
SHIPPING & HANDLING** ______
TOTAL ENCLOSED $ ______

Qty	Make	Model	Description	Price	Total

*California residents add 6% sales tax
**Add 3% for Shipping & Handling — Add 3% surcharge for credit cards. Orders cannot be shipped unless accompanied by payment, including shipping handling and tax where applicable

Certified Check or M.O.
Bank Wire Transfer

Cash/Credit Card(Allow 2 weeks clearance for checks)

CREDIT CARD # ______
Exp. Date ______ Signature ______

We reserve the right to correct typographical errors. This ad supercedes all previous ads.

MAXELL • DYSAN • EPSON • CCS • SHARP • CASIO • HP • VERBATIM • MEMOREX • SOROC • CORVUS • PERSONAL SOFTWARE • CCS

BMC • CENTRONIC • NEC • HAZELTINE • AMPEX • VERBATIM • HITACHI • HAYDEN • CANON • NOVATION • D.C. HAYS MICROCOMPUTERS • PAPER TIGER • PERSONAL SOFTWARE • GTE •

HP • D.C. HAYS MICROCOMPUTERS • PAPER TIGER • GTE • DIABLO • BMC • CENTRONIC • NEC • HAZELTINE • MEMOREX • AMPEX • PERSONAL SOFTWARE • ADDS • XEROX • DIABLO

Life in FORTH and BASIC

This version of LIFE, written in FORTH, also includes a BASIC program written to use the same logic. The reader can compare the two languages and learn something about FORTH by looking at the comparable BASIC code.

Nicholas J. Vrtis
5863 Pinetree S.E.
Kentwood, Michigan 49508

I have had FORTH running on my SYM for a couple of months, but I hadn't done much with it. I decided I needed a simple project to test my newly acquired FORTH skills. John Conway's *Game of Life* seemed like a good candidate, since it was one of the first assembler programs I wrote when I got my SYM.

Before discussing the programs, I would like to state that I am not a FORTH expert. I do like the language, but I do most of my programming in assembler for a number of reasons. One of the main reasons is that even my stripped down version of FORTH takes 6K and on an 8K system that doesn't leave much room. I do have BASIC on my SYM, but if I had to choose to have only one additional language on my machine besides assembler, it would be FORTH. So much for a disclaimer.

Most articles don't specify boundary conditions. If you have a 16-row by 32-column 'universe,' what is the status of the cell adjacent to column 32 (i.e. column 33)? I chose to make a donut of my universe. That is, the top and bottom rows are logically adjacent, and the left and right columns are also logically adjacent. This involves a few extra checks when counting neighbors, but keeps 'life forms' from falling off the end of the universe.

Most of this description will be on the workings of the FORTH version of the program. However, the BASIC version is included for comparison, since it is logically the same program, and most people are familiar with BASIC. I will include a short paragraph at the end to comment on the differences.

If you look at the listing, you can see that the word LIFE is not defined until two-thirds of the way through the listing. This is because FORTH requires that a word be previously defined before it can be referenced. The first two-thirds of the program define all the words to be used, so that by the time we get around to programming LIFE, all we are doing is putting together routines that FORTH already knows about.

Theoretically, the routines are also tested. In fact, FORTH encourages this two ways. The first way makes it very easy to test out a new subroutine. FORTH has the equivalent of BASIC's direct mode, plus a little. With FORTH, you can define a new piece of code, and still have values in variables and on the stack remain unchanged. Additionally, since the FORTH stack is the major input and output to a word, it is easy to force values onto the stack, use the word, and print the values left on the stack.

FORTH makes it difficult to change a word, therefore you are encouraged to test new words. In order to change a word and not waste memory, you have to "forget" the old version, and then define the new. The catch is that when you forget a word, you also forget everything following it! So, in addition to redefining the word in error, you have to re-define all the others. I guess that would be tolerable if there weren't too many, and you had a disk system, but with a cassette-based system like I have, you wouldn't want to do that often.

The Program

The first few lines define basic parameters of the universe. #/L defines the number of cells per line, and #R defines the number of rows. R*C (rows times columns) defines the total number of cells in the universe. The next two lines reserve space for the universe. FORTH does not have a DIM word, so I defined a variable called GEN-0 (while reserving two bytes for it), and then told FORTH to ALLOT two bytes less than the size of the universe immediately following the space for the GEN-0 value. This means that GEN-0 will return the address of an "array" the size of the universe.

FORTH syntax requires that a variable be defined with an initial value, which is what the 0 in front of the VARIABLE GEN-0 does. The variable IDX is defined to hold the current pointer address into GEN-0 during each generation.

The word CLEAR initializes the universe to zeros (dead cells). It is a simple DO loop, equivalent to the BASIC:

```
FOR I=0 TO R*C: GEN0(I)=0:
  NEXT: RETURN
```

PRINT is used to print a copy of the current universe, using spaces for dead cells, and asterisks for living ones. It is a little more complicated than before. PRINT is set up with nested DO loops. It also shows one of the weak areas of FORTH. There is provision for getting to the value of the innermost DO loop only. The word "I" returns this value, but there is no easy way to get at the index for the outer loop. It must be stored somewhere in the outer loop before starting the inner DO. In the case of PRINT, I put it on top of the FORTH stack.

SET is used to put a value into a particular cell in the universe. FORTH does not handle characters on the stack easily, so the input value is either a two or zero. Note that FORTH does not do subscript checking, nor does SET, so it is possible to overlay the wrong area. BASIC does subscript checking automatically, so it is possible to put the checking in SET.

Another thing to note about SET is the comment on the line starting with ":" (V R C ---). In FORTH it is important, but difficult, to keep track of the stack. I use comments to keep score. The bottom (least accessible) is on the left, and the top on the right. The dashes indicate the use of the word. Any values to the right of the dashes indicate what the stack has after return from the word.

In the case of SET, the top value on the stack is the column to store to, then comes the row, and finally, the value to put there. Also, the comments indicate that the three input values are removed from the stack, and nothing is returned on the stack. When testing a new FORTH word, I always make sure to print the current stack pointer (with SP@ .) before and after the word.

The next word defined is ADD. Its purpose is to check the status of the cell defined by the row and column on the stack, and add to the count of live neighbors if the cell is alive. It has to go through a few gyrations to get the right values on top of the stack.

This is one of the problems with passing parameters on the stack. The third one down is messy to get at, and anything deeper than that is tricky. You also may have the same problem I did with ADD, and may want to get it back where it was.

LIFE is an educational game, invented by John Conway and popularized by Martin Gardner in his "Mathematical Games" column in *Scientific American*. It roughly simulates the processes of reproduction, and death from either overcrowding or isolation. However, the game is really more of a mathematical curiosity, driving thousands of LIFE addicts to distraction.

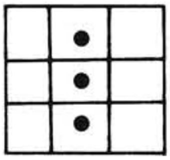
It takes a pencil, pad of graph paper, and lots of time, or a computer program (and lots of paper) to determine the results of each generation. MICRO has published several microcomputer versions of LIFE, including:

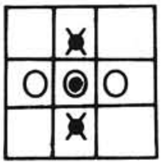
LIFE for Your PET (MICRO 5)
LIFE for Your Apple (MICRO 8)
LIFE for the KIM-1 and an Extended Keyboard Monitor (MICRO 9)
LIFESAVER (MICRO 11)
A Better LIFE for Your Apple (MICRO 15)
LIFE in the Fast Lane (MICRO 16)
A 60 × 80 LIFE for the PET (MICRO 19)
One Dimensional LIFE on the AIM 65 (MICRO 33)

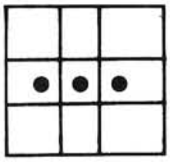
The basic unit of LIFE is a "cell," which lives, dies, and reproduces based on its position relative to other cells. At the beginning of the game, a pattern of cells is entered and the next generation is calculated using the following rules:

1. A cell with two or three neighbors will survive.
2. An unoccupied position with exactly three neighbors will generate a new cell (reproduction).
3. Cells with zero or one neighbor die from isolation, and cells with four or more neighbors die from overcrowding.

Using the oscillating pattern called "traffic lights" as an example, we see that two new cells are created perpendicular to the original line of cells (circles). The end cells die and the center cell remains alive. The corner positions have only two neighbors and remain unoccupied.







Terms such as oscillator, glider, and gun are used to describe the properties of various patterns. For more information consult one of the articles above, or one of the many *Scientific American* columns.

The words B1C (back 1 column), F1C (forward 1 column), D1R (down 1 row), and U1R (up 1 row) are used to count the live cells next to the current cell of interest. Each routine (1) adjusts the appropriate row or column value, (2) checks to make sure it is not out of range, (3) adjusts it if it is, and (4) calls ADD which will increment the neighbor count if necessary.

Those are the last of the new words needed to get LIFE running. All that remains is to put them together to get the desired results. Actually, as you can see, there is still quite a bit of putting together left to do. LIFE is a big DO loop that is performed for the requested number of generations.

Within the outer generation loop are two inner loops, basically the row loop, and the column loop. The program walks around the cell of interest and counts the living neighbors. What follows is an IF statement which will delight structured programming buffs, (but which I personally dislike). The IF statement makes use of the fact that FORTH uses the top value on the stack to decide whether the statement is true or not.

LIFE FORTH Listing

```
BLK #1
10 CONSTANT #/L
10 CONSTANT #R
#R #/L * CONSTANT R*C
0 VARIABLE GEN-0
R*C 2 - ALLOT
0 VARIABLE IDX
BLK #2
( CLEAR THE BOARD )
: CLEAR
R*C GEN-0 + GEN-0 DO
  0 I C!
LOOP ;
BLK #3
( PRINT THE BOARD )
: PRINT
#R 0 DO ( FOR EACH ROW )
  I
  #/L 0 DO ( FOR COLUMNS )
    DUP     ( SAVE ROW     )
    #/L * I + GEN-0 +
BLK #4
    C@
    IF  ( NOT ZERO )
      ." *"
    ELSE
      SPACE
    ENDIF
  LOOP
  DROP
  CR
LOOP ;
BLK #5
( SET A CELL )
: SET ( V R C --- )
SWAP  ( ROW ON TOP)
#/L * + GEN-0 +
C! ;
BLK #6
( ADD TO LIVING CELL COUNT )
: ADD ( # R C --- # R C     )
OVER
#/L *
OVER +
GEN-0 +
C@     ( # R C V )
2 AND
IF ( ALIVE )
BLK #7
  ROT     ( R C # )
  1+      ( ADD TO COUNT )
  ROT     ( C # R )
  ROT     ( # R C )
ENDIF ;

BLK #8
( BACK 1 COLUMN )
: B1C  ( # R C --- # R C )
1 - DUP 0<
IF ( WENT BACK TOO FAR )
  #/L +  ( #/L-1 )
ENDIF
ADD ;
BLK #9
( FORWARD 1 COLUMN )
: F1C  ( # R C --- # R C )
1+ DUP #/L =
IF ( PAST END OF LINE )
  DROP
  0
ENDIF
ADD ;
BLK #10
( DOWN 1 ROW )
: D1R  ( # R C --- # R C )
SWAP   ( # C R )
1 - DUP 0<
IF      ( TOO FAR )
  #R + ( #R-1 )
ENDIF
SWAP   ( # R C )
ADD ;
```

```
BLK #11
( UP 1 ROW )
: V1R  ( # R C --- # R C )
SWAP   ( # C R )
1+ DUP #R =
IF      ( PAST HIGHEST )
  DROP
  0
ENDIF
SWAP   ( # R C )
ADD ;
BLK #12
( LIFE - BY: NICK VRTIS )
( 1/8/81                )
( NUMBER OF ITERATIONS  )
( ASSUMED TO BE ON      )
( TOP OF STACK          )
(                       )
: LIFE
BLK #13
0 DO  ( GENERATION LOOP   )
  GEN-0 IDX !
  #R 0 DO  ( FOR EACH ROW )
    I      ( ROW -        )
BLK #14
    #/L 0 DO ( EACH COL    )
      DUP    ( SAVE ROW    )


      0 SWAP ( # R C       )
      B1C    ( BACK 1 COL )
      V1R    ( UP 1 ROW    )
BLK #15
      F1C    ( FORWARD 1 COL)
      F1C    ( AGAIN        )
      D1R    ( DOWN 1 ROW   )
      D1R    ( AGAIN        )
      B1C    ( BACK 1 COL   )
      B1C    ( AGAIN        )
BLK #16
      DROP DROP     ( #        )
      IDX @ C@      ( # V      )
      SWAP          ( V #      )
      3 - DUP       ( V #-3 #-3)
      IF ( NOT = 3 )
        1+          ( V #-2      )
        IF ( NOT = 2 OR 3        )
BLK #17
                    ( DEATH      )
        ELSE
          IF        ( NO CHANGE)
            3
          ELSE
            0
          ENDIF
        ENDIF
      ELSE
        DROP 1+     ( BIRTH      )
      ENDIF
BLK #18
      IDX @ C!      ( NEW VALUE)
      1 IDX +!      ( NEW INDEX)
    LOOP
    DROP            ( DROP ROW )
  LOOP
BLK #19
  ( MOVE NEW GEN BACK          )
  GEN-0 R*C + GEN-0 DO
    I C@     ( GET BOTH VALUES )
    1 AND    ( KEEP NEW ONLY   )
    IF       ( ALIVE           )
BLK #20
      2
    ELSE
      0      ( DEAD            )
    ENDIF
    I C!     ( STORE NEW VALUE )
  LOOP
  PRINT      ( SHOW THIS GEN   )
  CR
LOOP ;
BLK #21
-30-  ( THE END ..............)
```

LIFE BASIC Listing

```
1  REM  BASIC LIFE
2  REM  N. VRTIS - 1/10/81
3  GOTO 8010
10  REM  ADD TO COUNT
20  IF G%(R * Q + C) AND T

   THEN N = N + W

30  RETURN
100  REM  BACK 1 COLUMN
110 C = C - W: IF C < Z THEN C = Q - W
120  GOTO 20
200  REM  UP 1 ROW
210 R = R + W: IF R = P THEN R = Z
220  GOTO 20
300  REM  FORWARD 1 COLUMN
310 C = C + W: IF C = Q THEN C = Z
320  GOTO 20
400  REM  DOWN 1 ROW
410 R = R - W: IF R < Z THEN R = P - W
420  GOTO 20
500  REM  PRINT THE BOARD
510  FOR R = 0 TO P - 1

   : FOR C = 0 TO Q - 1

520 I = R * Q + C: IF G%(I) THEN

    PRINT "*";: GOTO 540

530  PRINT " ";
540  NEXT : PRINT : NEXT
550  RETURN
1000  REM  START EACH GENERATION
1010  FOR Y = 1 TO X
1015 L = Z
1020  FOR I = 0 TO P - 1

    : FOR J = 0 TO Q - 1

1030 N = Z:R = I:C = J
1040  GOSUB 110
1050  GOSUB 210
1060  GOSUB 310

1070  GOSUB 310
1080  GOSUB 410
1090  GOSUB 410
1100  GOSUB 110
1110  GOSUB 110
1120 H = G%(L)
1130  IF N = E THEN H = H + W

    : GOTO 1160

1140  IF N <  > T THEN  GOTO 1160
1150  IF H THEN H = E
1160 G%(L) = H
1170 L = L + W
1180  NEXT : NEXT
1200  FOR I = 0 TO PC
1210  IF G%(I) AND W THEN G%(I) = T

    : GOTO 1230

1220 G%(I) = Z
1230  NEXT
1240  GOSUB 510
1300  NEXT
1310  END
8000  REM  INITIALIZATION
8010 W = 1:R = 0:C = 0:Z = 0:T = 2
8020 P = 10:Q = 10
8030 PQ = P * Q
8040  DIM G%(PQ)
8050 E = 3:H = 0:L = 0:I = 0:J = 0
8060  FOR I = Z TO PQ:C%(I) = Z: NEXT
8080  INPUT "V,R,C?";I,R,C
8090  IF I < Z GOTO 8200
8100 G%(R * Q + C) = I
8110  GOTO 8080
8200  INPUT "GENS?";X
8210  GOSUB 510
8220  GOTO 1010
```

If the value is zero, it is considered false, and the ELSE part is performed. If it is anything else, it is considered true and the statements after the IF are executed. The end result is a number left on top of the stack with the low bit set for the next generation. The second bit is left the way it was to start.

After taking care of births and deaths for every cell in the universe, the new generation is moved over to take the place of the old, and PRINT is used to show the resulting pattern. Finally, the whole process is repeated if more generations are requested.

Before I go into a discussion of the BASIC version, I would like to point out a few differences between my implementation of FORTH and fig-FORTH. FORTH is really designed to run on at least a 16K system, preferably with diskettes. Unfortunately, I only have an 8K cassette-based SYM. In order to make it fit, I had to remove some of the standard FORTH code, primarily the double integer arithmetic, which is not needed for LIFE anyway.

I also had to change the basic unit of input. Fig-FORTH is designed around a 16-line by 64-character "screen." I only had room for a 128-byte input buffer, and a 128-byte output buffer, if I wanted any room left for programs. Even then I had to put the output buffer on page one.

In order to minimize cassette I/O, I use variable length lines, and put as many as I can into a cassette block, so the listing you see is by blocks, while real FORTH would list by screens. I also compress spaces when putting lines to the buffer, but that is not obvious in the listing, since I put them back on output.

Finally, in fig-FORTH, the word "-->" is used to force compilation to continue from one screen to the next. In mine, I assume you want to continue unless specifically stopped by the new word "-30-". Again, this was done to save valuable buffer space. Other than arranging the source into screens, and adding the --> to the end of each screen this version should run with any standard fig-FORTH system.

The BASIC version was not written with speed as the major criterion. It was mainly written to be comparable to the FORTH code. Where possible, I did make some attempts to speed the BASIC. For example, I used variables instead of constants in lines where they are used frequently, to avoid the conversion overhead. There are also some things which I chose not to do which would have speeded it up. For example, I left in the REM statements for readability, but they can easily be removed, since I did not refer to them in any GOTOs or GOSUBs. Also, I started each routine on an even 100-numbered line. This makes it a little easier to separate the different sections. For speed, it would have been better to start with the line number 1, and increment by 1, to keep the line numbers as small as possible. This would cut down on the source size, and also the amount of time to convert from characters to internal line numbers in GOTO and GOSUB statements.

There are some basic differences between the two languages, which make exact translation impossible. The FORTH version uses only a single byte for each cell, but the minimum in BASIC is two bytes for an integer variable. I could have used PEEK and POKE to cut it down to one byte, but that would have involved setting memory size, and make things harder to understand.

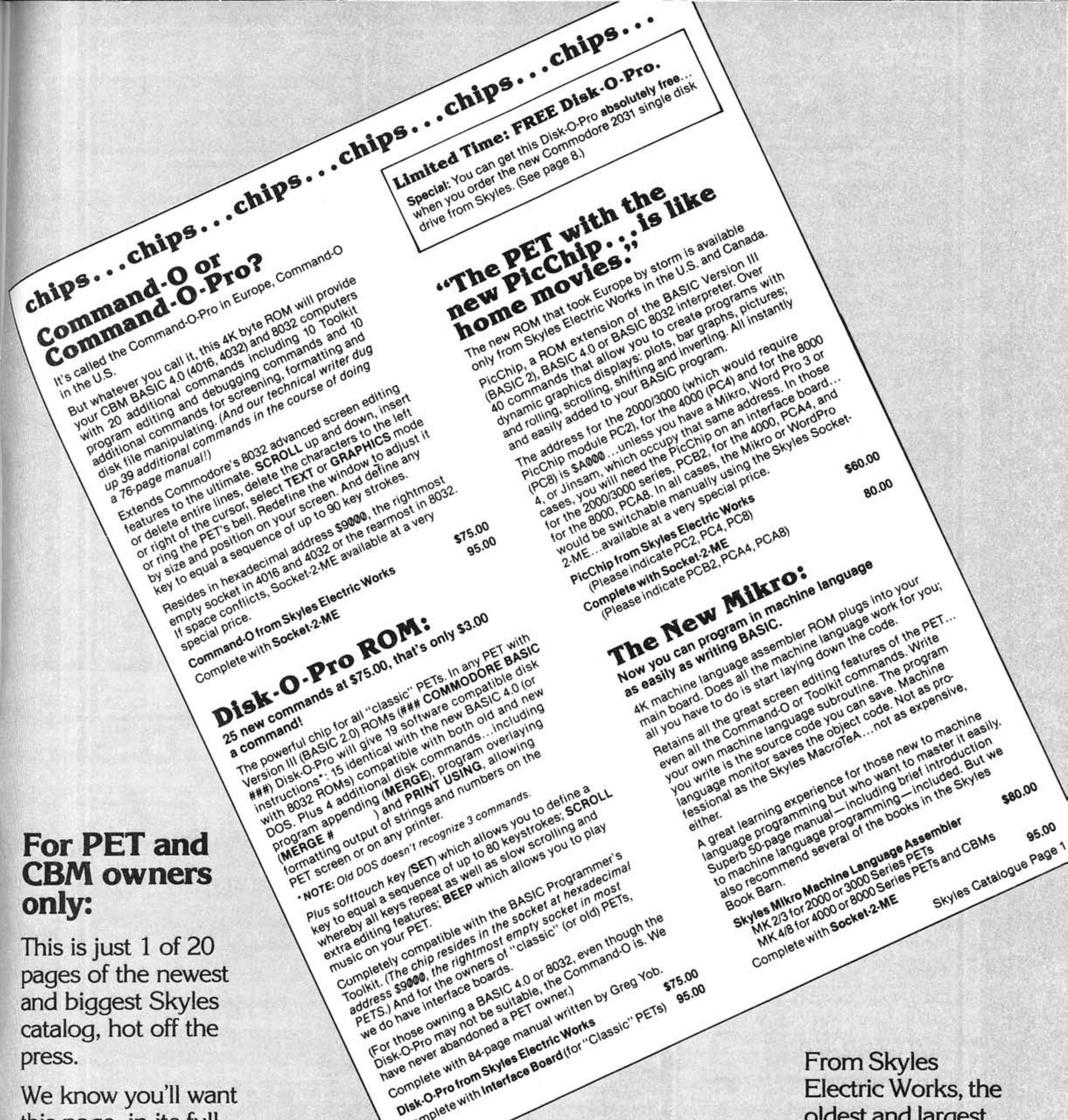
BASIC doesn't have the IF...THEN ...ELSE structure that FORTH has, so I used GOTOs to finish the THEN portion, and the normal statment flow is the ELSE portion. Also, at the end of the BASIC versions of Back 1 column, etc., I used a GOTO to get to the ADD routine and returned from there. To be more faithful to the FORTH version, I should have used a GOSUB and a RETURN, but I just could not bring myself to write code *that* inefficiently.

You should also note that both FORTH and BASIC consider a zero result in an IF statement to be false (i.e., a non-zero value causes the THEN portion to be executed). Most of the BASIC and the FORTH version IF statements are coded the same, but I had to reverse the logic in the section which determines what the value of the next generation of a cell will be (lines 1130 to 1160 in the BASIC version). FORTH uses a compound IF structure which is not available in BASIC. For the FORTH version, the ELSE portion of that big IF statement is equivalent to the THEN part of the BASIC version. I'll let you decide which is easier to understand.

Finally, some speed and size comparisons. Excluding the compiler, the FORTH version takes 614 bytes for a 10 × 10 universe, and goes through four generations in 15.4 seconds (running on a 4800 baud CRT). The BASIC version takes 1318 bytes for the same size universe, and runs four generations in 78.8 seconds. Both were run with the same starting pattern. The FORTH version took me longer to write, but it was also one of my first FORTH programs, so the comparison is not valid. My biggest problem was learning to write GOTOless code!

I will copy my version of FORTH to a cassette that you supply for $5.00 to cover my time and postage. You will still need the fig-FORTH installation manual available from the FORTH Interest Group, P.O. Box 1105, San Carlos, CA 94070. It was $10.00 last time I heard.

Nicholas Vrtis is Manager of Technical Support for Amway Corporation in Ada, Michigan. He has been in data processing since 1969. In 1978 he bought a SYM with 8K. Being a fiddler at heart, his small SYM gives him an opportunity to exercise his talents.



chips . . . chips . . . chips . . . chips . . . chips . . . chips . . .

Limited Time: FREE Disk-O-Pro.

Special: You can get this Disk-O-Pro **absolutely free**... when you order the new Commodore 2031 single disk drive from Skyles. (See page 8.)

Command-O or Command-O-Pro?

It's called the Command-O-Pro in Europe, Command-O in the U.S.

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 disk file manipulating. *(And our technical writer dug up 39 additional commands in the course of doing a 76-page manual!)*

Extends Commodore's 8032 advanced screen editing features to the ultimate. **SCROLL** up and down, insert or delete entire lines, delete the characters to the left or right of the cursor, select **TEXT** or **GRAPHICS** mode or ring the PET's bell. Redefine the window to adjust it by size and position on your screen. And define any key to equal a sequence of up to 90 key strokes.

Resides in hexadecimal address **$9000**, the rightmost empty socket in 4016 and 4032 or the rearmost in 8032. If space conflicts, Socket-2-ME available at a very special price.

Command-O from Skyles Electric Works $75.00
Complete with **Socket-2-ME** 95.00

Disk-O-Pro ROM:

25 new commands at $75.00, that's only $3.00 a command!

The powerful chip for all "classic" PETs. In any PET with Version III (BASIC 2.0) ROMs (**### COMMODORE BASIC ###**) Disk-O-Pro 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 program appending (**MERGE**), program 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 3 commands.*

Plus softtouch key (**SET**) which allows you to define a key to equal a sequence of up to 80 keystrokes; **SCROLL** whereby all keys repeat as well as slow scrolling and extra editing features; **BEEP** which allows you to play music on your PET.

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. We have never abandoned a PET owner.)

Complete with 84-page manual written by Greg Yob.

Disk-O-Pro from Skyles Electric Works $75.00
Complete with **Interface Board** (for "Classic" PETs) 95.00

"The PET with the new PicChip . . . is like home movies."

The new ROM that took Europe by storm is available only from Skyles Electric Works in the U.S. and Canada.

PicChip, a ROM extension of the BASIC Version III (BASIC 2), BASIC 4.0 or BASIC 8032 interpreter. Over 40 commands that allow you to create programs with dynamic graphics displays: plots, bar graphs, pictures; and rolling, scrolling, shifting and inverting. All instantly and easily added to your BASIC program.

The address for the 2000/3000 (which would require PicChip module PC2), for the 4000 (PC4) and for the 8000 (PC8) is $A000...unless you have a Mikro, Word Pro 3 or 4, or Jinsam, which occupy that same address. In those cases, you will need the PicChip on an interface board... for the 2000/3000 series, PCB2, for the 4000, PCA4, and for the 8000, PCA8. In all cases, the Mikro or WordPro would be switchable manually using the Skyles Socket-2-ME...available at a very special price.

PicChip from Skyles Electric Works $60.00
(Please indicate PC2, PC4, PC8)
Complete with Socket-2-ME 80.00
(Please indicate PCB2, PCA4, PCA8)

The New Mikro:

Now you can program in machine language as easily as writing BASIC.

4K machine language assembler ROM plugs into your main board. Does all the machine language work for you; all you have to do is start laying down the code.

Retains all the great screen editing features of the PET... even all the Command-O or Toolkit commands. Write your own machine language subroutine. The program you write is the source code you can save. Machine language monitor saves the object code. Not as professional as the Skyles MacroTeA...not as expensive, either.

A great learning experience for those new to machine language programming but who want to master it easily. Superb 50-page manual—including brief introduction to machine language programming—included. But we also recommend several of the books in the Skyles Book Barn.

Skyles Mikro Machine Language Assembler $80.00
MK 2/3 for 2000 or 3000 Series PETs
MK 4/8 for 4000 or 8000 Series PETs and CBMs
Complete with **Socket-2-ME** 95.00

Skyles Catalogue Page 1

For PET and CBM owners only:

This is just 1 of 20 pages of the newest and biggest Skyles catalog, hot off the press.

We know you'll want this page, in its full 8½ x 10 splendor, and another 19 pages of peripherals, software and books that will make your PET or CBM computer even nicer to live with. So, if we missed sending you your very own copy within the last few weeks, call us at **(800) 227-9998 (unless you live in California, in which case call (415) 965-1735.**

From Skyles Electric Works, the oldest and largest professional specialists in the business.

Skyles Electric Works
231-E S. Whisman Road
Mountain View, California 94041

FORTH Resource List

AB Computers
252 Bethlehem Pike
Colmar, PA 18915
(215) 822-7727
FORTH for PET/CBM

Cap'n Software
P.O. Box 575
San Francisco, CA 94101
(415) 540-0202
FORTH for Apple II

Empirical Research Group Inc.
P.O. Box 1176
Milton, WA 98354
(206) 631-4855
For MC68000

FORTH, Inc.
2309 Pacific Coast Hwy.
Hermosa Beach, CA 90254
(213) 372-8493
General FORTH product support

FORTH Interest Group (FIG)
P.O. Box 1105
San Carlos, CA 94070
(415) 962-8653
2500 members worldwide
FORTH Dimensions — FIG magazine
General FORTH support

FSS
1903 Rio Grande
Austin, TX 78705
(512) 477-2207
FORTH for PET/CBM

George B. Lyons
280 Henderson St.
Jersey City, NJ 07302
FORTH for Apple and others

Frank Hogg Laboratories
130 Midtown Plaza
Syracuse, NY 13201
(315) 474-7856
TRS-80 Color Computer

MicroMotion
12077 Wilshire Blvd., #506
Los Angeles, CA 90025
(213) 821-4340
FORTH 79 for Apple II, II+ and Z80

Mountain View Press
P.O. Box 656
Mountain View, CA 94040
(415) 961-4103
Apple, 6809, and others
Manuals, guides, documentation, machine-readable disks

KV33 Corporation
P.O. Box 27246
Tucson, AZ 85726
(602) 889-5722
Screen editor

This list is not meant to be comprehensive.

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.

™ DiskDoubler is a trademark of Modular Systems

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

OSI ❁ NEW ❁ OSI

FROM THE PRETZELLAND ARCADE

HUMANOID DEFENDER

AS DEFENDER OF THE HUMANOID COLONIES, YOU'VE GOT TO STOP THE ALIEN LANDERS THAT ARE TRYING TO PICK UP AND MUTATE THE HUMANOIDS. IF A LANDER PICKS UP A HUMANOID, YOU HAVE TO BLAST THE LANDER, THEN CATCH THE HUMANOID IN MID-AIR AND LOWER IT SAFELY TO THE GROUND FOR A BONUS! EVERY NOW AND THEN, A BAITER SHIP APPEARS OUT OF HYPERSPACE TO KEEP THINGS INTERESTING! WITH COLOR AND LOTS OF SOUND! 8K CASSETTE.....$9.95
SPECIFY YOUR SYSTEM!

-ASTRO BLASTER-

ASSIGNMENT: CLEAR THE AREA OF HAZARDOUS ASTEROIDS WHICH ARE DRIFTING IN FROM DEEP SPACE, BY BLASTING THEM INTO RUBBLE. BE CAREFUL BECAUSE THE LARGE ONES SPLIT INTO MANY SMALLER ONES WHEN HIT, WHICH FLY OFF IN ALL DIRECTIONS! JUST WHEN YOU THINK YOU'VE BLASTED THEM ALL, MORE APPEAR!
ASTRO BLASTER by JOHN WILSON IS A **MACHINE CODE** OSI VERSION OF ONE OF THE MOST POPULAR ARCADE GAMES OF ALL TIMES! THE ACTION IS VERY SMOOTH AND THE ASTEROIDS ARE THE BEST LOOKING EVER ON OSI- NO LITTLE CIRCLES HERE! AVAILABLE FOR BOTH C1P AND ***C4P*** PLEASE SPECIFY YOUR SYSTEM. 8K CASSETTE $9.95

SEND $1.00 FOR OUR PHOTO-ILLUSTRATED CATALOG & GET A $1.50 CREDIT ON YOUR FIRST ORDER! CASSETTES ONLY!

Pretzelland Software
2005 D WHITTAKER RD.
YPSILANTI, MI. 48197

The BEST games are from Creative Computing Software

1978: Adventure
1979: Air Traffic Controller
1980: Super Invader
1981: Blister Ball and Mad Bomber

Blister Ball

Blister Ball is the first completely original arcade-type game for a computer. Not a copy, not an adaptation, not a spinoff. **Blister Ball** is new—it's a new idea—better than Invaders, better than Circus, better than Asteroids, better than Galaxian. If you've played other games for hours, you'll play **Blister Ball** for days.

How does it work? Well, some mean but fun-loving aliens have produced some bouncing bombs. First they drop one and you've got to position yourself under it and zap it with your laser. If you miss, that's OK. It will bounce around, although each bounce is lower, and you have several chances to zap it. Got the hang of it? OK, here come two bouncing bombs. You zap them. Then you're faced with three, then four and five.

As they bounce longer and longer the walls begin to close in so you're faced with either zapping the bombs or being hit. Each hit knocks you a little further toward the gutter. But you can survive two hits which is usually enough to zap all the bombs.

Feeling confident? Don't. Because after 5 bombs the murderous little devils drop 5 bonus bombs, worth ten times as much. These don't bounce, so you get only one shot. You need nerves of steel and the reflexes of a tail gunner.

After you complete one round, the game starts again with bombs that bounce faster and lower (and are worth more) than the previous ones.

Blister Ball is a fantastic solo game. But there are two-player options as well in which players can play as a team or as opponents. Each player can move the entire width of the screen and zap any of the bombs. Here, you're not only trying to survive, but trying to outscore your opponent. The game has two skill levels.

Mad Bomber

In **Mad Bomber** you are faced with aliens in a huge ship hovering overhead. They have bomb racks which they constantly fill with bombs. Your object is to move from side to side on the ground and zap the bombs in the bomb racks or as they fall.

As the game progresses, the aliens fill up their bomb racks more quickly and the bombs fall faster. You lose after ten bombs have hit the area which you are defending.

Mad Bomber can be played by one player solo or by two players as a team or as opponents. Two skill levels.

Order Today

Blister Ball and **Mad Bomber** are available together for $24.95 on disk (DOS 3.2) only and require a 48K Apple with paddle controls. (We recommend using the Super Paddles from Peripherals Plus).

To order send $24.95 plus $2.00 shipping and handling to the address below. Credit card customers should include card number and expiration date of Visa, MasterCard or American Express card. Credit card orders may also be called in to our toll-free number in the continental U.S.

If you also wish to order a set of Super Paddles from our Peripherals Plus subsidiary, the cost is just $39.95. The paddles are backed by a 90-day limited warranty from the manufacturer as well as Peripherals Plus' moneyback guarantee of satisfaction.

Blister Ball and **Mad Bomber** are colorful, challenging, fast and noisy. They are the games of the year from Sensational Software.

creative computing

Attn: Gladys
39 E. Hanover Avenue
Morris Plains, NJ 07950
Toll-free 800-631-8112
In NJ 201-540-0445

XA-478

MICRO

The Single Life

By Brad Rinehart

Welcome back! This is the second in our continuing series of single-board computer articles. For those of you who missed last month's column, we introduced some of the manufacturers supporting the single-board machines, and described some of the benefits of this type system. This month we will move FORTH and spotlight one of these vendors, Rehnke Software. To many of you this name will sound familiar. For those who are rather new to the single-board scene, Rehnke Software is headed by Mr. Eric Rehnke, a long-time expert in the industry.

Rehnke Software has produced a very fine 6502 FORTH system. This is a "full-featured" system, as opposed to a subset, available for the KIM-1, SYM-1, and AIM 65 computers. It is available on ROM, cassette, or disk for HDE disk systems. 6502 FORTH adheres to the internationally recognized FORTH 79 language standard. In addition, it contains its own editor, assembler, interpreter, compiler, and virtual memory manager. Therefore, 6502 FORTH should be thought of as a complete operating system, not just a high-level language.

Several years ago, little was known about the FORTH language and many people within the industry expected little to come from the introduction of this threaded language. However, FORTH has found its way into the computer industry in many dimensions. It has been used in word processing, data base management systems with remote data gathering, telescope control, financial management systems, numerical control machines, and telecommunications systems. Because of its versatility, FORTH will find its way into many control and data processing applications.

The system resides in slightly less than 12K of memory. It also requires two pages of memory (512 bytes) for stacks, and zero page. The virtual memory manager interfaces text buffers in memory to the cassette or floppy disk mass storage. Any number of buffers may be configured, depending upon the amount of available RAM. The more buffers the user assigns in memory the lower the number of mass storage accesses the system will have to make. Thus the user may build in his own throughput factor.

OK, you say, this is all well and good, but why should I learn FORTH? Well, FORTH will not be the "end-all" language, but it is a very nice language system! FORTH combines the advantages of high-level languages, such as structured programming, extendability, ease of readability, etc., without losing grasp of the machine internals. 6502 FORTH easily interfaces to assembly language routines and has direct access to memory. As I mentioned, 6502 FORTH is a complete system, not a subset. This makes its use in special applications, such as control functions, somewhat unique. A manufacturer may purchase the HDE disk version to produce code for an industrial controller, for example, and then use the ROM version in the device! Keystone Data Consultants is already taking a hard look at possible applications.

In addition to conforming to the FORTH 79 standard, 6502 FORTH combines some unique language features into an already extensive package. Included in its math-handling features is the ability to work with single precision (16-bit), double precision (32-bit), and 11-digit floating point (48-bit) numbers. The built-in floating point routines include the F+, F−, F*, F/, and FSQRT (square root) operators. In addition, necessary stack manipulation words such as FDUP, FSWAP, FDROP, F>, and F=, etc., are also part of the package. Transcendental functions may be performed by adding additional words, as described in the documentation.

6502 FORTH also includes some familiar string handling functions. The BASIC programmer will find LEFT$, RIGHT$, MID$, VAL, etc., quite familiar. This effort to conform a powerful language such as FORTH to command words familiar to most 6502 programmers will help make 6502 FORTH a popular programming tool. This versatility is found in few other high-level languages.

Those programmers who are familiar with HDE's FODS disk operating system will find an interesting parallel in 6502 FORTH. All I/O is routed through jump locations near the beginning of the system, similar to the external jump table in HDE's FODS. This makes it relatively easy to interface 6502 FORTH to any type of I/O device. This can be useful to the manufacturer who is using a disk system to develop his software and wishes to convert the system to a stand-alone device. In addition, a software switch activated by two commands, H-ON and H-OFF, is provided to route output to a hardcopy device, provided one is available. This is similar to HDE's '#' command (such as #LIS), or the CALL function in HDE Disk BASIC (CALL "PTR").

FORTH's editor operates on the 1K blocks called "screens." Commands are included to enter a line of text, delete a line of text, open a space between two lines of text, and edit a line. The edit function is similar to that in HDE's TED or TEXT EDITOR. An 80-column by 24-line CRT is recommended to take full advantage of the editor.

A 6502 macro assembler is also included in 6502 FORTH. The macros include *begin ... until*, *if ... else ... then*, *if ... then*, and several other looping and branch constructs. These macros are used to completely eliminate the use of labels in the assembler. For example, if we had written a loop as:

```
         LDX #8
LOOP     LDA 200,X
         STA 300, X
         DEX
         BNE LOOP
```

We could express the same thing in the 6502 FORTH assembler as:

```
         8 # LDX,
BEGIN,
         0200 ,X LDA,
         0300 ,X LDA,
         DEX,
         0 = UNTIL,
```

The BEGIN statement marks the beginning of the loop, and the 0 = UNTIL statement causes a BNE instruction, as well as the proper offset, to be assembled into memory. This structure may seem somewhat odd at first, but those of you who are familiar with structured languages should recognize the method used here. In addition, once you start using it, the structured language makes more sense. It makes even more sense when you consider that the structure of the FORTH assembler is entirely consistent with the rest of the system! Looping and branching constructs such as *begin ... until, if ... then,* etc., are the same in the FORTH assembler as they are in the high-level FORTH system.

6502 FORTH combines the features that we like in a high-level language system: readability, conformity to existing software, ease of use, etc. Coupled with a disk system, this will make an exciting development tool. As 6502 FORTH is used in more and more applications, I would like to hear about them.

I encourage you to write to:

Rehnke Software
1067 Jadestone Lane
Corona, CA 91720

for information of this complete programming system. It will be a worthwhile investment.

Please address all correspondence for this column to:

Brad Rinehart
1508 Stanton Street
York, Pennsylvania 17404

MICRO

MICRO

6502 Bibliography: Part XLI

1177. Washington Apple Pi 3, No. 4 (April, 1981)

Warrick, Thomas S., "Right-Justification and the Use of Logical Expressions in Calculation," pg. 16-17.
Formatting on the Apple.

Landsman, Richard and Horton, Richard, "DOS 3.3/3.2 Boot Switch," pg. 24.
A switching mechanism using the Control' key on the Apple keyboard to change from 13- to 16-sector operation.

1178. SoftSide 4, No. 8 (May, 1981)

Kinzebach, Wayne; Field, Dave; and Hinkle, Robert, "Atari Oneliners," pg. 35.
Three simple graphics programs for Atari micros.

Jackson, T. and Humphrey, Joe, "Apple Oneliners," pg. 46.
Two simple programs for the Apple.

Truckenbrod, Joan, "Computer Graphics," pg. 53.
Tutorial with a pattern generation program for the Apple.

Voskuil, Jon, "Math Decathlon," pg. 56-58.
Several more events for the Apple in this continuing series.

1179. Creative Computing 7, No. 5 (May, 1981)

Magree, Melvyn D., "Displaying Numbers in Tabular Format," pg. 128-129.
Formatting Floating Point numbers on the OSI C4P-MF micro.

Young, Leland D., "Executive Privilege," pg. 136-142.
Backing up text files with the Apple EXEC command.

Piele, Donald T., "How to Solve It — With the Computer," pg. 146-154.
Part 7 of this continuing series — with an example program in Applesoft BASIC.

Carpenter, Chuck, "Apple Cart," pg. 200-207.
Notes on Apple Master Disk 3.3; the FRE(O) command; graphics; yes/no answers; communications; etc.

Yob, Gregory, "Personal Electronic Transactions," pg. 208-209.
Notes for PET users.

1180. The Transactor 3, Issue 1 (May, 1981)

Anon., "The BMB String Thing!" pg. 7-11.
A string manipulation routine for the Commodore systems.

Anon., "Spooling Disk Files to Printers," pg. 17-18.
Getting the Commodore 2040 disk to talk to the printer while simultaneously running another program.

Collins, John, et. al., "Some Commodore Disk Utilities," pg. 21-23.
Includes an ID reader for disk 4040 and 8050; subroutines returning blocks-free for DOS 2.0 and DOS 2.5; test for PET/CBM and disk.

Easton, John, "202X Bar Graph Printer," pg. 29-32.
A utility for CBM micros.

Anon., "Positioning for DATA READs," pg. 35.
Tips on read statements.

1181. OKC Apple Times 2, Issue 3 (March/April, 1981)

Anon., "IAC Apnote: Apple Post," pg. 9-10.
Review of a mailing list program for the Apple.

1182. The Paper 3, Issue 8/9 (February/March, 1981)

Fowler, Jim, "Assembly Language Programming," pg. 17-18.
Part III of this series discusses style and debugging of PET assembly programs.

Volcheck, Emil, "Auto Repeat Keys; Version Three," pg. 19-20.
The saga of auto-repeat keys for the PET continues with several improvements.

MacArthur, James F., "Merge," pg. 25-27.
A MERGE program for the old ROM 8K PETs.

Batcher, Bill, "Defining Programmable Characters," pg. 28-29.
With your new 2022 or 2023 Commodore printer you can design your own characters.

Anon., "Assembly Language Coding Sheet," pg. 30a, b insert.
Coding sheet and table of op-codes for the PET.

Anon., "PET Memory Map," pg. 53.
Diagram of the PET memory assignments.

1183. Apple/Sass 3, No. 3 (April, 1981)

Lynch, Ron and Burger, Mike, "REM Formatter," pg. 16.
A utility for the Apple.

1184. L.A.U.G.H.S. 3, No. 4 (April, 1981)

Finn, Mike, "Inside Applesoft BASIC," pg. 6-11.
An overview of Applesoft BASIC with floating point addresses.

1185. OSIO Newsletter 3, No. 5 (May, 1981)

Kirshner, Joe, "OS-65D Notes," pg. 1-3.
Notes on the disk directory; random access files; etc.

Valentine, Don, "Switch and Boot with Disk B," pg. 4.
Hardware modification to simplify troubleshooting boot problems.

1186. The Apple Peel 3, No. 5 (May, 1981)

Rogers, James T., "The Epson MX-80 Revisited," pg. 8-9.
Graphics mod for the MX-80 in Apple service.

1187. The Harvest 2, No. 9 (May, 1981)

Stewart, Rob, "Data Communication," pg. 6.
All about modems, bulletin boards, transfer programs, etc. on the Apple.

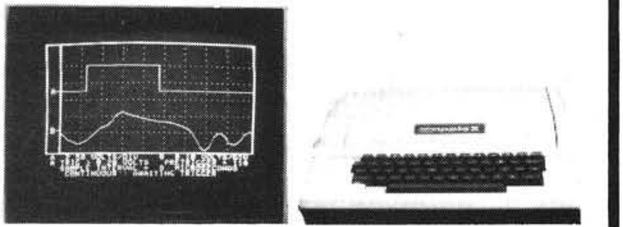
Wilson, J., "Disk Filing Programs," pg. 8-9.
Tips on using Apple disk filing programs.

Anon., "Garbage Collecting Visited," pg. 9.
How Apple memory can be squandered.

APPLESCOPE

DIGITAL STORAGE OSCILLOSCOPE

Interface for the Apple II Computer



The APPLESCOPE system combines two high speed analog to digital converters and a digital control board with the high resolution graphics capabilities of the Apple II computer to create a digital storage oscilloscope. Signal trace parameters are entered through the keyboard to operational software provided in PROM on the DI control board.

- DC to 3.5 Mhz sample rate with 1024 byte buffer memory
- Pretrigger Viewing
- Programmable Scale Select
- Continuous and Single Sweep Modes
- Single or Dual Channel Trace
- Greater than or less than trigger threshold detection

Price for the two board Applescope system is **$595**
EXTERNAL TRIGGER ADDAPTER **$29**

APPLESCOPE EXPANSION CAPABILITY

APPLESCOPE - HR12 High resolution 12 bit analog to digital converter with sample rates to 100 Khz. Software included on disk. Price per channel **$695**

APPLESCOPE - HRHS High Resolution AND High Speed. Combines two 6 bit flash Analog to Digital converters to give 10 bit converter accuracy at a maximum 7 Mhz. sampling rate. Software included on disk. Price per channel **$695**

SCOPE DRIVER Advanced software for the Applescope system provided on 5¼" floppy disk. Available options include:

- Signal Averaging - Acquires 1 to 255 signal sweeps and displays the averaged result.
- Digital Volt Meter - Allows use as a real time DVM or use to measure points on an acquired sweep.
- Hard Copy - Uses graphics printer to produce hardcopy output of displayed traces.
- Disk Storage - Allows automatic storage and recover of acquired data on floppy disks.
- Spectrum Analyzer - Calculates and displays frequency spectrum of acquired data.

The basic SCOPE DRIVER package cost is **$49** plus **$10** for each selected option.

BUS RIDER
LOGIC ANALYZER for the APPLE II

The BUS RIDER circuit card silently rides the Apple II peripherial bus and allows real time tracking of program flow. Software provided on EPROM allows set up of trace parameters from the keyboard and read back of disassembled code after a program has been tracked.

- 32 bit by 1024 sample memory buffer
- Monitors Data and Address bus plus 8 external inputs
- Trigger on any 32 bit word or external trigger
- Pretrigger viewing

The BUS RIDER is an invaluable development tool for anyone working with Apple II or Apple II+ computers. Price **$295**

Apple II BUS EXTENDERS **$19.95**
Allow easy access to Apple II peripherial circuit cards.

SCOPE PROBES 100 Mhz. Bandwidth X1 & X10 switch selectable oscilloscope probes. Price each **$49.95**

For further information contact:
VISA Master Charge
Dealer Inquiries Invited

RC ELECTRONICS INC.
7265 Tuolumne Street
Goleta, CA 93117
(805) 968-6614

Tufts, Terry, "Printing Speeds," pg. 12.
Some revealing information on printing tests on various printers for the Apple.

1188. Personal Computing 5, No. 5 (May, 1981)

Gaylord, Sam, "Generate Lower Case Characters with Pascal," pg. 65-67, 91.
Apple/Pascal without Dan Paymar lower case system modification.

1189. The Michigan Apple-Gram 3, No. 5 (May, 1981)

Rivers, Jerry, "Printer Throughput Testing," pg. 5-7.
A program to evaluate printer speeds.

Wiggington, Randy, "Fast Garbage Collection," pg. 10-15.
A machine language routine to assist in management of the efficient use of Apple's memory; with demo program.

Rivers, Jerry, "INSTR$ Function in Applesoft," pg. 20-24.
A string finding routine for the Apple.

1190. MICRO No. 36 (May, 1981)

Baker, Robert, "KIM/SYM Home Accounting System," pg. 13-16.
A very simple and basic application that requires a minimum of hardware to implement.

Schram, Scott D., "Applesoft Variable Dump," pg. 23-24.
A debugging utility to provide you with a dump of current variable values.

Paris, Greg, "How Microsoft BASIC Works," pg. 31-37.
Variables, their manipulation and the similarity of FNx definitions to variables.

Vrtis, Nicholas J., "SYM-1 Communications Interface," pg. 38-39.
A machine language program for the SYM.

Lourash, Kerry V., "Cursor Control for the C1P," pg. 75-80.
Provide the C1P with some new abilities such as editing, user-selectable windows, one-key screen clear, etc.

1191. Softalk 1, No. 9 (May, 1981)

Anon., "The Mill," pg. 25.
A review of a new Apple peripheral board based on the new 6809E microprocessor, offering Apple users a 8/16-bit architecture, direct page register, extensive addressing modes, fast speed, etc.

Smith, William, V.R., "The BASIC Solution," pg. 42.
A subroutine called READ SCREEN, together with a demo listing for an Apple Auto-Run program.

Wagner, Roger, "Assembly Lines," pg. 67-70.
Part 8 of this informative series including several sound routine examples.

1192. Atari Computer Enthusiasts 2, Issue 5 (May, 1981)

Jones, William B., "Auto-Screen-Editor," pg. 3.
An Atari program to insure that words displayed on the screen are not broken in the middle.

1193. KB Microcomputing 5, No. 5, Issue 53 (May, 1981)

Platt, Charles, "Hot Rod Word Processors," pg. 40-42.
Comparison of the Wordstar (a Cadillac type of word processor) and the WP6502 (a Chevrolet) for OSI micros.

Fowler, Reese C., "Word Processing Roundup," pg. 45-51.
Included in the group of word processors reviewed are the WordPro 4 and the Wordcraft-80 for Commodore systems.

Anon., "Word Processing Directory," pg. 68-71.
A tabular comparison of 33 different word processors for microcomputer systems.

Bazaral, Michael, "AIM for Total Control," pg. 102-104.
The AIM 65 single-board micro functions effectively as a dedicated controller.

1194. Byte 6, No. 5 (May, 1981)

Williams, Gregg, "The Commodore VIC 20 Microcomputer," pg. 46-64.
Review of Commodore's low-cost, high-performance consumer computer.

Sauter, John A., "Faster BASIC for the Ohio Scientific," pg. 236-242.
Several listings to improve the speed of some operations on OSI micros.

White, George M., "Using Interrupts on the Apple II System," pg. 280-294.
Notes on the implementation of the seldom used Apple interrupts.

1195. Applesauce 2, No. 4 (May, 1981)

Mazur, Jeff, "Coping with Apple's Serial Card," pg. 8.
Part II describes a hardware modification for this peripheral card.

Siegal, Ed, "Epson MX-80 Printer Initialization," pg. 10-12.
A utility for this Apple printer combination.

Blumenthal, Jerome B., "Applewriter to Apple Pie Conversion," pg. 23.
Routine to convert Applewriter binary files into Apple Pie text files.

1196. Popular Electronics 19, No. 6 (June, 1981)

Anon., "Atari Model 800 Personal Computer," pg. 48-50.
A review of this 6502-based microcomputer.

1197. Apple-Dayton 2, No. 5 (May, 1981)

Mathews, John, "Telephone Author," pg. 10-11.
Program to make it possible to transfer Apple Writer files over the telephone.

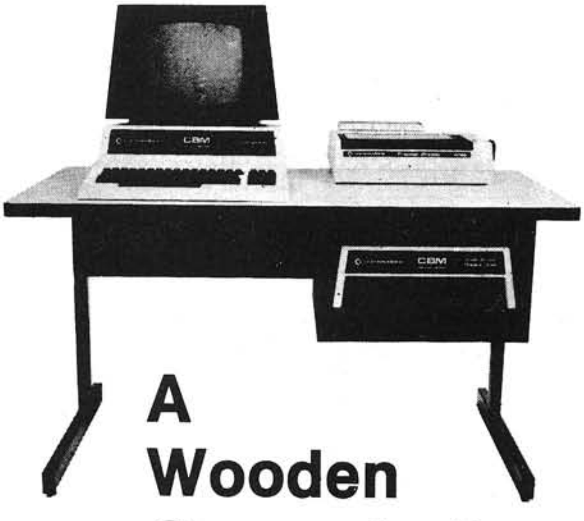
1198. Apple Bits 3, No. 5 (May, 1981)

Kovalik, Dan, "Taking the Mystery and Magic Out of Machine Language," pg. 2-5.
Here is a BASIC program on a Lo-Res Apple graphics game which was then converted to Assembler, statement by statement, so a comparison can be made running both from BASIC and from machine language.

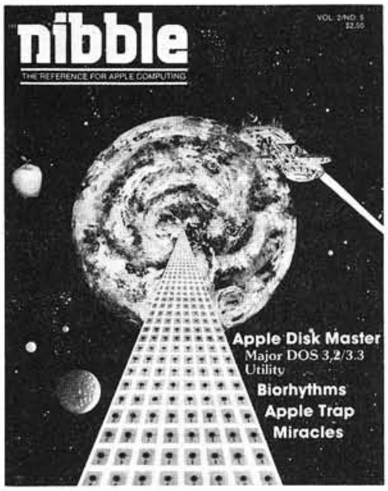
Tulk, Stuart P., "Letter Head," pg. 7.
Here is a program to print a letterhead using the Apple and the MX-80 printer together with a clock card.

1199. Interface Age 6, Issue 6 (June, 1981)

Baxley, David, "Teaching an Old PET New Tricks," pg. 88-91, 148-149.
How to quadruple the resolution on the PET screen.



"NIBBLE® IS TERRIFIC" (For Your Apple)



NIBBLE IS: *The Reference for Apple computing!*

NIBBLE IS: One of the Fastest Growing new Magazines in the Personal Computing Field.

NIBBLE IS: Providing Comprehensive, Useful and Instructive Programs for the Home, Small Business, and Entertainment.

NIBBLE IS: A Reference to Graphics, Games, Systems Programming Tips, Product News and Reviews, Hardware Construction Projects, and a host of other features.

NIBBLE IS: A magazine suitable for both the Beginner and the Advanced Programmer.

Each issue of NIBBLE features significant new Programs of Commercial Quality. Here's what some of our Readers say:

- *"Certainly the best magazine on the Apple II"*
- *"Programs remarkably easy to enter"*
- *"Stimulating and Informative; So much so that this is the first computer magazine I've subscribed to!"*
- *"Impressed with the quality and content."*
- *"NIBBLE IS TERRIFIC!"*

In coming issues, look for:

☐ Stocks and Commodities Charting ☐ Assembly Language Programming Column
☐ Pascal Programming Column ☐ Data Base Programs for Home and Business
☐ Personal Investment Analysis ☐ Electronic Secretary for Time Management
☐ The GIZMO Business Simulation Game

And many many more!

NIBBLE is focused completely on the Apple Computer systems.

Buy NIBBLE through your local Apple Dealer or subscribe now with the coupon below.

Try a NIBBLE!

nibble

We accept Master Charge & Visa

Box 325, Lincoln, MA. 01773 (617) 259-9710

I'll try nibble!

Enclosed is my $19.95 (for 8 issues) Price effective Jan. 1, 1982

(Outside U.S., see special note on this page.)

☐ **check** ☐ **money order**

Your subscription will begin with the next issue published after receipt of your check/money order.

Card # ______________ Expires ________

Signature ______________

Name ______________

Address ______________

City ______________

State ______________ Zip ________

NOTE:
— Domestic U.S. First Class subscription rate is $36.50.
— Write or call for Foreign subscription rates.

All payments must be in U.S. funds drawn on a U.S. bank.

© 1980 by MICRO-SPARC., INC. Lincoln, Mass. 01773. All rights reserved.

* Apple II is a registered trademark of Apple Computer Company.

MICRO Software Catalog

Name: **Color Berserk**
System: TRS-80 Color Computer
Memory: 16K
Language: 6809 Machine Language
Hardware: No extra

Description: *Color Berserk* closely duplicates the popular arcade game with dynamite sound effects and super joy stick action. The exciting combination of angry robots and Evil Orville will provide many hours of constant challenge in this one- or two-player game. *Color Berserk* is a winning addition to the popular Mark Data Products adventure game series.

Price: $24.95 includes game cassette and pre-paid shipping in the U.S.

Author: Ron Krebs

Available:
Mark Data Products
23802 Barquilla
Mission Viejo, CA 92691

Name: **Pilot plus™**
System: TRS-80, Apple II with Z80 card
Language: A high level Pilot with extensions

Description: The first commercially-supported course-authoring language that is transportable across a wide range of micros and minis. This transportability allows authors to develop courseware to be implemented on a variety of systems without modification. This language supports video tape, video disk, touch panel, light pen, color graphics and stored digital voice.

Price: $150.00 per copy, $10.00 for manual

Author: Online Computer System Inc.

Available:
International Institute of Applied Technology, Inc.
20010 Century Blvd.
Suite 100
Germantown, MD 20874

Name: **Directory Master**
System: Apple II
Memory: 48K
Language: Applesoft
Hardware: ROM

Description: A fast machine language utility which lets you directly customize your disk catalogs. Allows creation of attractive diskette headers and file titles with flashing, inverse or normal catalogs. Files may be sorted, hidden, deleted, locked, unlocked and restored with a few keystrokes.

Price: $39.95 includes thorough documentation

Author: Dr. Sandy Mossberg

Available:
Micro-Sparc Systems
Dept. P
P.O. Box 325
Lincoln, MA 01773

Name: **Sight 'n Sound**
System: Apple II or Apple II Plus
Memory: 48K
Language: Applesoft/Assembly
Hardware: Disk drive, cassette player or stereo

Description: Music/sound from a stereo or cassette player is synchronized with a light show of the user's design on the Hi-Res screen. Demos include laser design, kaleidoscope EKG and a guitar player whose lips move with the sound. No hardware modifications.

Price: $24.95 includes diskette and 20-page manual

Author: Ray Balbes, Ph.D.

Available:
Compugraphics Software
#6 White Plains Dr.
Chesterfield, MO 63017

Name: **EAP (Extended Arithmetic Processor)**
System: OSI 65D3 & 65U
Memory: 48K
Language: BASIC
Hardware: OSI

Description: Multiple precision arithmetic for 9-digit Microsoft BASIC. Thirteen-digit input, 26-digit output, rounding function from 0 to 9 decimal places. Easy to use. All functions incorporated as additional BASIC commands. Switchable within program from 9-digit to EAP and *vice versa*. User manual and sample programs.

Price: $95.00 end user, includes 8" disk, 39-page manual

Author: Joan Tirino

Available:
Northeast Financial Systems
16 Maple Ave.
West Nyack, NY 10994

Name: **The Rooms of Cygnes IV**
System: OSI C4P, TRS-80 Models I & II
Memory: OSI - 8K, TRS-80 - 16K
Language: BASIC and machine
Hardware: Amplifier, joysticks (OSI)

Description: You are in a room with walls placed randomly throughout. There are three to ten robots bent on destroying you with laser fire or by touching you. You must destroy all the robots in a room to advance a level. The higher the level the faster the robots. Watch out — the walls are electrified! There are two skill levels. OSI version has sound, color and uses joysticks. TRS-80 version has sound.

Price: $9.95 includes instructions

Author: Mark A. Dickenson

Available:
Comput-U-Gamer Software
P.O. Box 802
Nevada, MO 64772

Name: **The Game Show**
System: Apple II or Apple II Plus, 3.2 or 3.3 DOS
Memory: 48K
Language: Any Apple
Hardware: Disk drive

Description: *The Game Show* is an entertaining one- or two-player educational game. It is particularly helpful for vocabulary development from grade 3 to adult. While *The Game Show* comes with 16 subject areas, you can add your own by using the included authoring system. The author requires no programming expertise.

Price: $39.00 includes detailed manual and diskette, one year warranty and backup diskette.

Author: Geoff Zawolkov, Pete Rowe and Ted Perry

Available:
Computer-Advanced Ideas, Incorporated.
1442 A Walnut St.
Suite 341
Berkeley, CA 94709

Name: **Data Foreman**
System: HDE Disk Systems
Memory: 32 - 56K
Language: HDE Disk BASIC
Hardware: AIM, SYM, KIM using HDE disk system (5" or 8")

Description: General purpose application for data entry and retrieval, report writing, and reload keeping. Allows user-defined data descriptors (headers) and report formats. Typical uses: sales records, telephone list, employee performance evaluations, etc.

Price: $79.95 includes user's manual, shipping within continental U.S.

Available:
Keystone Data Consultants Incorporated
P.O. Box 606
York, PA 17405

Name: **Grafpak.8023 (Hi-Res Graphics Dump)**
System: Apple II or Apple II Plus
Memory: 48K or more
Language: BASIC and 6502 assembly
Hardware: Disk II and NEC PC-8023A-C printer

Description: Dump either Hi-Res page in horizontal 1x or 2x formats, or vertical 1x-3x formats. Dump both pages in a perfectly registered panorama in 1x-3x formats. User-friendly drivers in both BASICs and user program interface instructions. Works with all known production I/O cards for Apple. Works with Apple III in emulator mode. Composer allows edge cropping, image positioning, and image framing.

Price: $34.95 ppd., includes diskette, user's guide, and sample graphics

Author: Robert Rennard

Available:
SmartWare
2281 Cobble Stone Court
Dayton, OH 45431
(513) 426-3579

Name: **Sorcerer of Siva**
System: TRS-80 (cassette or disk), Apple (disk)
Memory: TRS-80 disk, 32K; TRS-80 cassette, 16K; Apple disk 48K
Language: Applesoft
Hardware: TRS-80 (Model I or II), Apple (Applesoft or language card)

Description: The player is a wizard with a variety of magical spells. He must avoid the evil sorcerer who may take his most valuable power away. He battles a multitude of monsters as he searches through more than 300 chambers and five levels of the mine of Siva, seeking the only exit hidden somewhere on the highest level. A built-in scoring system challenges the player to make his way through the mine as quickly as possible. He also receives points for killing monsters and recovering the most valuable treasures. But some measures may even cost him points.

Price: $29.95 inlcudes game box, rule book, special instructions

Author: Gene Rice

Available:
Automated Simulations, Inc.
P.O. Box 4247
Mountain View, CA 94040

Name: **VMIX**
System: VisiCalc File Consolidation Program
Memory: 48K
Language: Applesoft and Assembly
Hardware: Apple II Plus

Description: *VMIX* is a VisiCalc file utility program. Features include: process VisiCalc files of any size, consolidate up to fifty files (nesting consolidations makes the effective number of input files limitless), automatic tabbing between fields, all fields are fully edited.

Price: $80.00 in U.S., $95.00 in Canada

Available:
Evolution Software Inc.
1632 Bathurst Street
Toronto, Canada M5P 3J5
(416) 787-3441

Name: **Masterdisk**
System: Apple II
Memory: 48K
Language: Applesoft
Hardware: 1 disk drive

Description: *Masterdisk* is a disk utility that allows the user to examine and edit any standard disk. Four formatted displays, printer screen dump, undelete, single drive copy.

Price: $29.95 includes disk, documentation, DOS 3.2 and 3.3.

Available:
Masterworks Software Inc.
1823 W. Lomita Blvd.
Lomita, CA 90717
(213) 539-7486

Name: **Autobahn**
System: Apple II or Apple II Plus
Memory: 48K
Language: Machine
Hardware: Disk drive, 13- or 16-sector controller

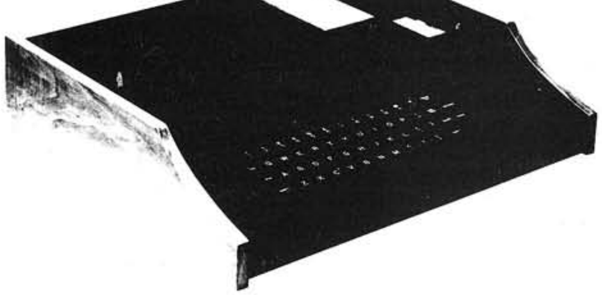
Description: *Autobahn* provides hair-raising excitement at 200 kilometers per hour! This incredible arcade game provides high speed driving excitement through heavy traffic, puddles, dark tunnels (with headlights), and even includes an occasional fire truck you have to avoid. *Autobahn* features super high-resolution graphics and has toggles for sound on/off, restart, and easy shifting of speeds. You can use keyboard or paddle control.

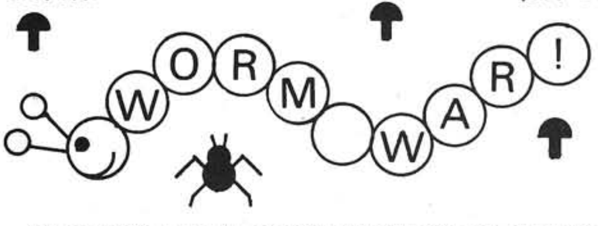
Price: $29.95 includes disk and documentation

Author: NASIR - Presented by Siruius Software, Inc.

Available:
Your local Apple dealer or software store

Name: **Educational Assessment Test Scoring System**
System: Apple II Plus using Microsoft Softcard
Memory: 48K
Language: Microsoft BASIC-80 with CP/M operating system
Hardware: Apple II, Radio Shack Models I, II, III, and PET





Description: The AES *Educational Assessment Test Scoring System* correlates student objectives with specific performance indicators and produces reports for three levels within a school district. The system produces a separate page for each student, indicating for each question whether that student's answer was right or wrong. A second report indicates for each student the percentage of questions answered right or wrong for each educational objective. These same printouts are available to reflect the results of the entire school and the entire school district.

Price: $2000 includes installation, one-day training session, operating manual

Author: Robert C. Hamilton

Available:
Applied Educational Systems
RFD 2 Box 213
Dunbarton, NH 03301

Name: **Castle Wolfenstein^{TM}**
System: Apple II or Apple II Plus
Memory: 48K
Language: Assembly

Description: *Castle Wolfenstein^{TM}* is a new category of action-adventure computer game which bridges the gap between arcade-type games and the more complex adventure/fantasy games. *Castle Wolfenstein^{TM}* demands fast thinking and quick manual response as the player tries to escape from an unlimited variety of guarded rooms in a World War II German-occupied castle.

Price: $29.95 includes documentation

Author: Silas Warner

Available:
Muse Software
330 N. Charles St.
Baltimore, MD 21201

Name: **SMART**
System: Apple II Plus
Memory: 48K RAM
Language: Applesoft BASIC
Hardware: Apple II Plus, disk drive and monitor. Second drive, printer and micromodem recommended.

Description: *SMART* (Securities Market Analysis, Reporting and Transaction System) is a comprehensive, integrated system designed for institutional money managers and sophisticated individual investors. The system provides capabilities for retrieving, storing, graphing and analyzing securities and economic data. A portfolio module provides capabilities for maintaining and reporting data and transactions. Users can automatically access data from one or more remote data bases. Graphic tools include open-high-low-close, volume histograms, line charts and point plots. Analytic tools include moving averages, exponential smoothing, momentum, trend lines, on balance volume and user-defined formulas and routines.

Price: $2150, introductory price includes program disk, data disk, manual

Author: Software Resource, Incorporated

Available:
Software Resources, Inc.
186 Alewife Brook Pky.
Suite 310
Cambridge, MA 02138
(617) 497-5900

Name: **TASC^{TM}, the Applesoft Compiler**
System: Apple II
Memory: 48K
Language: Applesoft BASIC
Hardware: Apple II or Apple II Plus, Applesoft firmware card, one disk drive. Supports but does not require the Microsoft RAMCard or Apple Language system

Description: This Applesoft Compiler from Microsoft converts Applesoft BASIC programs into true machine code. Designed for the Apple owner who writes large and complex programs in Applesfot BASIC, *TASC* can compile at speeds 2 to 20 times faster than the Applesoft interpreter. Other features: reduced program compilation time, minimal code expansion, compatibility with Applesoft so few program modifications are required, BASIC language extensions such as True Integer Arithmetic, inter-program communication possible.

(Continued)

Software Catalog
(continued)

Price: $175.00 includes disk and reference manual

Author: James Peak and Michael T. Howard

Available:
Microsoft
400 108th NE, Suite 200
Bellevue, WA 98004

Name: **Pascal File Exchange (PFX)**
System: Apple II
Memory: 48K RAM
Language: Pascal
Hardware: Apple II or Apple II Plus, language system, two disk drives, Micromodem II or coupler and Apple COM card

Description: *PFX* is an executable Pascal module that permits error-checked telephone transmission of Pascal files between two Apple II computers. Once PFX is running on both machines, the operators may establish a telephone connection, type messages in a "chat" mode, inspect the local and remote directories, schedule and exchange one or more files and initiate the execution of local and remote Pascal code modules. Since a copy of PFX is required at both ends, an auxiliary routine called "Pascal Pull Through" (PPT) is provided to transmit and store a copy of PFX at the "far-end" computer.

Price: $45.00

Author: Graeme Scott

Available:
Arrow Micro Software
11 Kingsford
Kanata, Ontario, Canada
K2K 1T5

Name: **OSI BASIC Enhancer**
System: OSI C1P, Superboard, C4P
Memory: 8K
Language: Machine code w/BASIC-in-ROM
Hardware: C1P, Superboard, C4P

Description: Get real delete action, replace cursor with one of your own choice (defaults to checkerboard square), commands to RENUMBER your programs to make them easy to read, AUTOSEQUENCER will save you from typing in line numbers, screen control stops scrolling, one-key screen clear. A clear screen command has been added to running BASIC. LOAD and SAVE files w/filenames on a token I/O system to reduce load-save times by 50%. Runs in approximately 1.5K of RAM.

Price: $19.95 postpaid includes autoload, autorun cassette only, Users Manual and bug-free guarantee. Or send $1.00 for complete catalog.

Available:
Timothy W. Jackson
c/o Computer Science Engineering
57 Beals Street
Rm. 57-12
Brookline, MA 02146

Name: **Graphic Writer**
System: Apple II or Apple II Plus
Memory: 48K
Language: Applesoft
Hardware: One dirve, DOS 3.3, Graphic Printer, Applesoft Tool Kit

Description: Graphic driver which allows users to get hard copy of the character sets in the Applesoft Tool Kit. May be used in conjunction with Applewriter without affecting any commands. May be incorporated in user's own program for use with print statements, for Silentype and IDS models 440G, 445G, 460GG, 560G. IDS versions require Apple parallel or Apple centronics interface.

Price: $34.95 includes diskette plus full documentation

Available:
Computer Station
11610 Page Service Dr.
St. Louis, MO 63141
(314) 432-7019

Name: **Touch Typing Tutor**
System: Commodore VIC 20
Memory: 5K
Language: BASIC
Hardware: VIC 20 and tape player

Description: Two programs: "19 Lessons" for you to gradually learn proper finger

placement on all keys; "Practice" pseudo words (5-letter in length or random length) with letters occurring at the same frequency they appear in the English language. Learn your WPM rate, % errors, and list errors.

Price: $15.95 includes cassette and manual

Author: Marion H. Taylor

Available:
Taylormade Software
8053 E. Avon Lane
Lincoln, NE 68505
(402) 464-9052

Name: **Super Stellar Trek**
System: Apple II
Memory: 48K
Language: Applesoft in ROM
Hardware: Apple II, Applesoft in ROM, DOS 3.2 or 3.3

Description: This space action game is in Hi-Res color and real time. As a successor to *A Stellar Trek*, it has increased speed, a one-stroke display change, improved visual displays, more sound effects, and ion storms. This program comes with a complete operation manual.

Price: $39.95 includes manual and floppy disk

Author: Tom Burlew

Available:
Rainbow Computing Inc.,
Mail Order Dept.
19517 Business Center Dr.
Northridge, CA 91324

Name: **PLOT65**
System: H.D.E. (Hudson Digital Electronics)
Memory: At least 16K
Language: BASIC
Hardware: Disk drive

Description: *PLOT65* is a collection of subroutines to ease the chore of creating charts and plots for applications programmers. Supports auto axis generation (tic lines), auto scaling, bar generations, shading, writing of test fields, and plotting of single characters. Graphic output is drawn on a Houston Instr. HiPlot plotter Dmp2.

Price: $150.00 includes 5.25 disk containing software, and manual

Author: Micro System Design

Available:
Micro System Design
Assoc. and H.D.E.
distributors

Name: **Compu-Read 3.0; Compu-Math Fractions**
System: Apple, Atari
Memory: 48K
Language: Applesoft, Atari BASIC
Hardware: Apple II, Apple II Plus, Atari 800

Description: *Compu-Read 3.0* contains a series of instructional modules which builds learners' skills by strengthening the perceptual processes required for competent reading. *Compu-Math Fractions*, which builds mathematics skills, allows a learner to interact comfortably with the computer through a sequence of concepts and exercises which reinforce correct performance.

Price: $29.95 for Compu-Read; $39.95 for Compu-Math Fractions — includes documentation

Author: Edu-Ware Services, Incorporated

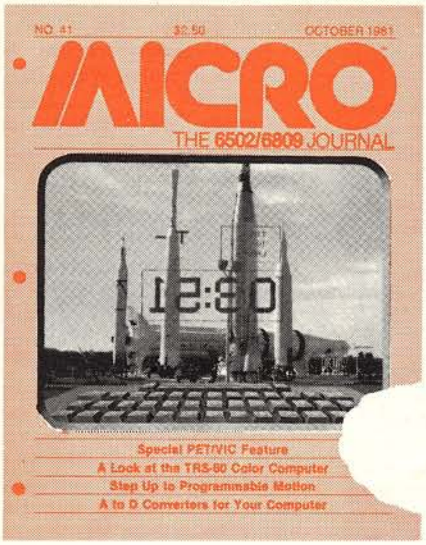
Available:
Edu-Ware Services, Inc.
22222 Sherman Way,
Suite 203
Canoga Park, CA 91303

Answer to 6502/6809 Puzzle: On the 6502, the X and Y registers are 8-bit, and the total time for this double loop is approximately 328 milliseconds, providing a delay of approximately ⅓ of a second for a diskette motor drive to get started. On the 6809, the X and Y registers are 16-bit, and the time for this double loop becomes a bit longer — approximately 9 hours, 32 minutes, and 30 seconds or so!

Answer to Border Puzzle: Welcome to MICRO's world of 1's and 0's!!

MICRO

A feast of computing ideas.



You'll love every byte.

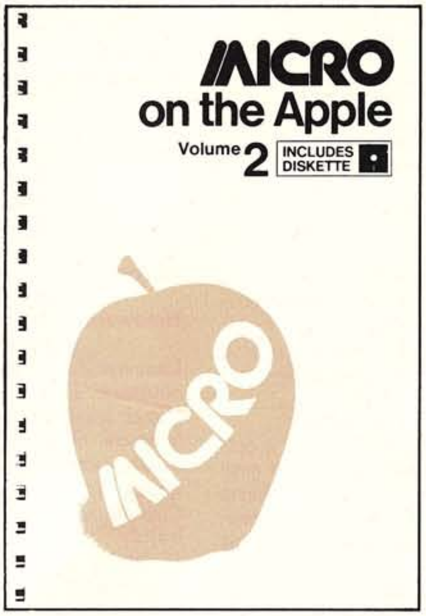
If you work with a 6502 or 6809 based system, you're probably hungry for the facts and ideas that will help you understand the inner workings of your computer. You want to go beyond canned software–use your computer for more than games–learn the advanced programming techniques that enable you to get the most out of your 6502/6809 system.

MICRO, The 6502/6809 Journal, gives you page after page, month after month, of solid information to sink your teeth into. **MICRO** is the premier how-to magazine for serious users of the Apple, PET/CBM, OSI, Atari, AIM, SYM, KIM, and all 6809 based systems including the TRS-80 Color Computer. It's a resource journal internationally respected by professionals in business, industry, and education.

SUBSCRIPTION RATES (U.S. dollars)
Yearly subscription (ISSN 027-9002) saves 20% off the single-issue price.
U.S., $24 (SPECIAL OFFER: Save 30% off single-issue price: 2 years, $42)
Other countries, $27 (via surface mail. Foreign air rates available on request.)

Get more out of your Apple . . .

with the MICRO ON THE APPLE series



VOLUME 2 just released!

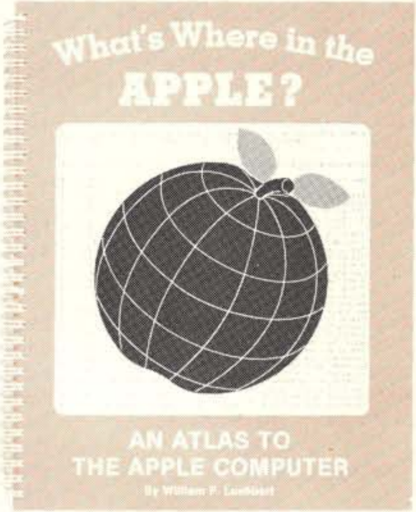
More than 40 new programs on diskette to help you get more from your Apple:

- Machine Language Aids
- I/O Enhancements
- Runtime Utilities
- Graphics and Games
- Hardware and Reference Information

31 choice articles
46 tested programs on diskette
(13 sector DOS 3.2 format)

Volume 1 also available at $24.95. Together **MICRO on the Apple 1 & 2** provide more than 70 programs on diskette for less than $1.00 each. No need to type in hundreds of lines of code.

with the most important book ever published for the Apple



The most comprehensive description of Apple II firmware and hardware ever published—all in one place.

What's Where in the Apple?

- Guides you—with a numerical Atlas and an alphabetical Gazetteer—to over **2,000** memory locations of **PEEKs, POKEs** and **CALLs.**
- Gives names and locations of various **Monitor, DOS, Integer BASIC,** and **Applesoft** routines—and tells you what they're used for.
- Helps BASIC users to speed up their programs.
- Enables assembly language programmers to simplify coding and interfacing.

All Apple users will find this book helpful in understanding their machine, and essential for mastering it!

★ Look for all these **MICRO INK** publications at your local computer store, or

★ Call our **toll-free** number: **1-800-227-1617, ext. 564** (In California, 1-800-772-3545, ext. 564) and charge it to your VISA or MasterCard, or

★ Use the order form below. Send your check (payable to MICRO) and the form to:
MICRO, Dept. OA, P.O. Box 6502, Chelmsford, MA 01824.

QTY	ITEM	PRICE EACH	COST
______	**MICRO on the Apple 2**........	@ $24.95	______
______	**MICRO on the Apple 1**........	@ $24.95	______
______	**What's Where in the Apple?**...	@ $14.95	______
______	**MICRO** (U.S.) .... 1 yr @ $24; 2 yrs	@ $42	______
______	**MICRO** (Foreign) .... 1 yr	@ $27	______
		Subtotal	______
	Massachusetts residents add 5% sales tax		______
	Add $2 **per book** for shipping MICRO Journal excluded. Allow 4-6 weeks for delivery.		______
		TOTAL	______

☐ Check enclosed ☐ Charge my credit card below

Name

Company

Street

City State ZIP

☐ VISA ☐ MasterCard

Signature

Card Number Expiration Date

0001101111001000000100110101001001010000110101001001001111001001110111

MICRO

It's All 1's and 0's

As any serious computerist knows, computers, for all of their great capabilities, can be very frustrating at times. How many times have you been ready to throw in the towel because some dumb little problem was causing your great computer program to fail? That little change which "couldn't be causing this"; that mistyped instruction that you overlooked a hundred times; that "minor exception" which wasn't documented; or, occasionally, a piece of hardware that really was at fault, can all lead to headaches. It happens to all computerists from time to time, and every one has different techniques to overcome these frustrations.

Over a period of several years I worked on a large project that had its share of difficulties. We came up with a rather unusual solution to relieve the tension that these difficulties caused. We had, as part of the project, interfaced a voice synthesizer to the computer and had a number of programs to make it speak. We wrote a special program, hooked it up to be called whenever anyone pressed the PANIC button. When we got disgusted, ready to scream and abandon all hope, we could press the PANIC button. Then the computer would tell us, in its slightly "drunken Swedish" accent: "Its all ones and zeros!" Believe it or not, it seemed to help reduce the frustration and put things back into better perspective.

The purpose of this regular recreational page will be to help relieve some of your computer-oriented frustrations and to keep things in perspective. Most of the material in MICRO is of a rather technical nature and is meant to help you get more out of your computer system. The material in "Its All Ones and Zeros" will be of an entertaining nature and is meant to help you get your computer system out of your nervous system!

Everyone is invited to contribute material. Anecdotes that relate to computers and programmers, puzzles, limericks, poems, cartoons, computer-generated art, unusual signs and slogans, and whatever you think would amuse your fellow MICRO computerists will be considered for this page.

A Few Unusual Computer Instructions

Way back in the distant past, about 1965 or there abouts, IBM was just coming out with its then-super computer, the IBM 360. At that time someone published a list of instructions for this new computer. Many of them have, apparently, been incorporated in other computers since then, but I can only remember a few of them. Does anyone remember more, or maybe have the original list? Here are the ones I remember.

READ AND MUTILATE PUNCH CARD

BACKSPACE AND STRETCH TAPE

INITIALIZE AND SCRATCH DISK

TEST AND DESTROY MEMORY

and, my personal favorite,

EXECUTE PROGRAMMER

A Simple Warning Message

I got my start in computers in about 1966 when I was doing my dissertation on cardiac activity and its relation to eye blinking (really!). I was able to do some of my data measuring and analyses on one of the original LINC computers which DEC had produced for the National Institutes of Health. The Psychology Lab at Duke University had one which I was able to use, but only after 8 p.m. And so, with no one to talk to, I sat there in the middle of the night trying to figure out how to do my project.

All programming was machine level, hand-assembled and entered *via* a bunch of toggle switches. Quite a challenge. One night I tried a new procedure as outlined in the manual. It said that "You will be informed if you make a mistake." Fair enough, I thought as I keyed in several dozen instructions on the switches. I pressed go, and was immediately, and very positively, informed that I had made a mistake. The CRT display showed, in big block letters:

STUPID!

By the cold light of day that may seem harsh, but sitting there at that console at three in the morning, it broke me up. It also helped to relieve the tensions and kept me going until the day shift came in and dragged me away from the computer.

Doctor Bob

A 6502/6809 Puzzle

MICRO started covering the 6809 in June 1981 since it was similar to the 6502 in so many ways. Although the similarities are great, some surprising pitfalls can occur in trying to adapt code between the two. I have a program which works on the 6502 with the following timing loop, but did not work with the 6809 loop.

```
6502                                                6809
      LDX   #$0      INITIALIZE               LDX   #$0
      LDY   #$0      COUNTERS                 LDY   #$0
TLOOP DEX            DECREMENT          TLOOP LEAX  -1,X
      BNE   TLOOP    AND TEST                 BNE   TLOOP
      DEY            DECREMENT                LEAY  -1,Y
      BNE   TLOOP    AND TEST                 BNE   TLOOP
```

and the code continued from here. A note to programmers not familiar with the 6809. The LEAX −1,X is essentially equivalent to the DEX of the 6502 in that it will decrement X and set the zero flag when it reaches 0. Can you figure out what makes these two sections of code, which at first glance seem to be identical, behave very differently? For the answer, see page 121.

The 1's and 0's that make up the border contain a message in 8-bit ASCII. Hint: the message does **not** start at the top left.

00110010000001110111011011110111001001101100011001000010000001101111011001100010000000110001001001

1101110011001000000110000101101110011001000010000000110000001001110111

00110010000000100001001000010101011101100101011011000110001101101111011011010110010100100000011101

MICRO
Hardware Catalog

Name: **Saturn 32K RAM Expansion Board**
System: Apple II
Memory: 48K

Description: Now for the first time ever you can do everything all other 16K RAM expansion boards do, and more! Run Pascal, FORTRAN, PILOT, and all other languages available for the Apple II. Compatible with CP/M and Z-80 SoftCard. Increases VisiCalc memory. Comes with software that automatically relocates DOS onto one of the 16K banks on the board. Additional software will permit the user to store BASIC routines, arrays, data and program segments for overlaying and chaining.

Price: $239.00

Available:
Computer Data Services
P.O. Box 696
Amherst, NH 03031
(603) 673-7375

Name: **Time II**
System: Apple
Hardware: Real Time Clock

Description: *Time II* is a real time clock/calendar for the Apple II computer. Time in hours, minutes and seconds. Date with year, month, date, day of week and leap year. Dip switch selectable interrupts. On board battery backup power for over 4 months power off operation. Battery charges when Apple is on. Includes 16-sector disk containing many programs for your *Time II* clock.

Price: $129.00 (Texas residents add 5% sales tax)

Available:
Applied Engineering
P.O. Box 470301
Dallas, TX 75247
(214) 492-2027

Name: **Utility ROM**
System: Apple II or Apple II Plus
Hardware: M.C.'s ROMPLUS or Andromeda's ROMBoard

Description: Five Applesoft and disk utility programs that (almost) everyone has in software but never gets to use because it's too much trouble to load and run them. Because ROM memory never forgets, you can now access these five utilities instantly without having to load them from disk. With the *Utility ROM*, you can do automatic line numbering, control a program list-out with character or page mode, restore a crashed program in memory, alphabetize the catalog directory on a disk, and create a disk without DOS to provide an additional 8K of space on a disk.

Price: $39.95

Available:
Soft CTRL Systems
P.O. Box 599
West Milford, NJ 07480
(201) 728-8750

Name: **CPS Multi-Function**
System: Apple II

Description: Provides all the capabilities of a serial interface, parallel output interface and real-time calendar/clock—all on one card—occupying only one slot in the Apple II. Serial and parallel output may be used simultaneously from CPS.

Price: $239.00

Available:
Mountain Computer Inc.
300 El Pueblo Rd.
Scotts Valley, CA 95066
or from your local Apple dealer

Name: **Elf 2 Computer Interface** for IBM Electronic Typewriter
System: Any RS-232C or parallel printer port
Hardware: Any mini- or microcomputer

Description: *ELF 2 Interface System* accepts any ASCII-coded data from user's computer and transfers it in appropriate format to control IBM electronic typewriter Models 50, 60, and 75, producing letter-quality hard copy output. Ideal as a low cost word processor output device, the *ELF 2* also permits full use of all typewriter functions. Also available with IBM factory reconditioned Model 50 typewriter as a complete typewriter/printer package.

Price: $495.00 includes complete step-by-step installation manual and operating instructions. Brochure on request.

Available:
iPEX International, Inc.
5115 Douglas Fir Road
Calabasas, CA 91302
(213) 710-1444

Name: **Hayes Stack^{TM} Chronograph**

Description: *The Chronograph* is a stand-alone, RS-232C-compatible calendar-clock that reports date, weekday and time in 12- or 24-hour modes. Features include quartz-crystal precision, easy-to-read display, computer alarm, write-protect switch, battery backup and automatic leap-year adjust. *The Chronograph* is suitable for virtually any application requiring accurate timekeeping. It is covered by a two-year limited warranty.

Price: $249.00 includes Chronograph unit, power pack, batteries and owner's manual.

Available:
Hayes Microcomputer Products, Inc.
5835A Peachtree Corners E.
Norcross, Georgia 30092

Name: **PEDISK II**
System: Rockwell AIM
Memory: 16K minimum
Hardware: Rockwell AIM board

Description: *PEDISK II* is a high performance floppy disk system for the Rockwell AIM. Available with up to three drives, it can accommodate either 5¼" or 8" disk drives. The 8" drive offers IBM 3740 compatibilitiy.

Price: $595.00 includes controller, cable, 5¼" drive and DOS software.

Available:
CGRS Microtech
P.O. Box 102
Langhorne, PA 19047

Name: **GMS6514 6PIB Module**
System: Motorola Exorciser/Micromodule, Rockwell System 65/AIM 65
Hardware: 6" × 9.75" module

Description: A general purpose interface bus (6PIB) module which meets all IEEE-488 specifications, with control software, a wire-wrap section for custom input/output, operating on a single +5V power supply. Uses TMS 9914 LSI bus controller, has pass control and system control capabilities with device clear and trigger functions, parallel and serial pull, service request and remote/local selection.

Price: $250.00 in single piece quantity, includes 72-hour burn-in, one-year warranty

Available:
General Micro Systems, Inc.
1320 Chaffey Ct.
Ontario, CA 91762
(714) 621-7532

Name: **Input/Output Board**
System: Apple

Description: Provides eight buffered outputs to a standard 16-pin socket for standard dip ribbon cable connection. Power-up reset assures that all outputs are off when your Apple is first turned on. Features eight inputs that can be driven from TTL logic or a 5-volt source. Your inputs can be anything from high speed logic to simple switches. (Internal pull-up resistors provided.) Very simple to program — just PEEK at the data.

Price: $62.00 (Texas residents add 5% sales tax).

Available:
Applied Engineering
P.O. Box 470301
Dallas, TX 75247
(214) 492-2027

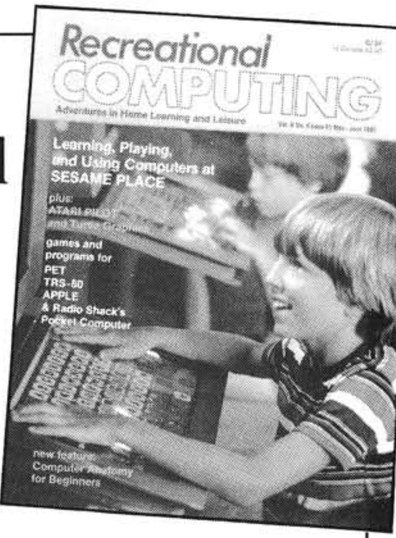
Name: **Eye Lupes, Magnifiers and Comparators**
Hardware: Optical Components

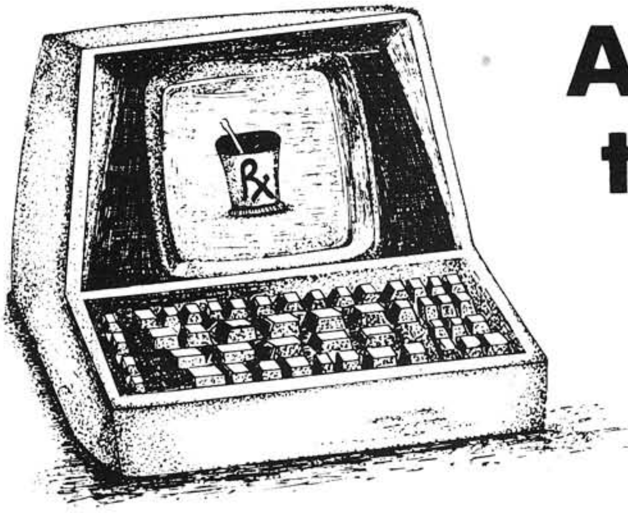
Description: Catalog 81 filled with optical components, eye Lupes, magnifiers and comparators. Also a special section on microscopes and micrographic equipment. Catalog available *free* on request.

Price: Products range from $2.00 - $279.00

Available:
Keyan Industries, Inc.
196 Plain St.
P.O. Box 183
Braintree, MA 02184
(617) 848-7636

MICRO


Recreational
COMPUTING
Adventures in Home Learning and Leisure
Learning, Playing, and Using Computers at SESAME PLACE
plus: ATARI PILOT and Turtle Graphics
games and programs for PET TRS-80 APPLE & Radio Shack's Pocket Computer
new feature: Computer Anatomy for Beginners


Rx

MICRO

Advertiser's Index

Next Month in MICRO

OSI Feature

- **A Cross-Reference Generator —** This article describes a cross-reference generator for OSI ROM BASIC which will aid in finding any variable or line number within a BASIC program.

- **Data Statement Generator —** A convenient, machine language program to convert machine language routines to BASIC DATA statements is presented. It can be applied to all OSI BASIC-in-ROM machines.

- **Autonumber Plus for the Cursor Control —** This program automatically increments and prints line numbers when you type in BASIC programs.

Math Feature

- **Numerical Solutions of Ordinary Differential Equations —** This article includes an Applesoft program and short description of a fourth-order Runge-Kutta method for solving ordinary differential equations.

- **Legrange Interpolating Polynomial —** Use this routine to fit a curve to your data using as few as four data points. Runs in any machine with floating point BASIC.

- **VisiCalc Formulas for Depreciation —** A discussion on the use of VisiCalc to compute depreciation.

- **SIN(X) The Hard Way —** Microsoft BASIC uses a series expansion formula to calculate the size of an angle. The logic of this machine language routine is emulated here in a BASIC program.

Other Articles

A Disassembler for the 6809
KIM-1 Bouncy Keypad Cure
An I/O Expansion for the AIM

Plus our regular columns and departments!

20% OFF

Your money goes farther when you subscribe. During the course of a year, when you subscribe, you save 20% (in the U.S.).

Pay only $24.00 ($2.00 a copy) for 12 monthly issues of MICRO sent directly to your home or office in the U.S.

More MICRO for Less Money When You Subscribe

But on the newsstand — if you can locate the issue you want — you pay $30.00 a year ($2.50 a copy).

Special Offer — Subscribe for 2 years ($42.00) and get 30% off the single issue price.

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 Effective January 1, 1982

Country	**Rate**
United States	$24.00 1 yr. 42.00 2 yr.
Foreign surface mail	27.00
Europe (air)	42.00
Mexico, Central America, Mid East, N. & C. Africa	48.00
South Am., S. Afr., Far East, Australasia, New Zealand	72.00

* Airmail subscriptions accepted for only 1 year.
For U.S. and Canadian 2-year rates, multiply by 2.

Job Title: ________________

Type of Business/Industry: ________________

HUNTINGTON COMPUTING

ONE OF THE WORLD'S LARGEST INVENTORIES

VisiCalc	List $200.00	now $149.00
VisiFile	List $250.00	now $199.00
Horriblescope (Disk)	List $15.00	now $3.99

MISCELLANEOUS

Escape from Acturus	$29.95 now	$25.39
Basic Mailer	$69.95 now	$59.49
Memory Management II	$49.95 now	$42.89
Castle Wolfenstein	$29.95 now	$25.39
Upper Reaches of Apshai	$19.95 now	$16.89
Bridge Tutor	$39.95 now	$35.19
PASCAL Animation Tools		$65.99
Hand Holding BASIC		$84.99
UT-100 Emulator		$65.99
Space Quark	$29.95 now	$25.39
Beneath Apple DOS (book)	$19.95 now	$16.89
Birth of the Phoenix	$14.95 now	$12.69
Goblins	$27.50 now	$23.29
Painter Power	$39.95 now	$33.89
U.S. Constitution	$29.95 now	$25.39
Merger	$49.95 now	$42.49
Super Stellar Trek	$39.95 now	$33.89
LISA	$79.95 now	$67.89
V-Plot (Yuccipa)	$29.95 now	$25.39
V-Stat	$29.95 now	$25.39
V-Print	$29.95 now	$25.39
Brain Surgeon	$49.95 now	$42.49
Info Master	$150.00 now	$127.49
Waterloo II	$49.95 now	$42.39
Fantysland 2U41	$59.95 now	$50.99
Torpedo Terror	$24.95 now	$21.19
Speedstar	$139.95 now	$114.69
Kaves of Karkhan	$49.95 now	$42.39
Dos Boss	$24.00 now	$20.39
Memorex Disks		10 for $24.99
Verbatim Datalife-plain w/hubs		10 for $27.99
Dysans		10 for $49.99
Flipsort Box		$29.80
Scotch Disk Cleaner	$29.95 now	$26.99
Atari*/TRS-80*/Pet*		Write for Information

GAMES

Hi-Res Soccer	$29.95 now	$25.39
Apple*-Oids	$29.95 now	$25.39
Wurst of Huntington Computing		$19.99
Gobbler	$24.95 now	$21.19
Ultima	$39.95 now	$33.89
Autobahn	$29.95 now	$25.39
Battle Cruiser Action	$44.95 now	$38.89
Gorgon	$39.95 now	$33.89
Super Stellar Trek	$39.95 now	$33.89
Hellfire Warrior	$39.95 now	$33.99
Gamma Goblins	$29.95 now	$25.39
Mission Asteroid	$19.95 now	$17.99
Wizardry	$49.95 now	$42.49
Star Mines	$29.95 now	$25.39
Warp Factor	$39.95 now	$33.99

WURST OF HUNTINGTON

Read **Softalk's** review on page 87 of the October issue

$19.99

NEW PROGRAMS

Sneakers	$29.25 now	$25.39
Copts & Robbers	$34.95 now	$29.69
Pot-O-Gold I	$39.95 now	$33.89
Pot-O-Gold II	$39.95 now	$33.89
Ring of Saturn	$39.95 now	$33.89
Streets of the City & Trucker	$24.95 now	$21.19
President Elect	$39.95 now	$33.89
Dis-o-Doc	$72.39 now	$61.49
Med Fly Mania	$29.95 now	$24.39
Hungry Boy	$24.95 now	$21.19
Anova	$74.95 now	$63.79
Solitaire	$29.95 now	$25.39
The Source	$100.00 now	$89.99
Race for Midnight	$29.95 now	$25.39
Hi-Res Secrets	$125.00 now	$106.19
Banners I	$34.95 now	$29.69
Complete Graphics (Tablet)	$119.95 now	$101.89
Special Effects (Tablet)	$69.95 now	$60.79
Inkwell Apple* III	$150.00 now	$130.49
Snack Attack	$29.95 now	$25.39
V-Plot	$29.95 now	$25.39
Disasem/65	$29.95 now	$25.39
The Inspector (Disk or Prom)	$49.95 now	$42.39
The Book	$19.95 now	$17.97
Don't (Book)	$8.95 now	$8.05
Using 6502 Assembly Language	$19.95 now	$16.89
Expandaport	$59.95 now	$50.89
Firebird	$29.95 now	$25.39
Thief	$29.95 now	$25.39
Threshold	$23.97 now	$20.38
Target Planner	$195.00 now	$165.69
Paddle-Adapple	$29.95 now	$26.89
So. Cal Research	$14.95 now	$13.45
Word Handler	$250.00 now	$212.49

GAMES

Microsoft Adventure	$29.95 now	$25.39
Wizard and the Princess	$32.95 now	$28.99
Flight Simulator	$34.95 now	$29.49
Odyssey	$29.95 now	$25.39
Sargon II	$34.95 now	$29.69
Space Eggs	$29.95 now	$25.39
Hi-Res Cribbage	$24.95 now	$21.19
Lords of Karma (cass.)	$20.00 now	$16.99
Oh Shoot	now	$19.99
ABM	$24.95 now	$21.19
Computer Conflict	$39.95 now	$35.99
Computer Air Combat	$59.95 now	$52.99
Temple of Apshai	$39.95 now	$33.89
Zork	$39.95 now	$33.99
All Nibble Software		15% off list
Robot Wars	$39.95 now	$33.99
Cranston Manor	$34.95 now	$29.69
Dragon's Eye	$24.95 now	$21.19
Computer Acquire	$20.00 now	$16.99
Twala's Last Redoubt	$29.95 now	$25.39
Alien Rain	$24.95 now	$21.19
Alien Typhoon	$24.95 now	$21.19
Raster Blaster	$29.95 now	$25.39
3-D Skiing	$24.95 now	$21.19
Creature Venture	$24.95 now	$21.19
Galaxy Space War I	$39.95 now	$33.99
Meteoroids in Space	$19.95 now	$16.99
Dragon Fire	$49.95 now	$42.49
Pool 1.5	$34.95 now	$29.69

NEC Green Screen Monitor	List $285.00	now $199.00
Zenith Green Screen Monitor	List $159.00	now $109.00
Ulysses	List $34.95	now $29.69
Speed/asm	List $39.95	now $33.89
Lisa 2.5	List $79.95	now $67.89
Dinosaurs	List $15.00	now $12.69

BMP Joystick

Here's a joytsick you won't want to be without. It's the only joystick we know about that allows you to change the axis with a flip of a switch. Only the highest quality parts are used.

A real bargain at

$49.95

Apple®

ORDER BY PHONE
800-344-4111 (outside Calif.)
800-692-4143 (inside Calif.)
209-992-5411 (foreign or local)

Apple is a registered trademark of Apple Computer, Inc.

HUNTINGTON COMPUTING
Post Office Box 787
Corcoran, California 93212

VISA

master charge

We take MasterCard, VISA or American Express (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.