



INTERNATIONAL
APPLE CORE
TM

APPLE RCHARD

Including CONTACT

IN THIS ISSUE:

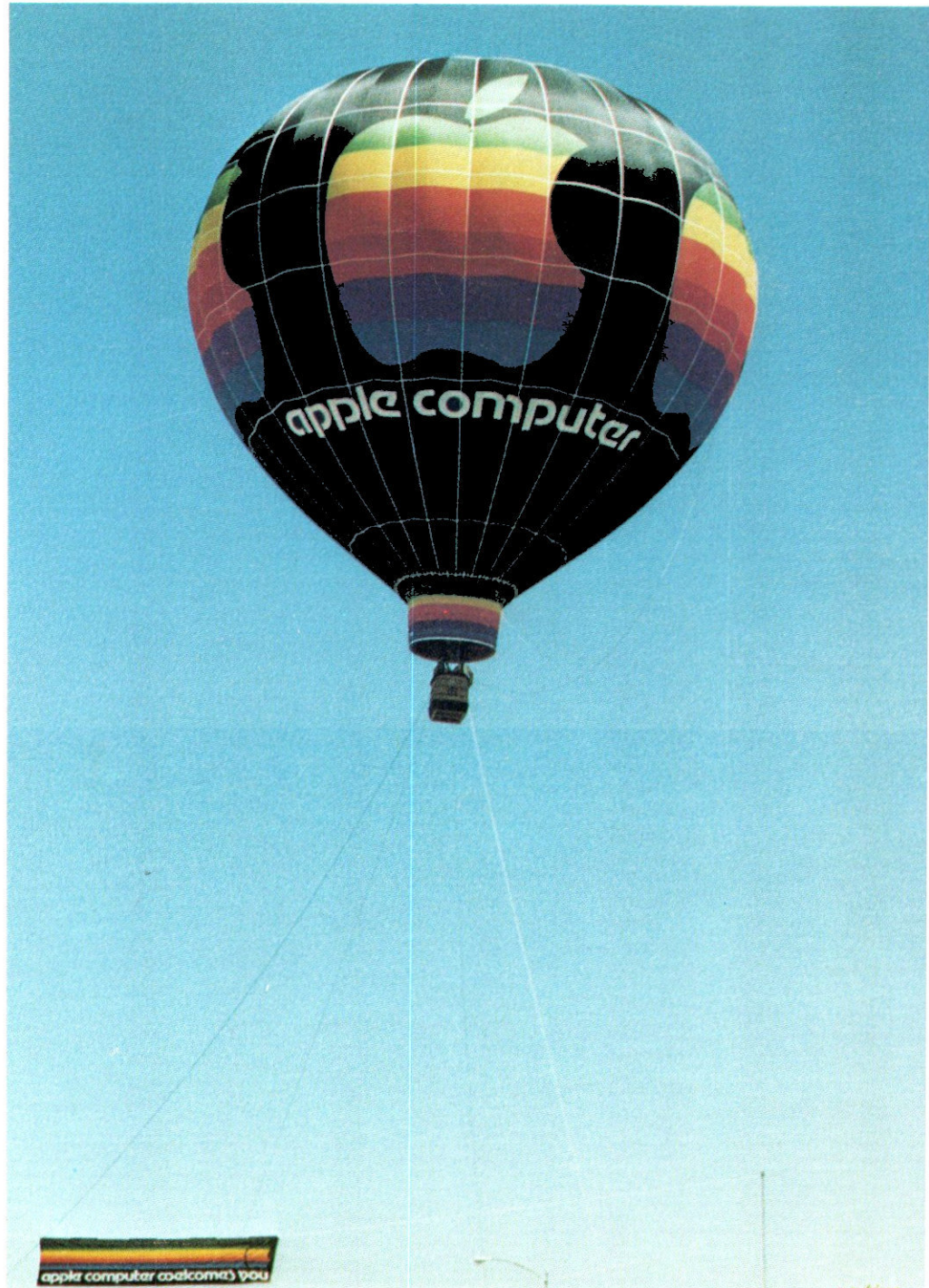
INTERFACING
BASIC TO RNTS
SEE PAGE 11

ALSO

APPLESOFT LIST
FORMATTER
ON PAGE 21

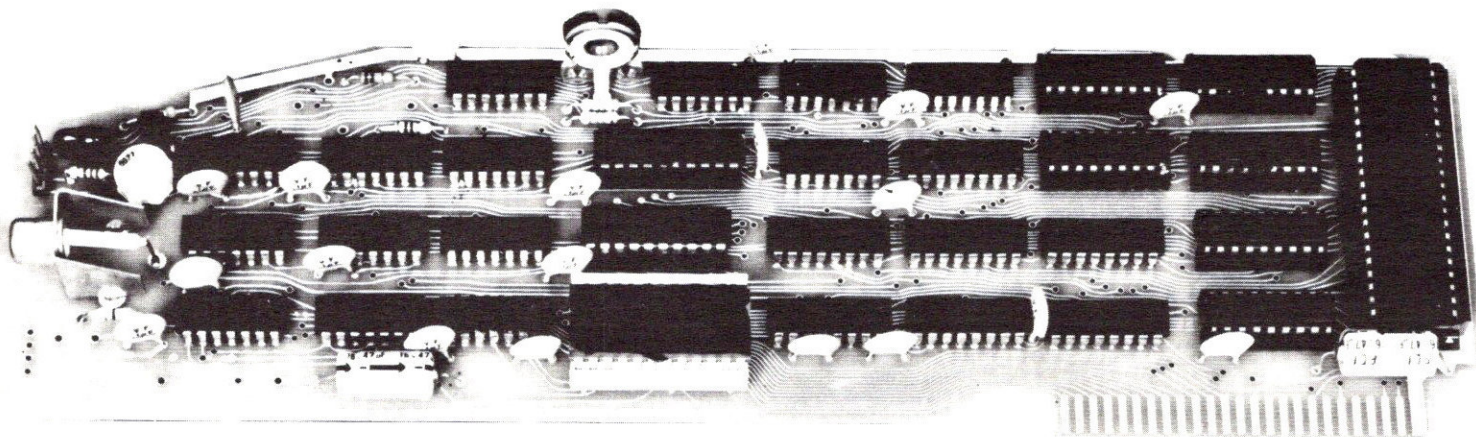
AND ALSO

INSIDE SILENTYPE
ON PAGE 43



WINTER 1980-81
\$3.50

SUP'R'TERMINAL™



SUP'R'TERMINAL IS AN 80 COLUMN BY 24 LINE PLUG-IN
COMPATIBLE BOARD FOR THE APPLE II COMPUTER

SPECIFICATIONS & FEATURES

- 80 Columns by 24 lines, upper and lower case; all 128 ASCII characters.
- Upper and Lower case data entry using the APPLE II keyboard.
- Includes an Upper and Lower case 5x8 dot matrix ASCII character set, and inverse alpha characters.
- Expands existing keyboard for more ASCII characters
- Character set can be user definable
- Includes VBC™ (video balance circuit) which enables the use of displaying 80 columns on an inexpensive 8 MHz CRT monitor
- Works with LEEDEX monitor (version 2.2) and other inexpensive CRT monitors
- Shift Lock Feature
- KEYPRESS function for PASCAL programs supplied
- Works with APPLE PASCAL and APPLE BASIC
- Incorporates PASCAL and BASIC control characters
- Follows protocols of PASCAL and BASIC operating systems
- ALL monitor-type escapes are valid
- Compatible with ALL APPLE II peripherals.
- Effective baud rate greater than 10,000; fast scrolling and clearing
- Can be used with APPLE II communication interface board to act as self contained terminal for time-sharing or other applications. Terminal program supplied when used with a D.C. Hayes micromodem.
- 3K bytes of bank switched static ram
- 2K bytes of ROM
- The only board with continuous direct memory mapped screened ram.
- The only board that interprets VTABS by firmware (version 2.2)
- The only board with an adjustable scrolling window.
- The only 80 column board that is synchronous with the APPLE II
- Fully programmable cursor
- Conversion program supplied to modify existing APPLESOFT programs to work with SUP'R'TERMINAL (automatically converts HOME, CALL-936 and VTABS) (version 1.0)
- Works with the new Easywriter and APPLE PI word processors.
- Uses less current on the +5V supply than any other 80 column board
- Works with CORVIS hard disc system

APPLE II is a trademark of APPLE Computer Co.
APPLE PI is a trademark of Programma International
Easywriter is a trademark of Information Unlimited
Micromodem is a trademark of D.C. Hayes

PATENT PENDING

M&R ENTERPRISES
P.O. BOX 61011, Sunnyvale, CA 94088

Mountain Computer makes more peripherals for the Apple Computer than Anybody.

and . . . a place to put them

INTROL X-10

Intelligent Home Controller for lights and appliances. Real-time schedules and energy conservation. Complete applications software package. Home security with random scheduler. Power usage accounting package for home energy cost control. No wiring required.

APPLE CLOCK

Real-time and date information. Interrupts permit Foreground/Background operation of two programs simultaneously. Battery back-up. Crystal-controlled for $\pm .001\%$ accuracy. On-board ROM for easy access from BASICS. Supports PASCAL. Time from one millisecond to one year.

SUPERTALKER SD200

Input/Output Speech Digitizer. Permits talking programs. I/O capability allows interactive programs with speech-prompted inputs. Use output for speech directed activities in business systems, announcements in a control-room, or sound effects in entertainment programs. Easy to use because input as well as output is under user control with special software operating system.

ROMWRITER

Program your own EPROMs. Create your own firmware. Programs 2K, 2716 5V EPROMs. Disk software package provides easy EPROM programming. EPROMs are verified after BURN. RUN your programs from on-board socket or install them on ROMPLUS+.

ROMPLUS+

More power for your system through firmware. Six sockets accept 2716 EPROMs or ROM equivalents. Six or any combination can be used at once. Scratch-pad RAM and two TTL connectors. Special 2K ROMs available for powerful system enhancement: Keyboard Filter ROM—COPYROM—Others coming soon.

MusicSystem

Sophistication previously available only on experimental mini and mainframe computer synthesizers. Digital instrumental music synthesizer system. 16 voices in stereo. Instrument definitions simulate the sound of real instruments—and more. Fully programmable waveforms. Envelope Control. Composition system—sheet music input using standard music notation. Chords and multi-part scoring up to 16 voices. A true instrument that anyone with an Apple can play.

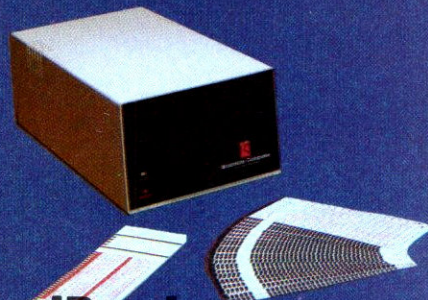
A/D+D/A

16 channels analog to digital input. 16 channels digital to analog output. Eight bit resolution. Super-fast 9μ sec. conversion time. Monitor and output to the real world. All on one card.



EXPANSION CHASSIS

By popular demand! Eight more slots for your Apple. Attractive sturdy enclosure. Its own heavy duty power supply. Easy to use. Address cards in Expansion Chassis the same way as in your Apple. Only one additional command to specify in Apple or in Expansion Chassis. Compatible with all Apple peripherals.



Card Reader

At last! An intelligent, high-quality device for data entry from user-marked cards. Implement BASIC programming, examination scoring, inventory maintenance and other applications requiring off-line data preparation for batch entry later. Connects to any computer having RS-232 interface. Software and cards are available for jobs in business, science and education.

MOUNTAIN COMPUTER has the most comprehensive line of Apple peripherals available. Anywhere. From anybody. We know the Apple inside and out and are committed to providing the most innovative and unique products to expand and enhance its capabilities and use. After all, we were the first company to make an Apple peripheral—except Apple Computer.

Available at Apple Dealers worldwide.



Mountain Computer
INCORPORATED

300 Harvey West Blvd., Santa Cruz, CA 95060
(408) 429-8600 TWX 910 598-4504

*Based on Manufacturer's published catalogs—Apr. 1980

Apple is a trademark of Apple Computer Inc.

DATA CAPTURE 4.0[©]

The most advanced and easiest to use telecommunications program for use with the MICROMODEM II™ or the Apple COMMUNICATIONS CARD™.

- Q. Will DATA CAPTURE 4.0 work with my Communications Card[®] and a modem?**
A. It makes using the Comm. Card almost as easy as using the Micromodem II.
- Q. Do I need an extra editor to prepare text for transmission to another computer?**
A. No. DATA CAPTURE 4.0 gives you control of the text buffer. You can use DATA CAPTURE 4.0 to create text.
- Q. Can I edit the text I have prepared?**
A. Yes. You can insert lines or delete any lines from the text.
- Q. How about text I have captured. Can I edit that?**
A. As easily as the text you have prepared yourself. You can delete any lines you don't want to print or save to a disk file. You can also insert lines into the text.
- Q. Just how much text can I capture with DATA CAPTURE 4.0?**
A. If the system with which you are communicating accepts a stop character, most use a Control S, you can capture an unlimited amount of text.
- Q. How does that work? And do I have to keep an eye on how much I have already captured?**
A. When the text buffer is full the stop character is output to the other system. Then DATA CAPTURE 4.0 writes what has been captured up to that point to a disk file. This is done automatically.
- Q. Then what happens?**
A. Control is returned to you and you can send the start character to the other system. This generally requires pressing any key, the RETURN key or a Control Q.
- Q. Are upper and lower case supported if I have a Lower Case Adapter?**
A. Yes. If you don't have the adapter an upper case only version is also provided on the diskette.
- Q. Do I need to have my printer card or Micromodem II[®] or Communications Card[®] in any special slot?**
A. No. All this is taken care of when you first run a short program to configure DATA CAPTURE 4.0 to your system. Then you don't have to be concerned with it again. If you move your cards around later you can reconfigure DATA CAPTURE 4.0.
- Q. Do I have to build a file on the other system to get it sent to my Apple?**
A. No. If the other system can list it you can capture it.
- Q. How easy is it to transmit text or data to another system?**
A. You can load the text or data into DATA CAPTURE 4.0 from the disk and transmit it. Or you can transmit what you have typed into DATA CAPTURE 4.0.
- Q. How can I be sure the other system receives what I send it?**
A. If the other system works in Full Duplex, it 'echoes' what you send it, then DATA CAPTURE 4.0 adjusts its sending speed to the other system and won't send the next character until it is sure the present one has been received. We call that 'Dynamic Sending Speed Adjustment'.
- Q. What if the other system works only in Half Duplex.**
A. A different sending routine is provided for use with Half Duplex systems.
- Q. What if I want to transmit a program to the other system?**
A. No problem. You make the program into a text file with a program that is provided with DATA CAPTURE 4.0, load it into DATA CAPTURE 4.0 and transmit it.

- Q. What type files can I read and save with DATA CAPTURE 4.0?**
A. Any Apple DOS sequential text file. You can create and edit EXEC files, send or receive VISICALC[®] data files, send or receive text files created with any editor that uses text files.
- Q. Can I leave DATA CAPTURE 4.0 running on my Apple at home and use it from another system?**
A. Yes. If you are using the Micromodem II[®] you can call DATA CAPTURE 4.0 from another system. This is handy if you are at work and want to transmit something to your unattended Apple at home.
- Q. Where can I buy DATA CAPTURE 4.0?**
A. Your local Apple dealer. If he doesn't have it ask him to order it. Or if you can't wait order it directly from Southeastern Software. The price is \$65.00. To order the Dan Paymar Lower Case Adapter add \$64.95 and include the serial number of your Apple.
- Q. If I order it directly how can I pay for it?**
A. We accept Master Charge, Visa or your personal check. You will get your order shipped within 3 working days of when we receive it no matter how you pay for it. Send your order to us at the address shown or call either of the numbers in this advertisement. You can call anytime of day, evening or Saturdays.
- Q. I bought DATA CAPTURE 3.0 and DATA CAPTURE 4.0 sounds so good I want this version. What do I do to upgrade?**
A. Send us your original DATA CAPTURE 3.0 diskette and documentation, the \$35.00 price difference and \$2.50 for postage and handling. We will send you DATA CAPTURE 4.0 within 3 working days of receiving your order.
- Q. What kind of support can I expect after I buy it?**
A. If you have bought from Southeastern Software in the past you know we are always ready to answer any questions about our products or how to use them.

Requires DISK II[®], Applesoft II[®] and 48K of Memory

DATA CAPTURE 4.0[©]

Copyright © 1980-Southeastern Software

* Apple[®], Apple II Plus[®], Disk II[®] and APPLESOFT II[®] are trademarks of Apple Computer Company.

* Micromodem II[®] is a trademark of D.C. Hayes Associates, Inc.

* Visicalc[®] Copyright by Software Arts, Inc.

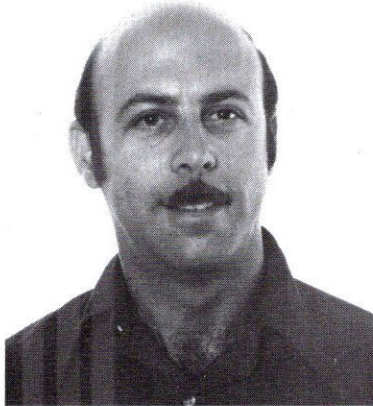


We welcome your personal check. We also accept Visa and Master Charge.

Southeastern Software

Dept. AO
6414 Derbyshire Drive • New Orleans, LA 70126
504/246-8438 504/246-7937

PRESIDENT'S CORNER



Like Apple Computer Inc., the International Apple Core's growth has been phenomenal. In one year our membership is over 180 clubs, 23 associates, and 18 sponsors. The IAC now represents over 18,000 Apple owners.

There is so much we would like to accomplish now, for you, our members, but it takes time in a volunteer organization to find people who are willing to help. In time, as we increase our financial foothold, this will be easier by hiring individuals to perform the work. Financial stability is the current goal of the Board of Directors and officers. The IAC funding, at this time, comes almost entirely from the sales of **THE APPLE ORCHARD**. If you haven't subscribed yet, you will find a form in this issue — remember your subscription helps support the IAC and you get a great publication concerning the Apple Computer.

At a recent Board of Directors meeting, the date and place for the 1981 **ANNUAL GENERAL MEETING** was set, and will be held in Chicago, Ill. on May 2nd & 3rd in conjunc-

tion with NCC. More information as to place and times for the meeting and seminars will be sent to all clubs as plans are finalized. All member clubs should start to plan on sending, if they wish, a representative to this meeting — remember your club has a vote as to the direction of the IAC — we want your input.

Like in 1980, we plan to have technical seminars from Apple experts at the meeting. They will be video taped and be available to all clubs for showing. As we obtain confirmation from these experts, we will inform you.

One of the items that came out of the Board meeting was the implementation of a new **BULLETIN** which we will publish every month and send to all members. The purpose of this is to advise you of what the IAC is doing, timely announcements, and information that can not wait for our quarterly publication.

*Ken Silverman, President
International Apple Core*

INTERNATIONAL APPLE CORE SPONSORING MEMBERS

I.A.C. sponsors are a special breed. They are the organizations who along with our advertisers, contribute to and support many I.A.C. activities. In addition, they will provide us with application notes concerning their products — notes that will benefit users by showing new and different ways to utilize their products or production/software modifications that have been made to upgrade their product. When considering a software or product purchase, we request that they be given special consideration.

Those organizations that would like to become sponsors or who would like additional information about the benefits and advantages of becoming a sponsoring member are urged to contact Michael Weinstock, Vice-President, International Apple Core, P.O. Box 976, Daly City, CA 94017.

A list of sponsoring members, current through the first of September, 1980, appears below.

Apple Computer, Inc.
10260 Bandle Drive
Cupertino, CA 95014
(408) 996-1010

Axiom Corp.
5932 San Fernando Rd.
Glendale, CA 91202
(213) 245-9244

Bell & Howell, Inc.
7100 McCormick Road
Chicago, IL 60645
(312) 262-1600

Compuserve - Micronet
5000 Arlington Centre Blvd.
Columbus, OH 43220
(614) 457-8600

Custom Computing System,
Inc.
122 2nd Ave. N.
Saskatoon, Sask.,
Canada S7K 2B2

Epson America, Inc.
23844 Hawthorne Blvd.
Torrance, CA 90505
(213) 378-2220

Hayes Microcomputer
Products
5835 Peachtree Corners East
Norcross, GA 30092
(404) 449-8791

Interactive Structures, Inc.
P.O. Box 404
Bala Cynwyd, PA 19004
(215) 667-1713

Malibu Electronics Corp.
2301 Townsgate Road
Westlake Village, CA 91361
(805) 496-1990

Nestar Systems, Inc.
430 Sherman Ave.
Palo Alto, CA 94306
(415) 327-0125

Okidata Corporation
111 Gaither Drive
Mt. Laurel, NJ 08054
(609) 235-2600

Peripherals Unlimited
3450 E. Spring St., Suite 206
Long Beach, CA 90806
(213) 595-6858

Programma International,
Inc.
3400 Wilshire Blvd.
Los Angeles, CA 90016
(213) 384-1116

Siro-tech Software Products
6 Main St.
Ogdenstring, NY 13669
(315) 393-5151

Source Telecomputing Corp.
1616 Anderson Road
McLean, VA 22102
(703) 821-6660

Syntauri Ltd.
3506 Waverly St.
Palo Alto, CA 94306
(415) 494-1017

Verbatim Corp.
323 Soquel Way
Sunnyvale, CA 94086
(408) 245-4400

Xerox Retail Markets Div.
L-140, 24500 Industrial Blvd.
Hayward, CA 94545
(415) 786-5205



THE APPLE ORCHARD

Vol. 1, No. 3

Winter 1980-81

Val J. Golding Editor
 Ken Silverman Associate Editor
 Jim Hoyt Associate Editor
 Kathryn Hallgrimson Assistant to the Editor
 David B. Garson Review Editor
 Larry Danielson Shipping Manager
 Vic Warren Design Cover Design
 Brian Bennett Cover Photo
 Buck Evans Apple Orchard Postcard
 Grawin Publications Production



ADVERTISING REPRESENTATIVES
 Dilithium Press
 P.O. Box 606
 Beaverton, OR 97075
 (503) 243-1160

SUBSCRIPTIONS
 Apple Orchard Subscriptions
 P.O. Box 2227
 Seattle, WA 98111
 \$10/year — Published Quarterly

Vol. 1, No. 3

Winter 1980-81

Entire Contents Copyright© 1980
 by **International Apple Corps**
 P.O. Box 976, Daly City, CA 95017

IN THIS ISSUE

President's Corner	Ken Silverman	3
IAC Sponsoring Members		3
PRINT FRE(ed)	Val J. Golding	7
Select One	The Editor	9
Fruit of the Earth	Kyrene Gould	9
A BASIC to Machine Language Interface	Pete Rowe	11
Applesoft Program Listing Formatter	Robert C. Clardy	21
Locations of Interest to Pascal and 6502 Users	Randy Hyde	28
Creativity Life: review	David B. Garson	29
A Machine Language Address Calculator	Russ Lavalee	31
Some Notes about the UCSD Assembler	Ron Haines	35
Float, Float, Float your Point	Guy A. Lyle	37
CONTACT Section		
Inside the Silentye Firmware	J.D. Eisenberg	43
IAC Member Club Additions		49
Binary to Decimal Shortcut	Steve Wozniak	52
Tabbing with Apple Peripherals	John Crossley	55
Apple Writer and the Telephone	Jim Hoyt	56
Dow Jones News and Quotes Reporter		57
Converting Strings to Numeric Variables	Jo & Charlie Kellner	59
Pascal Run-Time Errors	Jo Kellner	62
Saving and Loading Arrays in Applesoft		63
The Keypress Function		65
The Tax Planner		66
Applewriter & DOS Text Files		68
POKE Salad	Tom Brown	69
Using USR	Frank Evans	75
Space War: review	David B. Garson	79
Advertiser's Index		80

INTERNATIONAL APPLE CORE

Officers

Ken Silverman	President	(415) 878-9171
Michael Weinstock	Vice-President	(516) 360-0988
Dave Gordon	Treasurer	(213) 384-0579
Joe Budge	Secretary	(919) 489-4282

Regional Directors

Jon R. Lawrence	(north)	(313) 534-2433
Harlan G. Felt	(north)	(408) 866-1733
Jerry Vitt	(south)	(214) 369-7660
Scott Knaster	(south)	(303) 355-2379
Bernie Urban	(east)	(301) 229-3458
Tony Cerreta	(east)	(914) 636-3417
Joe Alinsky	(west)	(213) 703-1894
Fred Wilkinson	(west)	(415) 585-2240

International Directors

Roger Keating PO Box 448, Double Bay 2028, NSW, Australia
 Auby Mandell 409 Queen St. W. Toronto, Ont. Canada M5V 2A5
 Wolfgang Dederichs Auf Drenhausen 2 4320 Hattingen, West
 Germany

Committee Chairmen

Anotes	John Shanes	(804) 746-2711
Apple Fair	Bob Ramsdell	(617) 742-6100
Apple Orchard	Val J. Golding	(206) 932-6588
Constitution & Bylaws	Ken Silverman	(415) 878-9171
Education SIG	Ted Perry	(916) 961-7776
Ham Radio SIG	James E. Hassler, WB7TRQ	(307) 632-4934
Handicapped SIG	Bernie Urban	(301) 229-3458
I.A.C. Software	Neil Lipson	(215) 356-6183
Legal SIG	Butch Clayton	(803) 884-5370
Medical SIG	Dr. Larry L Stoneburner	(714) 953-9151
Newsletter Exchange	David Alpert	(312) 295-6078
Newsletter Library	Maj. Terry N. Taylor	(213) 372-4134
New Club Assistance	Randy Fields	(415) 775-7965
Standards	Mark Robbins	(303) 750-5813
Telecommunications	Craig Vaughn	

Why not kill two birds with one stone?

If you have an Apple* and you want to interface it with parallel and serial devices, we have a board for you that will do both. It's the AIO.TM

Serial Interface.

The RS-232 standard assures maximum compatibility with a variety of serial devices. For example, with the AIO you can connect your Apple* to a video terminal to get 80 characters per line instead of 40, a modem to use time-sharing services, or a printer for hard copy. The serial interface is software programmable, features three handshaking lines, and includes a rotary switch to select from 7 standard baud rates. On-board firmware provides a powerful driver routine so you won't need to write any software to utilize the interface.

Parallel Interface.

This interface can be used to connect your Apple* to a variety of parallel printers. The programmable I/O ports have enough lines to handle two printers simultaneously with handshaking control. The users manual includes a software listing for controlling parallel printers or, if you prefer, a parallel driver routine is available in firmware as an option. And printing is only one application for this general purpose parallel interface.

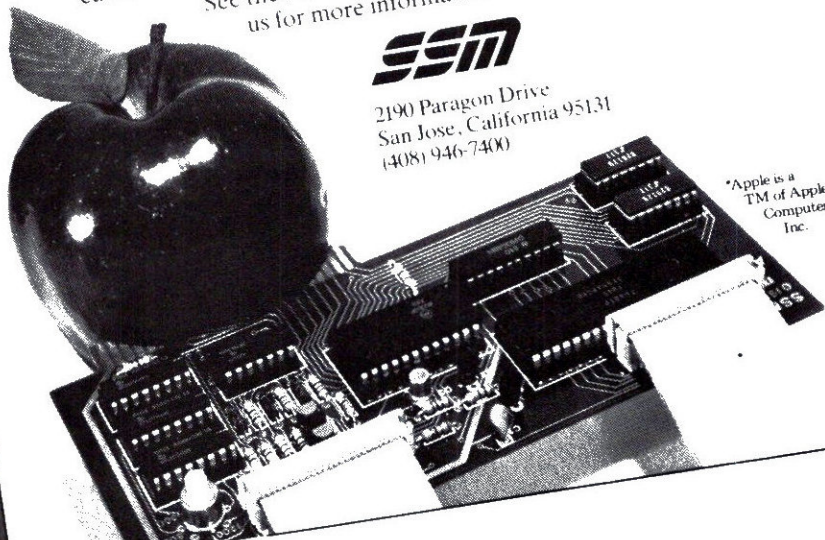
Two boards in one.

The AIO is the only board on the market that can interface the Apple to both serial and parallel devices. It can even do both at the same time. That's the kind of innovative design and solid value that's been going into SSM products since the beginning of personal computing. The AIO comes complete with serial PROM's, serial and parallel cables, and complete documentation including software listings. See the AIO at your local computer store or contact us for more information.



2190 Paragon Drive
San Jose, California 95131
(408) 946-7400

*Apple is a
TM of Apple
Computers,
Inc.



Maybe we can save you a call.

Many people have called with the same questions about the AIO. We'll answer those and a few more here.

Q: Does the AIO have hardware handshaking?

A: Yes. The serial port accommodates 3 types—RTS, CTS, and DCD. The parallel port handles ACK, ACK, BSY, STB, and STB.

Q: What equipment can be used with the AIO?

A: A partial list of devices that have actually been tested with the AIO includes: IDS 440 Paper Tiger, Centronics 779, Qume Sprint 5, NEC Spinwriter, Comprint, Heathkit H14, IDS 125, IDS 225, Hazeltine 1500, Lear Siegler ADM-3, DTC 300, AJ 841.

Q: Does the AIO work with Pascal?

A: Yes. The current AIO serial firmware works great with Pascal. If you want to run the parallel port, or both the serial and parallel ports with Pascal, order our "Pascal Patcher Disk."

Q: What kind of firmware option is available for the parallel interface?

A: Two PROM's that the user installs on the AIO card in place of the Serial Firmware PROM's provide: Variable margins, Variable page length, Variable indentations, and Auto-line-feed on carriage return.

Q: How do I interface my new printer to my Apple using my AIO card?

A: Interconnection diagrams for many popular printers and other devices are contained in the AIO Manual. If your printer is not mentioned, please contact SSM's Technical Support Dept. and they will help you with the proper connections.

Q: I want to use my Apple as a dumb terminal with a modem on a timesharing service like The Source. Can I do that with the AIO?

A: Yes. A "Dumb Terminal Routine" is listed in the AIO Manual. It provides for full and half duplex, and also checks for presence of a carrier.

Q: What length cables are provided?

A: For the serial port, a 12 inch ribbon cable with a DB-25 socket on the user end is supplied. For the parallel port, a 72 inch ribbon cable with an unterminated user end is provided. Other cables are available on special volume orders.

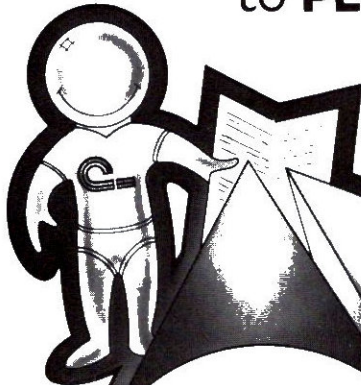
The AIO is just one of several boards for the Apple that SSM will be introducing over the next year. We are also receptive to developing products to meet special OEM requirements. So please contact us if you have a need and there is nothing available to meet it.



SSM Microcomputer Products
2190 Paragon Drive
San Jose, California 95131
(408) 946-7400

MR. RAINBOW

presents our valuable free catalog (over 100 pages). He **PROMPTS** you to **PEEK** at the latest collection of software and hardware products for your **APPLE II™**



A STELLAR TREK

the definitive Hi-Res color version of the classic Startrek game. Three different Klingon opponents. Many command prerogatives from use of weapons to repair of damages. Needs 48K Applesoft ROM.

Disk... **\$24.95**

VERSAWRITER II

A drawing tablet, simply plugs into your game I/O port. Trace, draw, design, or color any type of graphic. Adds words to pictures. Creates schematics. Computes Distance/Area of any figure. New - fill any area on the screen in seconds with over 100 different and distinct colors. Needs 32K Applesoft ROM and disk drive. A bargain at...

\$249.95

BOWLING DATA SYSTEM

This data management program provides accurate record keeping and report generation for bowling leagues of up to 40 teams with 6 bowlers per team. Needs 80-column printer, 32K Applesoft ROM.

Disk... **\$79.95**

SUPER SOUND

Musical rhythms, gunshots, sirens, laser blasts, explosions... add these and many more exciting sounds to your Apple. Use them in your programs, or create your own SUPER SOUNDS. Needs 16K Applesoft.

Have a blast for only

\$12.95... Tape

\$16.95... Disk

ADD \$2.00 U.S. \$10.00 FOREIGN FOR SHIPPING
CALIFORNIA RESIDENTS ADD 6% SALES TAX

Don't see what you want here, then write or call today for your free catalog. We're saving one just for you.

Visa / Mastercharge welcome.



Open Tues. - Sun.

GARDEN PLAZA SHOPPING CENTER
9719 RESEDA BOULEVARD DEPT. 12A0
NORTHRIDGE, CALIFORNIA 91324
PHONE (213) 349-5560

PRINT FRE(ed)

by Val J. Golding

In Nibble Magazine number 6, editor Mike Harvey wrote an eloquent and thought-provoking editorial on the subject of bootlegging and protected programs, which we suggest you read. Nibble's address may be found in their advertisement on Page 10.

Mike wrote of having heard that a user group had "broken" the Visicalc "non-copy" locks and was making copies of Visicalc for its members, an appalling thought. All IAC member clubs, in making their application to join, agree they will not be a party to the copying of copyrighted programs. Unfortunately, this is a difficult area to police, and word of mouth about illegal copies, which we too have heard, is not sufficient grounds for the expulsion of a member club.

This entire matter is regrettable and has already taken its toll in the

form of higher prices and "protected", harder to copy, software. It appears that software manufacturers generally accept that copying is inevitable, which has resulted in substantial increases in purchase prices (which must be amortized over a very short period to recoup costs before many illegal copies get out) and an ever-increasing number of software products relying on numerous different protection schemes.

The so-called "non copyable" diskettes can, of course, be copied by anyone endowed with sufficient skill and expertise in Apple DOS, but by and large they reduce the numbers of software thefts. However, in many cases, protected diskettes place an unfair burden on the legitimate software purchaser, in the form of lengthy delays in obtaining replacement diskettes,

and the further cost of paying a service charge for same, sometimes at a much more than nominal figure.

Again, this situation is regrettable, but it is also something we have brought upon ourselves by knowingly *accepting* and using, from whatever sources, bootlegged programs. If the situation gets too far out of hand, the decreasing returns to the talented programmers who have created such excellent pieces of software as Visicalc and Program Line Editor will turn their talents to other fields, and we shall be the poorer for it.

It is not too late to stem, if not turn, the tide. When you accept and use a piece of pirated software, *you* are an accessory after the fact. It makes no difference whether we are talking about a \$9.95 game or a \$700 Controller. Theft is *theft*, and we can not mince words. You have paid how much — \$1000, \$2000, \$3000 — for your Apple II, one of the world's finest and most powerful microcomputers. Is it worth the resultant damage to the industry to save a few bucks?

If you have an Apple, Pet or TRS-80 microcomputer,* you can have fantasy at your fingertips with Epyx computer games from Automated Simulations.

Like me, you're probably really into games, all sorts of games. But an Epyx game is more than a game — it's an experience, and it's a chance to use your computer for something other than work. The great thing about Epyx games is that you have a choice. Whether you're a beginner or an expert, you can find games that are easy to learn. Challenging. Fun to play for twenty minutes or

hours at a time. You can play these games over and over, because you're constantly trying new tactics and strategies.

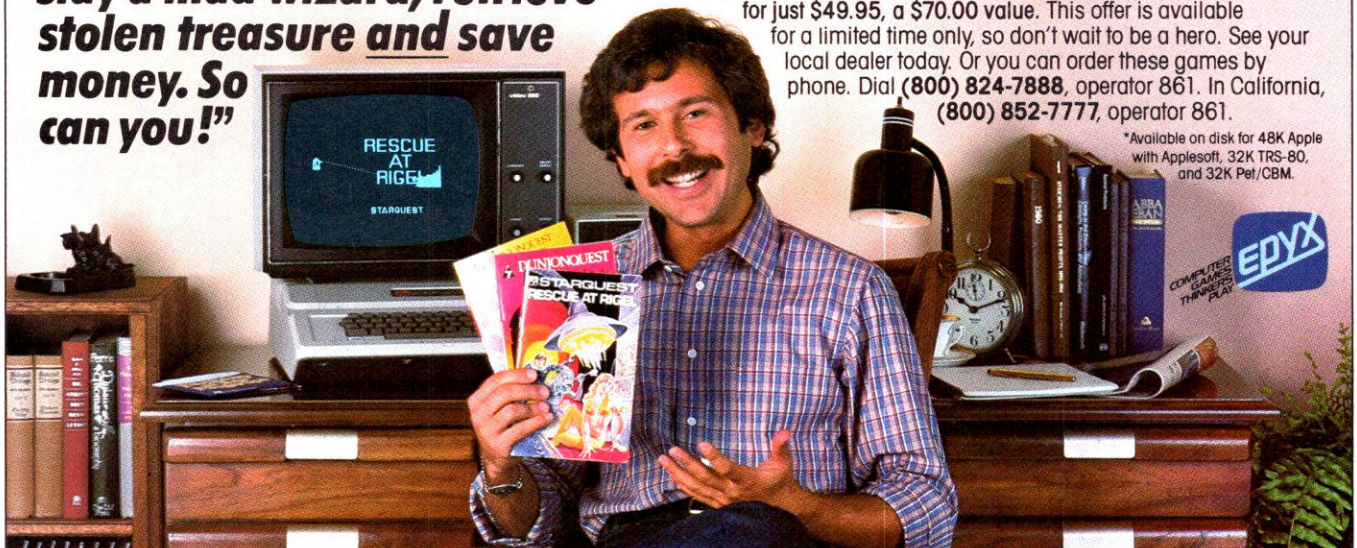
I've already entered and re-entered a world of monsters and misfits, demons and dwarves, trials, tribulations and treasures with a game called "Temple of Apshai." Now it's my chance to have fun with three more games from Automated Simulations... and I can save money, too!

With "Datestones of Ryn" and "Morloc's Tower," I get to escape from booby-trapped mazes, find more treasures and zap more monsters. And with "Rescue at Rigel," I get to outwit the nasty High Tollah and free 10 prisoners.

Automated Simulations has a special offer on "Datestones of Ryn," "Morloc's Tower" and "Rescue at Rigel." Buy all three for just \$49.95, a \$70.00 value. This offer is available for a limited time only, so don't wait to be a hero. See your local dealer today. Or you can order these games by phone. Dial (800) 824-7888, operator 861. In California, (800) 852-7777, operator 861.

*Available on disk for 48K Apple with Applesoft, 32K TRS-80, and 32K Pet/CBM.

**"I can rescue ten prisoners
slay a mad wizard, retrieve
stolen treasure and save
money. So
can you!"**



It's Almost Obscene...



The tricks our IBMS software can make your Apple* do!

The small businessman has never had it so good, or so easy. Because now there's our **Interactive Business Management System (IBMS)** . . . which lets your micro-computer perform like a larger unit, so you can mind, monitor and manage every aspect of your business accounting.

A Full System

While it's extremely easy to use, IBMS is a full system to handle the full job. The ten program modules can generate everything from the original invoice to the final profit/loss statements, plus many peripheral operations. The special Menu includes: System Start-up. Accounts Receivable. Accounts Payable. Perpetual Inventory. Payroll. Fixed Assets. General Ledger. Plus Mailing Labels, and an Appointments Calendar.

Save Maximum Time

Since IBMS is a totally interactive system, multiple-entering of data is eliminated. Make an entry in one area and it automatically updates all concerned areas! No duplication of effort, no wasted time, no problems.

Proven. And then some.

It took 3 years to develop IBMS, including shake-down and on-site testing. As a result, it's reliable and versatile and its documentation is thorough and easily understandable. No wonder we consider it 5 years ahead of anything else available to the Apple II user.

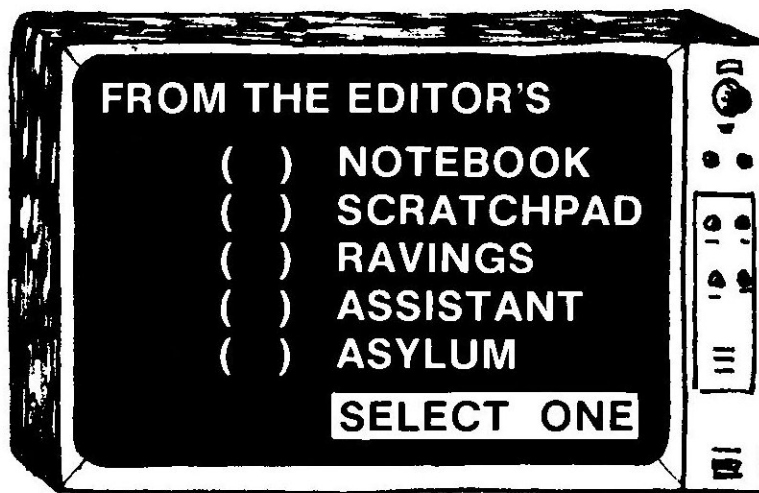
Introductory Offer

The complete IBMS software package, on mini-floppy disks, documentation, and the backing of Programma International, Inc. is offered for a limited time at the **Introductory Price of \$1495.00**. You'll be amazed how it can satisfy you . . . by saving you time, effort, money and employee growth.



PROGRAMMA
PROGRAMMA INTERNATIONAL, INC.
2908 N. Naomi Street, Burbank, Ca 91504
(213) 954-0240

* Apple is a trademark of Apple Computer, Inc.



By the Editor

Select One. The title of this feature was well chosen. This is the section to which the reader may turn and obtain a tad more information about the featured stories in this magazine, a few tantalizing hints of the valuable material inside.

Our first task normally is to dig into the feature story and then move on to some of the interesting articles, but this time the reader will have to wait for a moment as we take time to talk a bit about the International Apple Core, its publication Apple Orchard, and to chide, in fact, scold, some of the IAC member clubs.

The Orchard is the official publication of the IAC, a group formed to serve the needs of Apple Computer user groups world wide, and representing near 20,000 individual Apple users, a most respectable number, and a fair share of all Apples sold. The IAC offers a multitude of services to its membership, much of it either free or on a cost basis. Major funding for IAC activities is through the sales of advertising and bulk dealer copies of the Orchard.

Producing the Orchard requires a constant flow of editorial material, which is generated from three primary sources: material from Apple Computer, Inc., for the Contact section, reprints of particularly significant articles from the various club newsletters, and fresh stories and programs submitted by individuals, and user groups on behalf of their members. A page rate of \$25 per printed page is paid upon request.

As we are responsible to provide services to our member clubs, so too are they obligated to furnish the Orchard with material, and it is to this subject we must now address ourselves. *There can not be another issue of the Orchard without a vast influx of suitable material, and as you read this, the deadline is already upon us.* A small effort by each member club would provide hundreds of pages worth of stories in a matter of days.

Featured in this Winter issue of the Apple Orchard is **Pete Rowe's** "A BASIC to Machine Language Interface". Pete's article and four programs show how to use DOS's RWTS routines from both Integer and Applesoft Basics, a quite simple procedure, once you understand how it all fits together.

The lead story in Apple Computer's Contact section by **J.D. Eisenberg** and **Andy Hertzfeld**, "Inside the Silentype Firmware", is similarly interesting, just what you need to know to perform some fancy stunts with your Silentype printer.

Bob Clardy, along with a host of others from Apple Pugetsound have come up with some neat modifications to **Mark Capella's** "Applesoft Program Listing Formatter" which previously appeared in Call -Apple, while **Frank Evans** from the Apple Portland group presents some noteworthy instructional information in "Using USR", the little known Applesoft function, and **Randy Hyde** of the Original Apple Core in Los Angeles has contributed "Locations of Interest

to Pascal and 6502 Users". In addition, our overseas clubs are represented by **Ron Haines** from the New South Wales, Australia group with "Notes About the UCSD Assembler".

Rounding out the Contact section, "**Woz**", one of the original partners of Apple Computer, Inc., offers a short Assembly Language "Binary to Decimal Shortcut", **Jo** and **Charlie Kellner** demonstrate "Converting Strings to Numeric Variables" in Pascal, and other stories feature the Dow-Jones package, "Saving and Loading Arrays in Applesoft" and a collection of other subjects of interest.

FRUIT OF THE EARTH

by Kyrene Gould
BL Systems, LTD.
Birmingham, England

Slowly the dusky dawn lists
and scrolling upwards,
gently reveals the bruised crimson
of the ever present Apple.

With speed of ten, it rises,
to output its knowing silicon rays
into the darkened atmosphere
of minds.

Manipulable, yet master,
it saves those who would run,
and works with careful control,
characterizing their output,
and filing for future reference,
their mode and format.

Thus daily, it continues,
monitoring, and driving forward
manipulating, dating and
translating.

Ever constant in its bitten,
colored hue.

Ever present at the core of all
memory
now and in future time.

It computes unceasingly, until,
with day's demise,
its lifetime is broken with flick
of switch,
and it hangs,
once more lifeless,
until tomorrow's dawn.

"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:

- Numeric Keypad Construction Lab
- 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!

No. 4

nibble
Box 325, Lincoln, MA. 01773 (617) 259-9710

I'll try nibble!
Enclosed is my \$15 (for one year).

check **money order**

(Please allow 4 to 6 weeks for delivery of 1st issue)
BACK ISSUES of NIBBLE are available for
\$2.00 + .50 postage and handling.

Name _____

Address _____

City _____

State _____ Zip _____

NOTE:
First Class or Air Mail is required for all APO, FPO and all foreign addresses with the following additional amounts
— USA, Canada, Mexico, APO, FPO \$7.50
— Central and South America \$9.00
— Europe \$12.00
— Asia and elsewhere \$15.00

© 1980 by MICRO-SPARC, INC., Lincoln, Mass. 01773. All rights reserved.
*Apple II is a registered trademark of Apple Computer Company.

A BASIC TO MACHINE LANGUAGE INTERFACE

by Pete Rowe
Computer-Advanced Ideas, Berkeley

The Integer and Applesoft Basic languages provide good working environments for creating structured, readable code. But for some applications, neither BASICs can provide a needed function or fast execution speed. I am not beginning to make a case for developing all programs in a more primitive language (on the Apple, 6502 machine or assembly code). But I would like to convince you to use either Integer or Applesoft Basic and machine code together to create a more powerful working environment than either of the languages could provide separately.

Specifically, Integer Basic or Applesoft is well suited for controlling input and output. With the INPUT and PRINT statements, a programmer can easily and quickly code the interactive processes for a given application. But using either BASIC, he could NOT, for example, play a tune using the Apple's speaker or directly access user-specified, random sectors on an Apple II diskette without employing a machine language routine. To reduce all coding to machine language would place an unnecessary burden on the programmer and could result in excessive development time and hard-to-modify software. My proposal: Join an interactive BASIC language shell to a machine language, computational viscera.

To invoke machine code from

either BASIC is easy: Use the CALL statement. In many cases we also need to pass numeric or string data to and from the machine code. In the case of sound generation, we need to pass at least pitch and duration values. There are perhaps three classifiable methods of passing data between BASIC and machine code:

- (A) POKEing values directly into memory. Example:

```
100 INPUT P,D:POKE O,P:  
POKE 1,D:CALL 768
```

Refer to page 45 of the Apple II "Red" Reference Manual, Jan '78

- (B) Passing values in fixed locations in the BASIC variable table:

```
0 PITCH=0:DUR=0:REM  
MANDATORY FIRST VARIABLE ASSIGNMENT  
100 INPUT PITCH,DUR:  
CALL 768
```

- (C) Passing values just by name in the BASIC variable table:

```
100 INPUT PITCH,DUR:  
CALL 768:REM NO PREVIOUS  
ASSIGNMENTS NECESSARY
```

(A) and (B) above are examples of passing values by location. POKEing, (A), is the simplest method, but is vulnerable to code changes (moving or altering the machine code could change all of the BASIC reference address) and does not

promote very readable software. Passing values in a fixed location is an improvement, but places a constraint on the programmer to assign all the value-passing variables first and in a specified order. Programmer's Aid #1 uses this method for functions one and eight: Re-number and High-Resolution Graphics. If the programmer wishes to add more than one machine code routine to his BASIC program, using method (B), he would have to modify the second and all successive machine code routines to accept variable placement in different positions in the BASIC variable table.

Method (C) is my recommendation: Assign BASIC variables, numeric or string, as needed and at any time. Therefore, the variables can appear anywhere in the BASIC variable table, which is truly passing values by name and not location. This method offers two additional features: (1) Variables not needed for passing data to the machine code need not be assigned by the BASIC program. The machine code can create the missing BASIC variable and assign a value before returning to the BASIC program. And (2), the bulk of the overhead to find and create a BASIC variable is procured from the host language. Namely, the same code that creates and assigns variables during the execution of a BASIC program is used by the machine language code.

The following four source listings demonstrate two functions each in Integer Basic and Applesoft. The first routine supplies the missing CHR\$ function for Integer Basic. The CHR FP routine that follows is included to demonstrate variable passing in Applesoft even though Applesoft already contains a CHR\$ function.

```

3 *
4 *
5 * CHR FUNCTION FOR INTEGER BASIC

6 *
7 * PETE ROME          JULY 1980

8 *
9 *  COMPUTER-ADVANCED IDEAS
10 *
11 *
12 *      AST 25
13 *
14 *
15! *      ORG $300
16 *      OBJ $6300
17 *
18 *
19 * IBASIC FIND POINTER TO LOCATIO
    N OF VARIABLE VALUE
20 *
21 *
22 FIND EQU $E679
23 *
24 *
25 * VARIABLE POINTER AND IBASIC NO
    UNSTK
26 *
27 *
28 VARPNT EQU $6F
29 MOUNSTKH EQU $97
30 *
31 *
32 * INTEGER BASIC PAGE ZERO LOCATIO
    NS TO BE SAVED & RESTORED
33 *
34 *
35 VERBNOW EQU $D6
36 PRINOW EQU $D7
37 PXL EQU $E0
    
```

```

38 PXH EQU $E1
39 *
40 *
41 START TXA ;SAVE X-REG
42 PHA
43 *
44 * SAVE IBASIC ZPAGE POINTERS
45 *
46 LDA VERBNOW
47 PHA
48 LDA PRINOW
49 PHA
50 LDA PXL
51 PHA
52 LDA PXH
53 PHA
54 *
55 *
56 JSR FINDCHR LOCATE CHR
    VARIABLE
57 *
58 *
59 PHA ;SAVE IT
60 *
61 *
62 JSR FINDCHRSTR LOCATE C
    HR$ VARIABLE
63 *
64 *
65 PLA ;RETRIEVE CHR VALUE
66 *
67 *
68 STA ($6F),Y STORE AT CH
    R$ VALUE LOCATION
69 *
70 *
71 *
72 *
73 RESTORE PLA ;RESTORE IBASIC ZPA
    GE VALUES
74 STA PXH
75 PLA
76 STA PXL
77 PLA
78 STA PRINOW
79 PLA
80 STA VERBNOW
81 *
82 *
83 PLA
84 TAX ;RESTORE X-REG
85 *
86 *
87 AST 25
88 *
89 *
    
```

```

90 * FIND CHR AND FIND CHR$ SUBROUT
    INES
91 *
92 *
93 AST 25
94 *
95 FINDCHR JSR FINDVAR
96 ASC "CHR" (NSB
    ON)
97 HEX 00
98 *
99 *
100 FINDCHRSTR JSR FINDVAR
101 ASC "CHR"
102 HEX 4000 '$'
103 *
104 *
105 AST 25
106 *
107 *
108 * FIND VARIABLE, NAME STORED AFT
    ER
109 * JSR. VALUE OF VARIABLE RETURN
    ED IN A-REG.
110 *
111 *
112 AST 25
113 *
114 FINDVAR PLA
115 STA PXL
116 PLA
117 STA PXH NAME POINTE
    R ON STACK - 1
118 *
119 * PX=PX+1
120 *
121 INC PXL
122 BNE FINDV
123 INC PXH
124 *
125 * IBASIC FIND LOCATES VARIABLE N
    AME POINTED TO BY PXL,PXH
126 * FIND RETURNS VALUE POINTER IN
    MOUNSTK,H
127 *
128 FINDV JSR FIND
129 *
130 * MOVE HIGH BYTE OF ADDRESS FROM
    MOUNSTKH TO VARPNT+1
131 *
132 LDA MOUNSTKH IBASIC VAR
    PNTH
133 STA VARPNT+1
134 LDY #0
135 LDA (VARPNT),Y LOAD VAR
    IABLE VALUE
136 RTS
    
```

```

;
* 300, 34E

300- BA 48 A5 B6 48 A5 D7 48
308- A5 E0 48 A5 E1 48 20 27
310- 03 48 20 2E 03 68 91 6F
318- 68 85 E1 68 85 E0 68 85
320- 07 68 85 B6 48 AA 60 20
328- 36 03 C3 C8 B2 00 20 36
330- 03 C3 C8 B2 40 00 68 85
338- E0 68 85 E1 E6 E0 B0 02
340- E6 E1 20 79 E6 A5 97 85
348- 70 A0 00 B1 6F 60 00

3 *
4 *
5 * CHR FUNCTION FOR APPLESOFT DA
  SIC
6 *
7 * PETE ROME          JULY 1
  980
8 *
9 * COMPUTER-ADVANCED IDEAS, BERKE
  LEY
10 *
11 *
12     AST 25
13 *
14 *
15     ORG $300
16     OBJ $7000
17 *C
18 *
19 *
20 *
21 * APPLESOFT FIND POINTER TO LOCA
  TION OF VARIABLE VALUE
22 *
23 *
24 PTRGET EQU $DFE3
25 *
26 *
27 * APPLESOFT PAGE ZERO ADDRESSES
28 *
29 *
30 VARTYP EQU $11
31 TEMP  EQU $1D
32 TPSAV EQU $1E
33 VARPNT EQU $83
34 CHRGET EQU $81
35 TXTPTR EQU $80
36 *
37 *
38 *
39     LDA TXTPTR      ;SAVE
      TXTPTR LOW & HIGH BYTES

```

```

40     STA TPSAV
41     LDA TXTPTR+1
42     STA TPSAV+1
43 *
44     JSR FINDCH      ;FIND
      CHZ
45     INY              ;(Y=1)
46     LDA (VARPNT),Y ;GET LO
      W BYTE
47     STA TEMP        ;SAV
      E IT
48 *
49     JSR FINDCHS     ;FIND C
      H$
50     LDA (VARPNT),Y
51     BEQ RESTORE     ;LEAK CH
      $)=0
52 *
53     LDA TEMP        ;RET
      RIEVE CHZ LOW BYTE
54     STA (TXTPTR),Y ;SAVE C
      HZ INTO BEGINNING OF CH$
55 *
56 RESTORE LDA TPSAV      ;REST
      ORE TXTPTR
57     STA TXTPTR
58     LDA TPSAV+1
59     STA TXTPTR+1
60     RTS
61 *
62 *
63 * FINDCH LOOKS FOR CHZ
64 *
65 FINDCH JSR FINDVAR
66     ASC 'CHZ'      ;(MSB
      OFF)
67     HEX 00
68 *
69 * FINDCHS LOOKS FOR CH$
70 *
71 FINDCHS JSR FINDVAR
72     ASC 'CH$'
73     HEX 00
74 *
75 *
76 * ACTUAL ROUTINE THAT CALLS PTRC
      ET,
77 *
78 * FINDVAR RETURNS FOR:
79 *
80 * A. NUMERIC: VARPNT & TXTPTR PO
      INTING TO NUMERIC FIELD
81 * B. STRING: VARPNT POINTING TO
      STR LENGTH &
82 *     TXTPTR POINTING TO
      FIRST CHR IN STR.
83 *

```

```

84 *
85 * 1. RETRIEVE POINTER TO USER'S
      VARIABLE NAME
86 *     AND INC THE POINTER TO POIN
      T TO THE FIRST CHR.
87 *
88 FINDVAR PLA
89     STA TYTPTR
90     PLA
91     STA TXTPTR+1
92     INC TXTPTR
93     BNE CALLFP
94     INC TXTPTR+1
95 *
96 * 2. CALL APPLESOFT PTRGET TO LO
      CATE FIRST BYTE AFTER NAME.
97 *
98 CALLFP JSR PTRGET
99 *
100 * (A;Y NOW EQUALS VARPNT,VARPNT+
      1)
101 * 3. MOVE VARPNT TO TXTPTR
102 *
103     STA TXTPTR
104     STY TXTPTR+1
105 *
106 * 4. SET Y=0 AND CHECK VARIABLE
      TYPE.
107 *
108     LDY #0
109     BIT VARTYP
110     BPL FVRTS      ;NUME
      RIC. EXIT WITH HI,LO IN (VARPNT),
      Y
111 *
112 * STRING TYPE:
113 *
114     JSR CHRGET     ;BYPAS
      S LENGTH
115 *
116 * REPLACE TXTPTR C/ STRPTR ADDRE
      SS
117 *
118     LDA (TXTPTR),Y
119     PBA            ;TEMP SA
      VE LOW BYTE
120     INY
121     LDA (TXTPTR),Y
122     STA TXTPTR+1   ;SAVE S
      TRPTR HIGH BYTE
123     PLA
124     STA TXTPTR     ;SAVE
      STRPTR LOW BYTE
125     BEY            ;Y=0
126 *
127 FVRTS  RTS
      ;RETURN TO SECOND LEVEL JSR

```

```

*
* 300. 35B
300- A5 B8 B5 1E A5 B9 B5 1F
308- 20 24 03 C8 B1 B3 B5 1D
310- 20 28 03 B1 B3 F0 04 A5
318- 1D 91 B8 A5 1E B5 B8 A5
320- 1F B5 B9 60 20 32 03 43
328- 48 25 00 20 32 03 43 48
330- 24 00 68 B5 B8 68 B5 B9
338- E6 B8 D0 02 E6 B9 20 E3
340- 1F B5 B8 B4 B9 A0 00 24
348- 11 10 0F 20 B1 00 B1 B8
350- 48 C8 B1 B8 B5 B9 68 B5
358- B8 B8 60 00
    
```

How to use the CHR function:

- (1) Integer Basic
 - (A) BLOAD CHR INT,A768
 - (B) Assign CHR variable a positive number between 128 and 255.
 - (C) CALL 768

CHR INT has now created and assigned the string variable CHR\$ the ASCII value you assigned to CHR. Note that you could do this in immediate mode or in a program and you do not have to pre-assign or dimension CHR\$ before calling CHR INT.

- (2) Applesoft Basic
 - (A) BLOAD CHR FP,A768
 - (B) Let CH\$=" " (phony value)
 - (C) Assign CH% variable a positive value between 0 and 127.
 - (D) CALL 768

CHR FP has now created and assigned CH\$ the ASCII character you had in CH%. CHR FP will also work in immediate or program mode.

In the DOS 3.2 and 3.3 manuals, RWTS is described as a machine language subroutine available to "machine language programmers". The following two routines allow a BASIC programmer to use RWTS to read, write or format a diskette from Integer or Applesoft Basic:

```

4 *
5 * INTEGER BASIC TO RWTS INTERFA
   CE
6 *
7 *
8 * PETE ROME          JULY 1
   980
9 *
10 * COMPUTER-ADVANCED IDEAS, BERKE
    LEY
11 *
12 *
13     AST 25
14 *
15 *
16 * TO USE, ASSIGN:
17 *
18 *
19 * SL=LOCATION OF DISK CARD (=0
    DEFAULTS
20 * TO CURRENT SLOT & DRIVE)
21 * DR=DRIVE 1 OR 2
22 * TRK=0 TO 34
23 * SEC=0 TO 12
24 * CND=0 (LOCATE BUFFER)
25 *     =1 (READ A SECTOR)
26 *     =2 (WRITE A SECTOR)
27 *     =4 (FORMAT THE DISKETTE)
28 * BUF=RAW ADDRESS OF 256 BYTE BU
    FFER
29 * (IF 0, THEN BUF=CATALOG BU
    FFER)
30 *
31 *
32 * RETURNED:
33 *
34 *
35 * BUF=BUFFER LOCATION
36 * ERR=DISK ERRORS (NONE=0)
37 *
38 * (FOR MORE DETAILS, REFER TO DO
    S 3.2 MANUAL.)
39 *
40 *
41 *
42     AST 25
43 *
44 *
45     ORG $2F2
46     OBJ $7F2
47 *
48 *
49 * IBASIC FIND POINTER TO LOCATIO
    N OF VARIABLE VALUE
50 *
51 *
52 FIND EQU $E679C
53 *
    
```

```

54 *
55 * VARIABLE POINTER AND IBASIC NO
    UNSTK
56 *
57 *
58 VARPNT EQU $6F
59 NOUNSTKH EQU $97
60 *
61 *
62 * INTEGER BASIC PAGE ZERO LOCATI
    ONS TO BE SAVED & RESTORED
63 *
64 *
65 VERBNOW EQU $D6
66 PRINOW EQU $B7
67 PXL EQU $E0
68 PXH EQU $E1
69 *
70 *
71 * DOS PAGE 3 ADDRESSES
72 *
73 *
74 RWTS EQU $3D9
75 GETCATBUF EQU $3DC
76 GETIOB EQU $3E3
77 *
78 *
79 * IOB OFFSETS
80 *
81 *
82 IBLSLOT EQU $1 SLOT
83 IBDRWN EQU $2 DRIVE
84 IBVOL EQU $3 VOLUME
85 IBTRK EQU $4 TRACK
86 IBSECT EQU $5 SECTOR
87 IBBUFP EQU $8 BUFFER POIN
    TER
88 IBCHD EQU $C COMMAND
89 IBSTAT EQU $B ERROR STATU
    S
90 IOBPSN EQU $F LAST SLOT A
    CCESED
91 IOBPDN EQU $10 LAST DRIVE
    ADDRESSED
92 *
93 *
94     AST 25
95 *
96 *
97 * LOCAL VARIABLE
98 *
99 *
100 IOB EQU $48 POINTER TO
    IOB FOR RWTS
101 *
102 *
103 *
    
```


104 START TXA ;SAVE X-REG	149 *	198 JSR GETIOB PUT IOB INT
105 PHA	150 JSR FINDCMD FIND CMD VA	0 A+Y
106 *	LUE	199 JSR RWTS
107 * SAVE IBASIC ZPAGE POINTERS	151 *	200 *
108 *	152 *	201 *
109 LDA VERBNOW	153 BEQ RESTORE USER CMD=0	202 LDY #IBSTAT
110 PHA	154 *	203 LDA (IOB),Y RETRIEVE IB
111 LDA PRINOW	155 *	STAT
112 PHA	156 LDY #IBCMD	204 LDY #0
113 LDA PXL	157 STA (IOB),Y SAVE IN IOB	205 BCS ERROR RWTS DISK E
114 PHA		RRORS
115 LDA PYH	158 *	206 *
116 PHA	159 *	207 *
117 *	160 JSR FINDSL LOCATE SLOT	208 TYA ; FORCE IBSTA
118 JSR GETIOB GET ADDRESS	VALUE	T=0
OF IOB	161 *	209 *
119 STY IOB SAVE IT	162 BEQ INVOL SLOT=0	210 *
120 STA IOB+1	163 *	211 ERROR STA (VARPNT),Y STORE IB
121 *	164 ASL ;TIMES 16	STAT IN ERR
122 *	165 ASL	212 *
123 JSR FINDBUF FIND BUFFER	166 ASL	213 *
VARIABLE	167 ASL	214 INY
124 *	168 INY ;Y=1 (#IBSL0T)	215 LDA (IOB),Y
125 *	169 STA (IOB),Y SAVE IN IOB	216 LDY #IOBPSM
126 TAX ;SAVE BUF (LOW BY		217 STA (IOB),Y IOBPSM=IBSL
TE)	170 *	OT
127 INY	171 *	218 *
128 LDA (VARPNT),Y	172 JSR FINDDR LOCATE DRIV	219 *
129 BNE STOREBUF BUF (HIGH	E VALUE	220 LDY #IBDRVN
BYTE)0	173 *	221 LDA (IOB),Y
130 *	174 LDY #IBDRVN	222 LDY #IOBPDN
131 *	175 STA (IOB),Y SAVE IN IOB	223 STA (IOB),Y IOBPDN=IBDR
132 JSR GETCATBUF BUF=0, US		IVE
E CATBUF	176 *	224 *
133 PHA ;SAVE CATBUF (HIG	177 *	225 *
H BYTE)	178 INVOL LDA #0	226 *
134 TYA	179 LDY #IBVOL	227 RESTORE PLA ;RESTORE IBASIC ZPA
135 TAX ; CATBUF (LOW	180 STA (IOB),Y VOLUME=0	GE VALUES
BYTE)	181 *	228 STA PYH
136 LDY #0	182 *	229 PLA
137 STA (VARPNT),Y CATBUF	183 JSR FINDTRK LOCATE TRAC	230 STA PXL
(LOW)->BUF	K VALUE	231 PLA
138 INY	184 *	232 STA PRINOW
139 PLA	185 LDY #IBTRK	233 PLA
140 STA (VARPNT),Y CATBUF	186 STA (IOB),Y	234 STA VERBNOW
(HIGH)->BUF	187 *	235 *
141 *	188 *	236 PLA
142 *	189 JSR FINDSEC	237 TAX ; RESTORE X-R
143 STOREBUF LDY #IBBUF+1	190 *	EG
144 STA (IOB),Y BUF (HIGH B	191 LDY #IBSECT	238 RTS ; RETURN TO I
YTE)->IOB	192 STA (IOB),Y	BASIC
145 BEY	193 *	239 *
146 TAX	194 *	240 *
147 STA (IOB),Y BUF (LOW BY	195 JSR FINDERR	241 AST 25
TE)->IOB	196 *	242 *
148 *	197 *	243 *

```

244 * FIND BUF, CND, SL, DR, TRK, SE
      C AND ERR
245 *   BASIC VARIABLE VALUES
246 *
247 *
248     AST 25
249 *
250 *
251 FINDBUF JSR FINDVAR
252         ASC "BUF"
253         HEX 00
254 *
255 *
256 FINDCND JSR FINDVAR
257         ASC "CND"
258         HEX 00
259 *
260 *
261 FINDSL  JSR FINDVAR
262         ASC "SL"
263         HEX 00
264 *
265 *
266 FINDDR  JSR FINDVAR
267         ASC "DR"
268         HEX 00
269 *
270 *
271 FINDTRK JSR FINDVAR
272         ASC "TRK"
273         HEX 00
274 *
275 *
276 FINDSEC JSR FINDVAR
277         ASC "SEC"
278         HEX 00
279 *
280 *
281 FINDERR JSR FINDVAR
282         ASC "ERR"
283         HEX 00
284 *
285 *
286     AST 25
287 *
288 *
289 * FIND VARIABLE; NAME STORED AFT
      ER
290 * JSR. VALUE OF VARIABLE RETURN
      ED IN A-REG.
291 *
292 *
293     AST 25
294 *
295 FINDVAR PLA
296     STA PXL
297     PLA

```

```

298     STA PXL   NAME POINTE
      R ON STACK - 1
299 *
300 * PX=PX+1
301 *
302     INC PXL
303     BNE FINDV
304     INC PXL
305 *
306 * IBASIC FIND LOCATES VARIABLE N
      AME POINTED TO BY PXL,PXH
307 * FIND RETURNS VALUE POINTER IN
      MOUNSTKL,H
308 *
309 FINDV   JSR FIND
310 *
311 * MOVE HIGH BYTE OF ADDRESS FROM
      MOUNSTKH TO VARPNT+1
312 *
313     LDA MOUNSTKH IBASIC VAR
      PNTH
314     STA VARPNT+1
315     LBY #0
316     LDA (VARPNT),Y LOAD VAR
      IABLE VALUE
317     RTS

* 2F2, 3CF

2F2- 8A 48 A5 D6 48 A5
2F8-  D7 48 A5 E0 48 A5 E1 48
300- 20 E3 03 84 48 85 49 20
308- 88 03 AA C8 B1 6F D0 0E
310- 20 DC 03 48 98 AA A0 00
318- 91 6F C8 68 91 6F A0 09
320- 91 48 88 8A 91 48 20 8F
328- 03 F0 4E A0 0C 91 48 20
330- 96 03 F0 0E 0A 0A 0A 0A
338- C8 91 48 20 9C 03 A0 02
340- 91 48 A9 00 A0 03 91 48
348- 20 A2 03 A0 04 91 48 20
350- A9 03 A0 05 91 48 20 B0
358- 03 20 E3 03 20 D9 03 A0
360- 0B B1 48 A0 00 B0 01 98
368- 91 6F C8 B1 48 A0 0F 91
370- 48 A0 02 B1 48 A0 10 91
378- 48 68 85 E1 68 85 E0 68
380- 85 D7 68 85 D6 68 AA 60
388- 20 B7 03 C2 B5 C6 00 20
390- B7 03 C3 C8 C4 00 20 B7
398- 03 D3 CC 00 20 B7 03 C4
3A0- B2 00 20 B7 03 D4 D2 C8
3A8- 00 20 B7 03 D3 C5 C3 00
3B0- 20 B7 03 C5 B2 B2 00 68
3B8- 85 E0 68 85 E1 E6 E0 D0
3C0- 02 E6 E1 20 79 E6 A5 97
3C8- 85 70 A0 00 B1 6F 60 D9

```

```

4 *
5 * APPLESOFT BASIC TO RWTS INTERF
      ACE
6 *
7 *
8 * PETE ROME           JULY 1
      980
9 *
10 * COMPUTER-ADVANCED IDEAS, BERKE
      LEY
11 *
12 *
13     AST 25
14 *
15 *
16 * TO USE, ASSIGN:
17 *
18 *
19 * SLZ=LOCATION OF DISK CARD (=0
      DEFAULTS
20 *   TO CURRENT SLOT & DRIVE)
21 * DRZ=DRIVE 1 OR 2
22 * TRKZ=0 TO 34
23 * SECZ=0 TO 12
24 * CNDZ=0 (LOCATE BUFFER)
25 *   =1 (READ A SECTOR)
26 *   =2 (WRITE A SECTOR)
27 *   =4 (FORMAT THE DISKETTE)
28 * BUFZ=RAM ADDRESS OF 256 BYTE BU
      FFER
29 *   (IF 0, THEN BUF=CATALOG BU
      FFER)
30 *
31 *
32 * RETURNED:
33 *
34 *
35 * BUFZ=BUFFER LOCATION
36 * ERRZ=DISK ERRORS (NONE=0)
37 *
38 * (FOR MORE DETAILS, REFER TO DO
      S 3.2 MANUAL)
39 *
40 *
41 *
42     AST 25
43 *
44 *
45     ORG $2F2
46     OBJ $72F2
47 *
48 *
49 *
50 *
51 * APPLESOFT FIND POINTER TO LOCA
      TION OF VARIABLE NAME
52 *

```

```

53 *
54 PTRGET EQU $BFE3
55 *
56 *
57 * APPLESOFT PAGE ZERO ADDRESSES
58 *
59 *
60 VARTYP EQU $11
61 TEMP EQU $1D
62 TPSAV EQU $1E
63 VARPNT EQU $83
64 CHRGET EQU $81
65 TXTPTR EQU $88
66 *
67 *
68 * DOS PAGE 3 ADDRESSES
69 *
70 *
71 RWTS EQU $3D9
72 GETCATBUF EQU $3DC
73 GETIOB EQU $3E3
74 *
75 *
76 * IOB OFFSETS
77 *
78 *
79 IBLSLOT EQU $1 SLOT
80 IBDRVM EQU $2 DRIVE
81 IBVOL EQU $3 VOLUME
82 IBTRK EQU $4 TRACK
83 IBSECT EQU $5 SECTOR
84 IBBUFP EQU $8 BUFFER POINT
  TER
85 IBCHD EQU $C COMMAND
86 IBSTAT EQU $D ERROR STATUS
  S
87 IOBPSN EQU $F LAST SLOT A
  CCESED
88 IOBPDN EQU $10 LAST DRIVE
  ADDRESSED
89 *
90 *
91 * AST 25
92 *
93 *
94 * LOCAL VARIABLE
95 *
96 *
97 IOB EQU $48 POINTER TO
  IOB FOR RWTS
98 *
99 *
100 *
101 START LDA TXTPTR SAVE TXTPTR
  LOW & HIGH BYTES
102 STA TPSAV
103 LDA TXTPTR+1
104 STA TPSAV+1
105 *
106 *
107 JSR GETIOB GET ADDRESS
  OF IOB
108 STY IOB SAVE IT
109 STA IOB+1
110 *
111 *
112 JSR FINDBUF FIND BUFFER
  VARIABLE
113 *
114 *
115 TAX ;SAVE BUF (LOW BY
  TE)
116 DEY
117 LDA (VARPNT),Y
118 BNE STOREBUF BUF (HIGH
  BYTE)
119 *
120 *
121 JSR GETCATBUF BUFZ=0, U
  SE CATBUF
122 PHA ;SAVE CATBUF (HIGH
  H BYTE)
123 TYA
124 TAX ;CATBUF (LOW BYTE)
125 LDY #1
126 STA (VARPNT),Y CATBUF
  (LOW)->BUF
127 DEY
128 PLA
129 STA (VARPNT),Y CATBUF
  (HIGH)->BUF
130 *
131 *
132 STOREBUF LDY #IBBUFP+1
133 STA (IOB),Y BUF (HIGH B
  YTE)->IOB
134 DEY
135 TXA
136 STA (IOB),Y BUF (LOW BY
  TE)->IOB
137 *
138 *
139 JSR FINDCHD FIND CHDZ U
  ALUE
140 *
141 *
142 BEQ RESTORE USER CHDZ=0
143 *
144 *
145 LDY #IBCHD
146 STA (IOB),Y SAVE IN IOB
147 *
148 *
149 JSR FINDSL LOCATE SLOT
  VALUE
150 *
151 BEQ INVOL SLZ=0
152 *
153 ASL ;TIMES 16
154 ASL
155 ASL
156 ASL
157 STA (IOB),Y SAVE IN IOB
  (Y=1)
158 *
159 *
160 JSR FINDDR LOCATE DRIV
  E VALUE
161 *
162 INY ;(Y=2)
163 STA (IOB),Y SAVE IN IOB
164 *
165 *
166 INVOL LDA #0
167 LDY #IBVOL
168 STA (IOB),Y SET VOLUME=
  0
169 *
170 *
171 JSR FINDTRK LOCATE TRAC
  K VALUE
172 *
173 LDY #IBTRK
174 STA (IOB),Y
175 *
176 *
177 JSR FINDSEC
178 *
179 LDY #IBSECT
180 STA (IOB),Y
181 *
182 *
183 JSR FINDERR
184 *
185 *
186 JSR GETIOB PUT IOB INT
  0 A,Y
187 JSR RWTS
188 *
189 *
190 LDY #IBSTAT
191 LDA (IOB),Y RETRIEVE IB
  STAT
192 BCS ERROR RWTS DISK E
  RRRORS
193 *

```

```

194 *
195 LDA #0 FORCE IBSTA
    T=0
196 *
197 *
198 ERROR LDY #1
199 STA (VARPNT),Y STORE IB
    STAT IN ERR
200 *
201 *
202 LDA (IOB),Y (Y=1)
203 LDY #IOBPSN
204 STA (IOB),Y IOBPSN=IBSL
    OT
205 *
206 *
207 LDY #IBDRUN
208 LDA (IOB),Y
209 LDY #IOBPDN
210 STA (IOB),Y IOBPDN=IBDR
    IVE
211 *
212 *
213 *
214 RESTORE LDA TPSAW RESTORE TXT
    PTR
215 STA TXTPTR
216 LDA TPSAW+1
217 STA TXTPTR+1
218 RTS ;RETURN TO FP BASIC

219 *
220 *
221 AST 25
222 *
223 *
224 * FIND BUFZ, CHZ, SLZ, DRZ, TRK
    Z, SECZ AND ERZ
225 * BASIC VARIABLE VALUES
226 *
227 *
228 AST 25
229 *
230 *
231 FINDBUF JSR FINDVAR
232 ASC 'BUF' (NSB
    OFF)
233 HEX 00
234 *
235 *
236 FINDCHD JSR FINDVAR
237 ASC 'CHZ'
238 HEX 00
239 *
240 *
241 FINDSL JSR FINDVAR
242 ASC 'SLZ'
243 HEX 00
244 *
245 *
246 FINDDR JSR FINDVAR
247 ASC 'DRZ'
248 HEX 00
249 *
250 *
251 FINDTRK JSR FINDVAR
252 ASC 'TRZ'
253 HEX 00
254 *
255 *
256 FINDSEC JSR FINDVAR
257 ASC 'SEZ'
258 HEX 00
259 *
260 *
261 FINDERR JSR FINDVAR
262 ASC 'ERZ'
263 HEX 00
264 *
265 *
266 * ACTUAL ROUTINE THAT CALLS PTRC
    ET.
267 *
268 * FINDVAR RETURNS FOR:
269 *
270 * A. NUMERIC: VARPNT & TXTPTR PO
    INTING TO NUMERIC FIELD
271 * B. STRING: VARPNT POINTING TO
    STR LENGTH &
    TXTPTR POINTING TO
    FIRST CHR IN STR.
272 *
273 *
274 *
275 * 1. RETRIEVE POINTER TO USER'S
    VARIABLE NAME
276 * AND INC THE POINTER TO POIN
    T TO THE FIRST CHR.
277 *
278 FINDVAR PLA
279 STA TXTPTR
280 PLA
281 STA TXTPTR+1
282 INC TXTPTR
283 BNE CALLFP
284 INC TXTPTR+1
285 *
286 * 2. CALL APPLESOFT PTRGET TO LO
    CATE FIRST BYTE AFTER NAME.
287 *
288 CALLFP JSR PTRGET
289 *
290 * (A,Y NOW EQUALS VARPNT,VARPNT+
    1)
291 * 3. MOVE VARPNT TO TXTPTR
292 *
293 STA TXTPTR
294 STY TXTPTR+1
295 *
296 * 4. GET LOW BYTE VALUE AND SET
    Y=1
297 *
298 LDY #1
299 LDA (VARPNT),Y
300 RTS

* 2F2, 3C3

2F2- A5 B8 85 1E A5 B9
2F8- 85 1F 20 E3 03 84 48 85
300- 49 20 7A 03 AA 88 B1 83
308- D0 0E 20 DC 03 48 98 AA
310- A0 01 91 83 88 68 91 83
318- A0 09 91 48 88 8A 91 48
320- 20 81 03 F0 4C A0 0C 91
328- 48 20 88 03 F0 0C 0A 0A
330- 0A 0A 91 48 20 8F 03 C8
338- 91 48 A9 00 A0 03 91 48
340- 20 96 03 A0 04 91 48 20
348- 9D 03 A0 05 91 48 20 AA
350- 03 20 E3 03 20 D9 03 A0
358- 0D B1 48 B0 02 A9 00 A0
360- 01 91 83 B1 48 A0 0F 91
368- 48 A0 02 B1 48 A0 10 91
370- 48 A5 1E 85 B8 A5 1F 85
378- B9 60 20 AB 03 42 55 25
380- 00 20 AB 03 43 4D 25 00
388- 20 AB 03 53 4C 25 00 20
390- AB 03 44 52 25 00 20 AB
398- 03 54 52 25 00 20 AB 03
3A0- 53 45 25 00 20 AB 03 45
3A8- 52 25 00 68 85 B8 68 85
3B0- B9 E6 B8 D0 02 E6 B9 20
3B8- E3 DF 85 B8 84 B9 A0 01
3C0- B1 83 60 20

```

To use the Integer Basic machine language interface, you need only to:

- (1) BLOAD RWTS INT.A754
- (2) TRK=track number between 0 and 34

- (3) SEC=sector number between 0 and 12 (or 15 for DOS 3.3)
- (4) CMD=1 (for read), and
- (5) CALL 754

RWTS INT will assume you are using the same slot (SL) and drive (DR). If you care to change them, assign SL and DR legal values before calling RWTS INT. If you do not assign BUF a value for the RAM address of where you want the sector data to go, then RWTS INT will assign you an internal buffer in high RAM and put your sector data there and leave you the address in BUF. Any errors on reading, writing or formatting will be returned in ERR. To use the Applesoft version, postfix all the variable names with % (percent). Now you can build your own DISK ZAP!

SAMPLE INTEGER BASIC PROGRAM

```
10 INPUT "SLOT=",SL optional
20 INPUT "DRIVE=", DR optional
30 INPUT "TRACK=", TRK
40 INPUT "SECTOR=", SEC
50 INPUT "COMMAND(1=READ, 2=WRITE)=", CMD
60 INPUT "BUFFER=", BUF optional
70 CALL 754
```

Now BUF=address of 256 bytes read in (if CMD=1) from SL, DR and TRK, SEC. If Err= 0 now, then no errors.

interactive video

- Integrate the full computer power of the Apple* with the audiovisual impact of videotape using same TV screen
- Branch to any point on the videotape from keyboard or program command
- Provide a sophisticated teaching/training system or an audiovisual procedure manual
- Offer a sophisticated audio-visual database searchable by keyword, integrated with your card catalogue, etc.
- Available in Applesoft or PASCAL. Authoring systems that match your needs. Frame accurate stops and switches, no accumulated error.

Cavri
SYSTEMS, INC.

26 Trumbull Street, New Haven, CT 06511
(203) 562-4979

*TM - Apple Computer Co.

**IMPROVE
Your Data Entry!**

With ABT Apple Peripherals*

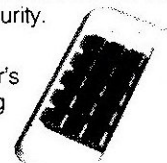
KeyPad™ Used for entering numeric data, it is essential to business applications. It features an accountants keyboard layout and permits a relaxed arm position.



BarWand™ Compatible with U.P.C., Paperbyte*, LabelCode and others. It sells Point-of-Sale inventory systems, but is also useful in libraries, factories and for security.



SoftKey™ A great programmer's aid, this features single key string entry and also customized key functions.



Available from your local Apple* Dealer

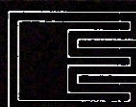
ADVANCED BUSINESS TECHNOLOGY, INC.

12333 Saratoga Sunnyvale Road, Saratoga, California 95070 408/446-2013

*Trademark of APPLE COMPUTER, INC. **Trademark of McGRAW-HILL

Selectric® Interface System

- E**ASILY interfaced to any IBM Selectric I, II, or III.
- S**TOP spinning your wheels. Letter quality at an affordable price.
- C**ONNECTS via Parallel or RS-232, accommodates varied handshaking.
- O**NLY \$575 to \$599. Dealer inquiries invited.
- N**EW design provides added features.



ESCON Products, Inc.
12919 Alcosta Blvd.
San Ramon, Ca., 94583
(415) 820-1256

SYNERGISTIC SOFTWARE

presents

GREAT ADVENTURES

Great adventure games utilizing the Apple's graphic capabilities.



DUNGEON CAMPAIGN

Explore the intricate complexities of a dungeon whose four levels are interconnected by stairways and pits. The dungeon is populated by numerous dragons, spectres, serpents, necromancers, dwarfs, elves, and an incredible variety of monsters. The inhabitant's varying powers and methods of attack will keep you guessing as your party searches the labyrinth for treasure and an assortment of useful magical devices. Try to collect your fortune and escape the dungeon before your party is destroyed.

Requires 32K APPLE and a color display. Cassette version is \$15.00; Disk version is \$17.50. Integer or Applesoft.

WILDERNESS CAMPAIGN

A surface adventure of even greater variety in which you move across the HIRES map of Draconia exploring ancient ruins, tombs, temples, and castles. Equipment and weapons can be purchased in village markets. Proper equipment will enable you to survive the numerous obstacles and hazards such as crevasses, quicksand, volcanos, avalanches, and hostile inhabitants. As you progress, you will gather enough men, weapons, and magical assistance to challenge the Great Necromancer's fortress itself.

Requires 48K. Cassette version is \$17.50; Disk version is \$20.00. Integer or Applesoft.

Get both Adventures on 1 disk for \$32.50.

ODYSSEY: THE COMPLEAT ADVENTURE

Odyssey is the ultimate adventure game for the Apple. Explore desolate islands of the dread Sargalo Sea. Learn how to enter the deserted castles, tombs, ruins, and other buildings in search of their treasures. Use your gold to buy weapons and supplies you need for your quest. With enough gold, you can buy a ship and set sail. Face pirates, monsters, storms, demon haunted dungeons, bandits, warlocks, sea serpents, and hundreds of other hazards before you try for the ultimate prize, the High One's vacant throne.

Odyssey utilizes the full capabilities of the Apple with its 3 interlocking programs; detailed and colorful high-res maps, sound effects, and varied animation effects.

Requires 48K and disk. Integer only, \$30.00

DOOM CAVERN

Doom Cavern is a high-resolution graphics version of the classic "Dungeon and Dragons" type games. Set up the persona (strength, intelligence, charisma, etc.) of your players with dice rolls, then venture forth into the dungeons of Hamardoom Castle. With perseverance, some luck, and reasoning, you can win treasures and survive to explore the dungeon.

SORCERER'S CHALLENGE

A high-res duel for supremacy between two mighty sorcerers using all their devastating spells. Strategic and tactical planning are required to outwit and defeat your opponent.

Both games available on 1 disk. Requires 48K, Integer only, \$20.00.

The high-resolution graphics and text for these games were done using our graphics utilities, HIGHER GRAPHICS II and HIGHER TEXT. Add high-resolution effects to your own programs. Each program available on disk in Integer and Applesoft. Each program \$35.00.



Synergistic
Software

PRESENTS

1. DUNGEON CAMPAIGN
 2. WILDERNESS CAMPAIGN
- CHOOSE ONE.

Available at your local dealer or send check or inquiry to
SYNERGISTIC SOFTWARE, 5221 - 120th Ave. S.E. Bellevue, WA 98005, (206) 641-1917
WA residents add 5.3% sales tax.

APPLESOFT PROGRAM LISTING FORMATTER

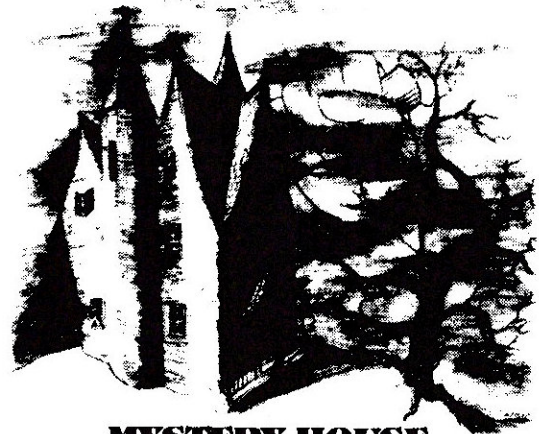
by Robert C. Clardy

The Applesoft Program List Formatter originally appeared in the May 1980 issue of CALL — APPLE and was written by **Mark Capella**. A simple but clever program, it formats listings of Applesoft programs for improved readability and logical comprehension. The program was added to, modified, crunched, and played with by a number of A.P.P.L.E. programmers*, but Mark's original concept is still its major feature. Briefly, the original program performed the following functions:

- 1) Global indenting of FOR-NEXT statements.
 - 2) Local indenting of IF-THEN statements.
 - 3) Single statements are put on single lines.
 - 4) 'LET' is added before assignment statements.
 - 5) Successive REM statements are grouped together.
 - 6) Titles and dates are allowed.
 - 7) List to screen or printer.
- Functions that were added include:
- 8) Stop list with continue or abort option.
 - 9) Paging of printout with title, user name, date, and page number at the top of each.
 - 10) Print statements, REMs, and DATA statements that go beyond one line do not break in the middle of a word..

- 11) Optional indent capability (to allow binding of output).
- 12) Optional extra line skips between statements or lines (for markups).
- 13) Lines, statements, and bytes are counted and displayed on the screen.
- 14) Start/stop list at any line number.
- 15) Return to original program when finished.
- 16) Column print width for hard copy.

*Jim Morrisett, Robert C. Clardy, Chris Anson, and Val Golding



MYSTERY HOUSE HI-RES ADVENTURE #1

Your APPLE computer becomes your eyes and ears as you enter a spooky old mansion in search of treasure. You are in complete control as you open cabinets, smash walls etc. Danger is ever present as you find your co-adventurers being murdered one by one. Can you find the killer before the killer finds you?

- OVER A HUNDRED HI-RES PICTURES
- YOUR GAME MAY BE SAVED FOR LATER CONTINUANCE
- RUNS ON BOTH 48K APPLE II AND APPLE II PLUS

Hi-Res Adventure #1 is available now at your local computer store and requires a disk drive. To order directly send \$24.95 to:

On-Line Systems
36575 Mudge Ranch Road
Coarsegold, CA 93614
209-683-6858
VISA, MST CHG, COD, CHECK ACCEPTED

Look for Hi-Res Football coming soon

To use the Listing Formatter, you must enter and save the two programs listed. MAKE FP LIST, when run, will create an EXEC file called FP LIST. This file is used to start the whole process. The user simply loads his Applesoft program and types EXEC FP LIST. FP LIST hides and protects your existing program, then loads and runs the PROGRAM LISTING FORMATTER. The user will be prompted by the FORMATTER for his choice of the options listed above. The FORMATTER then PEEKs at the program to be listed one byte at a time, interpreting each and producing its logically formatted listing.

When this style of listing is first seen, it looks peculiar and may take some getting used to. After some use, however, the visual grouping of statements within FOR-NEXT loops and after IF-THEN statements makes for much readier logical comprehension. PRINT, REM and DATA statements are easier to read, without annoying splits between words at the end of each line. Each new statement is on a new line,

allowing a quick grasp of all statements in each line. Assignments are easier to spot. The program even highlights some programming errors such as extra or missing NEXT statements. (Try some; they stand out vividly.) (overall, program comprehension, debugging, and readability are greatly enhanced.

Note that in specifying printer column width, this value may be exceeded by the number of characters in a given Applesoft token, less one. This is due to the effect of the Applesoft parsing routine. For example, if you have entered 64 as the maximum number of characters per line, and the command "RETURN" started on the 63rd character position, then that line would be 70 characters long. Accordingly, allowances should be made.

The user should be aware of two features. First, line 2 contains the variable PL% (page length). This may need to be varied to accommodate some printers/paper size. Secondly, when starting a list

in the middle of a lengthy program, there is a moderate delay while the program steps byte-by-byte up to the beginning line before starting to list. Don't hit RESET, it will get there shortly. Key variables for subsequent modifiers are:

LN% = line number
 MG% = margin indent
 IN = indent for FOR-NEXT loops
 TIN = indent for IF-THENS
 LC% = current line count
 PL% = page length
 BY = byte value
 PB = byte pointer
 NB = number of bytes left
 N1 = total number of bytes
 LP = characters printed this line
 LL% = line length
 PR% = printer slot or call number
 PN% = page number
 RF = REM flag
 QF = Quote flag
 SRF = single REM flag

(continued on page 27)

JLIST

1 REM

FP LIST FORMATTER

BY MARK CAPELLA AND OTHERS

2 PLZ = 60: GOTO 96

4 BY = PEEK (PB):PB = PB + 1:NB = NB - 1: IF NB > 1 THEN RETURN

6 NB = NB - 1: GOSUB 28: CALL 65171: CALL 65161: PRINT : PRINT : PRINT "RETURN TO ORIGINAL PROGRAM (Y/N)?: GET A\$: IF A\$ = "Y" THEN POKE 103, PEEK (0): POKE 104, PEEK (1): POKE 175, PEEK (2): POKE 176, PEEK (3)

8 PRINT : PRINT "OK.": END

10 IF LP = 0 THEN POKE 36,B + M GZ + IN + TIN: PRINT :

12 RETURN

14 PRINT : PRINT SPC(12 + MGZ) :LP = 13 + MGZ:LCZ = LCZ + 1: RETURN

16 LCZ = LCZ + 1

17 IF LCZ < PLZ OR NOT PRZ THEN RETURN

18 X = PEEK (36): FOR I = LCZ TO 70: PRINT : NEXT I
 20 PNZ = PNZ + 1: PRINT : PRINT T I\$: POKE 36,LLZ - 8: PRINT "PAGE ";PNZ: PRINT FI\$: POKE 36,LLZ - 8: PRINT DA\$"

*:LCZ = 10: POKE 36,X: PRINT : RETURN

22 IF PRZ > 0 AND PRZ < 9 THEN PRINT I\$*PR\$*PRZ:

24 IF PRZ > 10 THEN CALL PRZ
 26 RETURN

28 CALL 65171: CALL 65161:I = PEEK (37): VTAB 2: HTAB 7: INVERSE : PRINT LZ\$: HTAB 18: PRINT SZ\$: HTAB 28: PRINT N1 - NB: NORMAL : VTAB I + 1: IF PEEK (- 16384) < 127 THEN GOSUB 22: RETURN

30 POKE - 16368,0: PRINT "

PRESS ESC TO TERMINATE, ANY OTHER KEY TOCONTINUE": GET A\$: IF A\$ = CHR\$(27) THEN 6

32 VTAB PEEK (37) - 3: CALL - 956: GOSUB 22: RETURN

34 GOSUB 28: FOR I = 1 TO J: PRINT : NEXT I:LCZ = LCZ + J + (J = 0): GOSUB 17: RETURN


```

36 LP = 0:TI = 0:QF = 0:RF = 0: GOSUB
   4:X = BY: GOSUB 4:X = BY * 2
   56 + X
38 GOSUB 4:X = BY: GOSUB 4:LNZ =
   X + BY * 256: IF LNZ > STZ THEN
   44
40 GOSUB 4: IF BY THEN 40
42 GOTO 36
44 IF ENZ AND LNZ > ENZ THEN 6
46 J = S1Z: GOSUB 4: IF NOT BY THEN
   GOSUB 34: GOTO 38
48 IF NOT SRF THEN GOSUB 34: IF
   BY = 178 THEN SRF = 1
50 IF SRF AND BY < > 178 THEN S
   RF = 0: GOSUB 34
52 PRINT SPC( MGX);LNZ;LX = LX
   + 1:SZ = SZ + 1: GOTO 56
54 GOSUB 4: IF NOT BY THEN J =
   S1Z: GOSUB 34: GOTO 36
56 IF BY > 127 THEN 80
58 IF LP = 0 AND NOT RF AND NOT
   TF THEN GOSUB 10: PRINT "LE
   T ";
60 IF BY = 58 THEN PRINT " ";
62 IF TF THEN GOSUB 10
64 IF LP = 0 AND BY = 32 THEN 54

66 PRINT CHR$(BY);LP = LP + 1
   : IF LP > LLZ - 5 AND (BY =
   32 OR BY = 44) THEN GOSUB 1
   4
68 IF BY = 13 THEN GOSUB 16: IF
   QF THEN PRINT SPC( 15);LP
   = 16
70 IF BY = 34 THEN QF = NOT QF
72 IF BY < > 58 OR (QF) OR RF THEN
   54
74 SZ = SZ + 1: GOSUB 4: IF BY =
   58 AND LCZ > PLX THEN GOSUB
   18
76 IF BY < > 178 THEN J = S2Z: GOSUB
   34:LP = 0
78 GOTO 56
80 TF = 0: IF LP > 0 THEN PRINT
   " ";
82 IF BY = 130 THEN IN = IN - 4
84 GOSUB 10:I = BY - 127: PRINT
   TKN$(I);" ";LP = LP + LEN
   (TKN$(I)) + 2: IF LP > LLZ -
   5 THEN GOSUB 14
86 IF I = 2 THEN IN = IN + 4: REM

88 IF I = 69 THEN J = S2Z: GOSUB
   34:TIN = TIN + 4:LP = 0:TF =
   1
90 IF I = 4 THEN DF = 1
92 IF I = 51 THEN RF = 1:LP = 13

```

(continued on page 25)



THE WIZARD AND THE PRINCESS HI-RES ADVENTURE #2

Only ON-LINE SYSTEMS could deliver a HI-RES ADVENTURE game on such an epic scale. In this adventure you find you must do battle against an evil wizard in order to save the life of the princess. To find the wizard and his castle you must first cross deserts, oceans, mountains, travel to an island and encounter many strange beasts. You will be forced to learn magic, navigate at sea and dig for treasure. This game should provide months of adventure.

- HUNDREDS OF HI-RES PICTURES (looks great on b/w and color televisions)
- FULL 21-COLOR!! HI-RES GRAPHICS (each room a work of art)
- YOUR GAME MAY BE SAVED FOR LATER CONTINUANCE
- RUNS ON BOTH 48K APPLE II AND APPLE II PLUS
- BY FAR THE MOST AMBITIOUS GRAPHIC GAME EVER WRITTEN FOR THE APPLE!!

Hi-Res Adventure #2 is available now at your local computer store and requires a disk drive. To order directly send \$32.95 to:

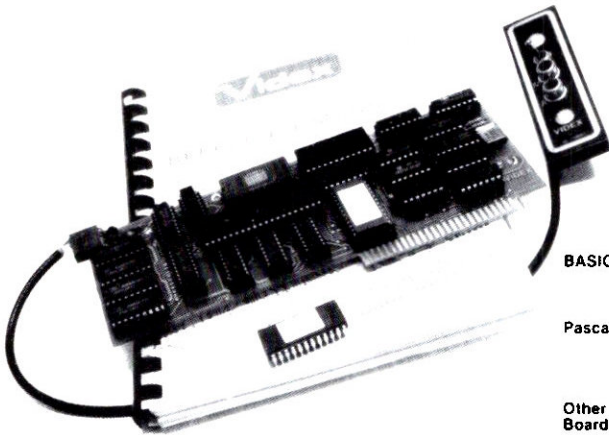
On-Line Systems
36575 Mudge Ranch Road
Coarsegold, CA 93614
209-683-6858

VISA, MST CHG, COD, CHECK ACCEPTED

Look for Hi-Res Football coming soon

The Text Solution for APPLE II®

Now APPLE II® Owners Can Solve Text Problems
With VIDEOTERM 80 Column by 24 Line Video Display
Utilizing 7 X 9 Dot Character Matrix



VIDEOTERM, MANUAL, SWITCHPLATE



7X12 MATRIX
18X80 OPTIONAL



7X9 MATRIX
24X80 STANDARD

Perhaps the most annoying shortcoming of the Apple II® is its limitation of displaying only 40 columns by 24 lines of text, all in uppercase. At last, Apple II® owners have a reliable, trouble-free answer to their text display problem. VIDEOTERM generates a full 80 columns by 24 lines of text, in upper and lower case. Twice the number of characters as the standard Apple II® display. And by utilizing a 7 by 9 character matrix, lower case letters have true descenders. But this is only the start.

VIDEOTERM

- BASICs** VIDEOTERM lists BASIC programs, both Integer and Applesoft, using the entire 80 columns. Without splitting keywords. Full editing capabilities are offered using the ESCape key sequences for cursor movement. With provision for stop/start text scrolling utilizing the standard Control-S entry. And simultaneous on screen display of text being printed.
- Pascal** Installation of VIDEOTERM in slot 3 provides Pascal immediate control of the display since Pascal recognizes the board as a standard video display terminal and treats it as such. No changes are needed to Pascal's MISC INFO or GOTOTOY files, although customization directions are provided. All cursor control characters are identical to standard Pascal defaults. And customized firmware for the Pascal system is available.
- Other Boards** The new Microsoft Softcard™ is supported. So is the popular D. C. Hayes Micromodem II™, utilizing customized PROM firmware available from VIDEX. The powerful EasyWriter™ Professional Word Processing System and other word processors are now compatible with VIDEOTERM. Or use the Mountain Hardware ROMWriter™ (or other PROM programmer) to generate your own custom character sets. Naturally, VIDEOTERM conforms to all Apple OEM guidelines, assurance that you will have no conflicts with current or future Apple II™ expansion boards.
- Advanced Hardware Design** VIDEOTERM's on-board asynchronous crystal clock ensures flicker-free character display. Only the size of the Pascal Language card, VIDEOTERM utilizes CMOS and low power consumption ICs ensuring cool, reliable operation. All ICs are fully socketed for easy maintenance. Add to that 2K of on-board RAM, 50 or 80 Hz operation, and provision of power and input connectors for a right per. Problems are designed out, not in.
- Available Options** The entire display may be altered to inverse video, displaying black characters on a white field. PROMs containing alternate character sets and graphic symbols are available from Videx. A switchplate option allows you to use the same video monitor for either the VIDEOTERM or the standard Apple II™ display, instantly changing displays by flipping a single toggle switch. The switchplate assembly inserts into one of the rear cut-outs in the Apple II™ case so that the toggle switch is readily accessible. And the Videx KEYBOARD ENHANCER can be installed, allowing upper and lower case character entry directly from your Apple II™ keyboard.
- Firmware** 1K of on-board ROM firmware controls all operation of the VIDEOTERM. No machine language patches are needed for normal VIDEOTERM use.
- Firmware Version 2.0**
- | | |
|------------|--|
| Characters | 7 x 9 matrix |
| Options | 7 x 12 matrix option.
Alternate user definable character set option
Inverse video option |
| Display | 24 x 80 (full descenders)
18 x 80 (7 x 12 matrix with full descenders) |

Want to know more? Contact your local Apple dealer today for a demonstration. VIDEOTERM is available through your local dealer or direct from Videx in Corvallis, Oregon. Or send for the VIDEOTERM Owners Reference Manual and deduct the amount if you decide to purchase. Upgrade your Apple II™ to full terminal capabilities for half the cost of a terminal. VIDEOTERM. At last.

PRICE: • VIDEOTERM includes manual \$345
• SWITCHPLATE \$ 19
• MANUAL refund with purchase \$ 18
• 7 X 12 CHARACTER SET \$ 38
• MICROMODEM FIRMWARE \$ 25

KEYBOARD ENHANCER

Upper and Lower Case Character Entry
Direct from Your Apple II® Keyboard

The Apple II™ has a simple, reliable keyboard. Unfortunately, you can only enter upper case text directly from the keyboard. Now, Videx introduces a powerful KEYBOARD ENHANCER to correct this minor annoyance.

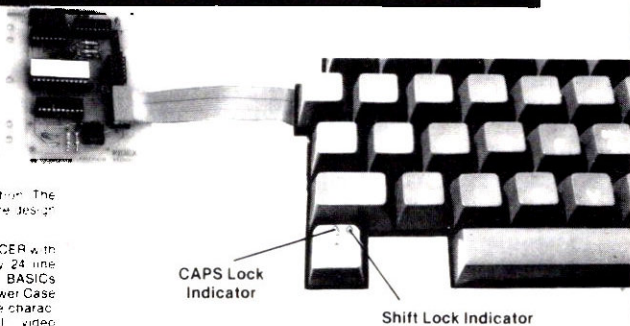
Your Apple II™ suddenly performs as if it has an ordinary typewriter keyboard. Three entry modes are now keyboard selectable. The original keyboard entry mode is still fully functional, adding the type writer mode with upper and lower case entry. Finally, the shift lock mode is available for the type writer mode also. In both of the last modes, the shift keys will perform exactly as they do on any type writer.

But that's not all. In the normal Apple II™ mode, KEYBOARD ENHANCER allows you to enter 9 new characters directly from your keyboard utilizing the Shift keys in conjunction with other alphabetic keys. A new Power key cap is included with two built-in LEDs for instant positive identification of which mode you are in. Accidental RESETs are prevented by requiring that the Control key be depressed with

the Reset key to activate the RESET operation. The easy installation and simple, rugged hardware design mean many years of trouble-free use.

You may utilize the KEYBOARD ENHANCER with Videx's VIDEOTERM for full 80 column by 24 line terminal quality display (usable with both BASICs and Pascal). Or use it with Dan Paymar's Lower Case Adaptor for display of upper and lower case characters on the standard 40 column wide Apple II™ video display. Or use it alone to simplify your word processing text entry.

KEYBOARD ENHANCER is recommended for use with Apple II™ Revision Zero and One keyboards (those lacking the Control-Reset feature). The kit includes 5 ICs mounted on a PC board, the necessary mounting screws (no drilling necessary), a power cable, Power key cap with LEDs and cable assembly, and instructions for quick installation and trouble-free use. Visit your local Apple dealer today, or contact Videx directly. And upgrade to full typewriter keyboard performance with the KEYBOARD ENHANCER.



CAPS Lock Indicator

Shift Lock Indicator

PRICE: • Keyboard Enhance \$129
• Dealer Inquiries Invited



VIDEX
897 N.W. Grant Ave.
Corvallis, OR 97330
Phone: (503) 758-0521

```

94 GOTO 54
96 TEXT : HOME : VTAB 5: HTAB 7:
  PRINT "PROGRAM LISTING FORM
  ATTER": PRINT : PRINT "BY MA
  RK CAPELLA, ROBERT C. CLARDY
  , JIM MORRISSETT, CHRIS ANS
  ON, & VAL GOLDING":D$ = CHR$(
  13) + CHR$(4): DIM TKN$(1
  27): FOR I = 1 TO 107: READ
  TKN$(I): NEXT :TKN$(36) = TK
  N$(36) + " ":TKN$(37) = TKN$(
  37) + " ": INPUT DU$

98 VTAB 12: INPUT "ENTER PRINTER
  SLOT OR CALL NUMBER (
  DEFAULT=NO PRINTER)?" :A$:PRZ
  = VAL(A$):LLZ = 35: IF PR
  Z THEN INPUT "DESIRED LINE
  LENGTH = " :LLZ

100 INPUT "
  START AFTER WHICH LINE NUMBER (RE
  TURN FOR START OF PROGRAM)
  ? " :A$:STZ = VAL(A$): INPUT
  "

ENDING LINE NUMBER (RETURN FOR EN
  D OF PROGRAM)? " :A$:ENZ =
  VAL(A$)

102 PRINT "
  INDENTATION FOR LEFT MARGIN (DEFA
  ULT=0)?" :GET A$:MGZ = VAL
  (A$): PRINT "

LINES TO SKIP BETWEEN LINE NUMBER
  S?" :GET A$:S1Z = VAL(A$)
  : PRINT "

LINES TO SKIP BETWEEN STATEMENTS?
  " :GET A$:S2Z = VAL(A$): IF
  PRZ THEN TEXT : HOME : INPUT
  "TITLE : " :TI$: INPUT "NAME
  : " :FI$: INPUT "DATE :
  " :DA$

104 IN = 0:TIN = 0:SRF = 0:PB = 8
  * 256 + 1:NB = PEEK(103) +
  PEEK(104) * 256 - PB - 1:N
  1 = NB: IF NB < 1 THEN HOME
  : VTAB 10: PRINT "NOTHING TO
  LIST.": END

106 HOME : INVERSE : PRINT "
  PRESS ANY KEY TO HALT LISTIN
  G.

LINES=0 STATE.=0 BYTES=0 0
  F " :HTAB 36: PRINT NB:
  POKE 34,2: NORMAL : VTAB 23
  : IF PRZ THEN GOSUB 22: GOSUB
  20

108 GOTO 36

```

(continued on page 27)

ON-LINE SYSTEMS PRESENTS HI-RES FOOTBALL

By JAY SULLIVAN And KEN WILLIAMS

This is the football game you always thought your Apple was capable of, but no one seemed able to deliver!

Of course, Hi-Res Football portrays the field and players in full animated Hi-Res Graphics, but do not confuse it with football games of the arcade variety. This game captures the strategy aspects of an actual football game. As coach you: call the plays, try to use the clock to your advantage, make those critical fourth down decisions, battle for field position and more! On the playfield, as quarterback, you must be able to read the defense while constantly eyeing your receiver and at the same time trying to "Fake Out" the defense. It is impossible to describe here the feelings one gets when you successfully complete that two minute goal-to-goal drill or on defense when you sack the quarterback on third and goal. Please visit a computer store in your area to request a demonstration.

Thank you.

Runs on any 48K Apple II or II Plus, DOS 3.2 or 3.3. Available now for \$39.95 on disk from your local computer store or you may order directly from:

ON-LINE SYSTEMS
36575 Mudge Ranch Road
Coarsegold, CA 93614
209-683-6858

ORDERS MAY BE CHECK, VISA, MASTER CHARGE OR C.O.D.

SAMPLE OUTPUT
APPLE ORCHARDPAGE 1
WINTER 1980-81

```

10      REM *****
15      REM ***          ***
20      REM ** SAMPLE  OUTPUT **
25      REM ***          ***
30      REM *** MARK CAPELLA ***
35      REM *** FEB 5 1980 ***
40      REM ***          ***
45      REM *****
50      REM

55      LET VIDEOMODE = 1 :
        LET TESTMODE = 0

100     REM
105     REM ***
110     REM *** FIRST PART OF SAMPLE
115     REM ***
120     REM

125     FOR I = 1 TO 100

130         IF I < 50 THEN
            PRINT I,I * 2,I * 3

135         IF I = 50 THEN
            PRINT :
            PRINT "***** "I" *****" :
            PRINT

140         IF I > 50 THEN
            PRINT I - 50,(I - 50) * 2,(I - 50) * 3

145         FOR J = 1 TO 50 :
            NEXT J : REM *** DELAY !!

150     NEXT I

200     REM
205     REM ***
210     REM *** SECOND PART OF SAMPLE
215     REM ***
220     REM

225     IF I = 3 THEN
        FOR K = 1 TO 100 :
            PRINT :
            PRINT "THIS IS A TEST" :
        NEXT

300     IF YES THEN
        9999 :
        IF TRUE THEN
            GOSUB 2000 :
            IF STILLTRUE THEN
                ABCD = 8888 :
                IF TESTING THEN
                    X = 0 : REM

```

LIST FORMATTER from page 22

DF = DATA flag
 TF = THEN flag
 TI\$ = title
 DA\$ = date
 FI\$ = name
 ST% = starting line number
 EN% = ending line number
 S1% = skip lines between lines
 S2% = skip lines between statements
 L% = line counter
 S% = statement counter
 TKN% = tokens

LIST

```

10 REM *****
15 REM ***
20 REM ** SAMPLE OUTPUT **
25 REM ***
30 REM *** MARK CAPELLA ***
35 REM *** FEB 5 1980 ***
40 REM ***
45 REM *****
50 REM
55 VIDEOMODE = 1:TESTMODE = 0
100 REM
105 REM ***
110 REM *** FIRST PART OF SAMPLE
115 REM ***
120 REM
125 FOR I = 1 TO 100
130 IF I < 50 THEN PRINT I,I *
    2,I * 3
135 IF I = 50 THEN PRINT : PRINT
    "***** "I" *****": PRINT
140 IF I > 50 THEN PRINT I - 50
    ,(I - 50) * 2,(I - 50) * 3
145 FOR J = 1 TO 50: NEXT J: REM
    *** DELAY !!
150 NEXT I
200 REM
205 REM ***
210 REM *** SECOND PART OF SAMPLE
215 REM ***
220 REM
225 IF I = 3 THEN FOR K = 1 TO
    100: PRINT : PRINT "THIS IS
    A TEST": NEXT
300 IF YES THEN 9999: IF TRUE THEN
    GOSUB 2000: IF STILLTRUE THEN
    ABCD = 8888: IF TESTING THEN
    X = 0: REM

```

FORMATTER PROGRAM from page 25

```

110 DATA END,FOR,NEXT,DATA,INPU
    T,DEL,DIM,READ,GR,TEXT: DATA
    PR$,IN$,CALL,PLOT,HLIN,VLIN
    ,HGR2,HGR,HCOLOR=,HPLOT
112 DATA DRAW,XDRAW,HTAB,HOMER
    OT=,SCALE=,SHLOAD,TRACE,NOTR
    ACE,NORMAL: DATA INVERSE ,
    FLASH , COLOR= , POP , VTAB
    ,HIMEM,LOMEM, ONERR , RESUME
    , RECALL
114 DATA STORE,SPEED=,LET,GOTO,
    RUN,IF,RESTORE,&,GOSUB,RETUR
    N: DATA REM,STOP,ON,WAIT,LO
    AD,SAVE,DEF,POKE,PRINT,CONT:
    DATA LISU,CLEAR,GET,NEW,TA
    B( ,TO,FN,SPC( ,THEN,AT: DATA
    NOT,STEP,+,-,*,/,↑,AND,OR,>
116 DATA =,<,SGN,INT,ABS,USR,FR
    E,SCRN(,PDL,POS: DATA SQR,R
    ND,LOG,EXP,COS,SIN,TAN,ATN,P
    EEK,LEN: DATA STR$,VAL,ASC,
    CHR$,LEFT$,RIGHT$,MID$

```

For use--



in business (slide show preparation)



at home (fun for children)



game development



art

HI-RES GRAPHICS FOR THE APPLE II**PADDLE-GRAPHICS/TABLET GRAPHICS**

The most powerful graphic development system available. Upper/lower case text may be drawn in any size, direction or color. Pictures may be sketched and filled in with any of 21 HI-RES colors (msut be seen to believe!!) A shape may be constructed automatically from any object appearing on the HI-RES screen.

Paddle-graphics' is for use with the standard game paddles distributed with your APPLE and TABLET-GRAPHICS is for use with APPLES' GRAPHICS TABLET.

Paddle and Tablet-Graphics are available now at your local computer store and require 48K Applesoft in rom and a disk drive. To order directly send \$39.95 for Paddle-Graphics or \$49.95 for Tablet-Graphics to:

On-Line Systems
 36575 Mudge Ranch Road
 Coarsegold, CA 93614
 209-683-6858
 VISA, MST CHG, COD, CHECK ACCEPTED

Look for Hi-Res Football coming soon

LOCATIONS OF INTEREST TO PASCAL & 6502 USERS

by Randy Hyde
Original Apple Core

Pascal is an island all to itself. None of the existing BASIC, FORTH, or 6502 assembly language programs may be used with UCSD Pascal. Pascal does support a fairly powerful, though difficult to use,

6502 assembler. This assembler is great for those time critical routines. Although Pascal runs at an average of 2-10 times faster than BASIC, this often is not good enough for some applications (fast

animation anyone?). In any case, the experienced programmer will probably find some reason for using 6502 assembly language programs in conjunction with his Pascal programs. The new user to the Pascal system's assembler will really feel cut off, because there is no provision for I/O! Actually, there is a provision for I/O, but much like the early Apple II owners, the documentation for the Pascal "monitor" (known as the "BIOS") has not yet been made generally available. There is a very good reason for this, the Pascal system has not yet been fully defined and as a result addresses may change in later systems. Since a change of only one byte would render a piece of software totally useless, Apple has elected not to make the information available until the Pascal System becomes a little more stable.

Fine, Apple has good intentions. But what about those of us who don't care? What do we do in the mean time? Well, the listed addresses may be of help to such persons:

Pascal Memory Map

\$0	- \$3FF	: System variables and stack.
\$400	- \$BFF	: Screen display area.
\$C00	- ????	: The heap starts here and grows up.
\$BDDE	- \$BEFE	: SYSCOM area.
\$BEFE	- \$BFFF	: Variables used by the BIOS.
\$C000	- \$CFFF	: I/O Memory space.
\$D000	- \$DFFF	: Low portion of the interpreter.
\$D000	- \$DFFF	: (bank switched) the BIOS.
\$E000	- \$FEFF	: The p-code interpreter.
\$FF00	- \$FFE0	: Jump vectors for the BIOS.
\$FFF6	- \$FFFF	: Reset and interrupt vectors.

APPLE & PET

MAE

The Most Powerful Disk-Based
Macro Assembler/Text Editor
Available at ANY Price

Now includes the Simplified Text Processor (STP)

For 32K PET, disk 3.0 or 4.0 ROMS or 8032 (specify) — OR — 48K APPLE II or APPLE II+ and DISK II

MAE FEATURES

- Control Files for Assembling Multiple named source files from disk
- Sorted Symbol table — Up to 31 chars./label
- 27 Commands, 26 Pseudo-ops, 39 Error Codes
- Macros, Conditional Assembly, and a new feature we developed called Interactive Assembly
- Relocatable Object Code
- String search and replace, move, copy, automatic line numbering, etc.

STP FEATURES

- 17 text processing macros
- Right and left justification
- Variable page lengths and widths
- Document size limited only by disk capacity
- Software lower case provision for APPLE II without lower case modification

ALSO INCLUDED

- Relocating Loader
- Sweet 16 macro library for APPLE and PET
- Machine Language macro library
- Sample files for Assembly and text processing
- Separate manuals for both APPLE and PET

PRICE

- MAE, STP, Relocating Loader, Library files, 50 page manual, diskette — \$169.95

SEND FOR FREE DETAILED SPEC SHEET

EASTERN HOUSE SOFTWARE
3239 LINDA DRIVE
WINSTON-SALEM, N. C. 27106

(919) 924-2889

(919) 748-8446

- CONSOLE READ: \$FF00 - Reads a character from the keyboard type-ahead buffer. Character is returned in the accumulator.
- CONSOLE WRITE: \$FF03 - Writes a character to the 80-column screen. Character is passed in the accumulator.
- CONSOLE INIT: \$FF06 - Initializes screen variables for input/output. Pointer to SYSCOM must be at SP+3, SP+4 and a pointer to break vector must be at SP+5, SP+6. (see memory map)
- PRINTER WRITE: \$FF09 - outputs character in accumulator to the printer.
- PRINTER INIT: \$FF03 - Initializes the printer.
- DISK WRITE: \$FF0F - Writes data to the disk. Block number at SP+3, SP+4. Byte count at SP+5, SP+6. Data area address at SP+7, SP+8. Drive number at SP+9, SP+10.
- DISK READ: \$FF12 - Same parameters as disk write.
- DISK INIT: \$FF15 - Inits all disk drives.
- REMOTE READ: \$FF18 - Reads a character from the serial card in slot #2.
- REMOTE WRITE: \$FF1B - Writes data to the serial card in slot #2.
- REMOTE INIT: \$FF1E - Initializes serial card in slot #2.

CREATIVITY LIFE SOFTWARE REVIEW

by David B. Garson

Program: The Creativity Life
Dynamic Package
Author: Avant-Garde Creations
P.O. Box 30161
Eugene, Oregon 97403
Distributor: Same
Purpose: To expand one's know-
ledge of creativity
Language: Applesoft (ROM), 48K,
Disk II
Price: \$19.95, Disk and Manual

RATINGS

Speed: 65
Ease of Use: 75
Documentation: 80
Error Comments: 60
Screen Display: 85
Reliability: 70
Technical Program Level: 60
Average: 71

This piece of software is truly different than most. The programs are all right, but the purpose of them is quite different. Most software serves a purpose of either pleasure (games), or usefulness (business, etc.). This package tries to enlighten your knowledge of creativity. This in itself is not bad, but the manner in which the author presents it is. The author talks at great lengths about creativity (in both the programs and the manual), giving you the impression that with the aid of this program your 'creative potential' may be expanded. Pretty heavy stuff for a program — almost like a religion? The author's point is well taken and the program does do this, but his style in presenting these ideas is too aggressive.

Thus, instead of presenting a program designed to aid in the designing of Hi-res displays, the

author's purpose is to expand our understanding and knowledge of creativity. This he does well and must be kept in mind when reading the rest of this review. Now on to the actual review of the software package. . .

The first noticeable problem with the program is that nowhere in the manual does it describe how to start the program. The manual simply begins describing what the program does. For experienced users this is no problem, but for a newcomer it can cause real headaches.

The manual is broken down into four sections, one for each of the major programs (I. Instant Graphics, II. Instant People, III. Music, and IV. Poem Writing) that appear on the diskette. The manual itself is 88 pages of first rate printing, but even with this length I still did not have a good feel for the programs. The author's style is very breezy and light-hearted as well as having little structure to the description and use of each of the programs. The most valuable pieces of documentation that I found to be of help, were two 5½" by 8½" light cardboard sheets that give a good overview of the commands for each of the programs.

The first program in the package is Instant Graphics. This piece of software is a Hi-res drawing program that allows the user to make many interesting designs as well as pictures. The user is guided through the program with the aid of the Apple's speaker. I found this technique very easy to determine just what key to press next. The program allows for a wide range of

different shapes (circles, triangles, rectangles, etc.) to be drawn anywhere on the screen in a variety of sizes and colors. All sorts of different backgrounds can be created from sliding diagonal walls, or a horizontal sliding wall, to random dots on the screen.

The program demonstrates many techniques to generate all sorts of unique pictures and designs (which are printed in the manual). The user has the ability to draw with the paddles at any time, plus the user can draw things like random lines or fireworks, to name just a couple. This program is an enjoyable one and will be appreciated by anyone with a fancy for Hi-res graphics.

The second program in the package is much like the first, but has a cute twist. Instant People allows you to draw Hi-res figures (people) on the screen instead of just miscellaneous designs. One can draw a man, woman, boy, or girl with a variety of different expressions as well as having control over the positions of the arms and legs.

One interesting note about both of these programs is the ability to save you work. Of course it has the standard method of saving the Hi-res screen, which can be loaded in and looked at any time, but it also has a save command that will save the drawing as you draw it. Thus when you retrieve that drawing it will come back just as you drew it. This capability also allows for animation (very limited) which is described in the manual, but is nothing when compared to any of the animation packages that are on the market today.

Once you have created a scene with the appropriate people you then can design the background of your choice. This feature can make for some very interesting displays. (After struggling for a while, this reviewer was able to make a man and a woman look at a mountain scene and turn their heads.)

The next two programs contained nothing new or exciting. The first was a music generation program, and as far as I am concerned there are much better ones in the public domain. The last program is called Poem Writing, but in my opinion was nothing more than a Mad-Libs program.

(continued on page 39)

WE CAN TAKE YOU FROM WATERLOO TO THE SUPER BOWL. (By way of the North Atlantic.)

In the few short months since we introduced Computer Bismarck™, we've transported over 2500 adventurous minds to the North Atlantic — there to recreate the historic battle between the awesome German warship and the British Home Fleet. The startling realism and excitement of that experience have prompted many well-seasoned travelers to proclaim it "...unique among computer games and board games alike.*" One enthusiast had this to say: "The wealth of detail...is hardly short of fantastic. Only real war rooms...in the Pentagon have ever before been able to simulate a battle in this manner.**" Now we offer two more strategy games to embark you on new flights of the imagination.

COMPUTER NAPOLEONICS™

takes you to the battlefields of Waterloo on the fateful day of June 18, 1815. Here, the greatest battle ever fought is about to begin, awaiting only your commands to set the amassed armies in motion.

You and your friend choose your role — either as the military genius, Napoleon, or as the Duke of Wellington, the iron-willed leader of the Anglo-Allied forces. The video screen displays the map of the Belgian countryside with the artillery, infantry, and cavalry units under your respective commands.

AS NAPOLEON, you must utilize your superior combat strength and numbers to deal Wellington a quick and decisive defeat before his Prussian ally can supply reinforcements. Speed is of the essence. But any tactical blunders in military deployment will result in a repeat of history — Napoleon's ignominious defeat.

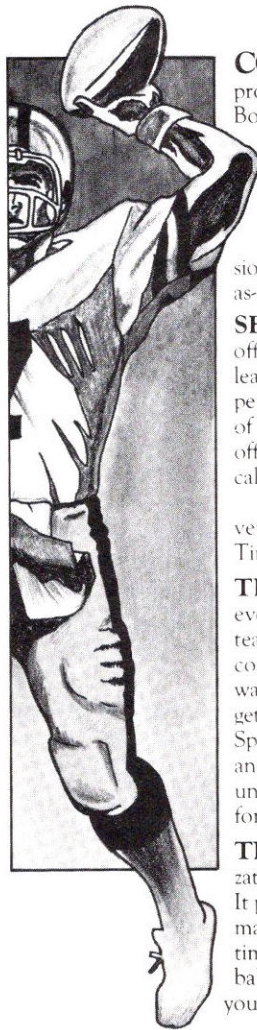
AS THE DUKE OF WELLINGTON, you must not only survive the onslaught of the French artillery, cuirassiers, and the dreaded Imperial Guard, you must also inflict sufficient damage to Napoleon's forces to prevent his relentless northward march of conquest.

THE COMPUTER, in our solitaire scenario, plays Wellington while you play Napoleon. Two levels of play are provided by making the entry of Prussian reinforcements variable. This makes the need for French military decisiveness and devastating execution even more critical.

FOR THE NOVICE AND THE ADVANCED.

Computer Napoleonics has all the advantages of your basic, traditional wargame — meticulous detail, realism, and playability. Plus one. Because the computer keeps track of all the rules, neither player can make an illegal move. This makes learning it a cinch (mastery is quite another matter), and it will convert the novice wargamer into a fanatic in no time.

The advanced wargamer will find the computer a worthy opponent indeed, and the two levels of play in the solitaire version will challenge the most experienced of strategists.



COMPUTER QUARTERBACK™

propels you onto the playing field of the Super Bowl. From its multiple offensive and defensive plays and its real-time playing conditions to the animated video display of the gridiron and the halftime statistics, no strategy football game has ever been more complete in detail or as exciting in realism. Three versions are offered: Semi-Pro, Pro, and Computer-as-Opponent.

SEMI-PRO presents you with a choice of 18 offenses and 14 defenses. Here, you will begin to learn the intricacies of football; the thrill of the perfectly executed two-minute drill; the agony of the fumble, interception, and penalty. On offense, you will learn to read the defense and call audibles as needed.

After you have mastered the Semi-Pro version, it's time to move on to the Big Time...the Pro version!

THE PRO VERSION not only gives you every offense (36) and defense (24 plus double-teaming capabilities and special alignments) you could ever want, it also gives you the team you want! With 2.7 million computer dollars, you get to draft a team to your style and specifications. Spend more on your quarterback and receivers and your passing game may very well be unstoppable...but your running game may suffer for lack of funds.

THE COMPUTER eliminates all the organizational drudgery of conventional board games. It plays scorekeeper, referee, umpire, and linesman. As timekeeper, it makes you play in real-time. Take longer than 30 seconds to hike the ball and five yards will be marched off against you for delay-of-game.

"THE ROBOTS" is the team most ably coached by your friendly computer. It's ready to play any time you are. It even "learns" your tendencies and patterns through time, and it will make the necessary tactical adjustments. It plays so well you must be in top form to stand a chance against it.

All you need to start on these mind journeys is an Apple II with Applesoft ROM card, 48K memory, and a mini-floppy disc drive. For \$59.95, Computer Napoleonics comes with the game program mini-disc, two mapboard cards, a rule book, and two player-aid charts. Computer Quarterback, for \$39.95, gives you the game disc, a rule book, and four play diagram charts.

Credit card holders, call 800-648-5600 (toll free) and charge your order to your VISA or MASTERCHARGE. In Nevada, call 800-992-5710. For Computer Quarterback, ask for Operator 178; for Computer Napoleonics, Operator 179.

While you're at it, you can also get our other games:

- Computer Bismarck for your Apple: \$59.95 (Operator 180)
- Computer Bismarck, TRS-80 48K Disc: \$59.95; 32K Cassette: \$49.95
- Computer Ambush (a tactical simulation of man-to-man combat in World War II) for your Apple: \$59.95 (Operator 181)

To order by mail, send your check to Strategic Simulations Inc., Dept. CA, 450 San Antonio Road, Suite 62, Palo Alto, CA 94306. Our 14-day money back guarantee assures your satisfaction.



STRATEGIC SIMULATIONS INC.

*Creative Computing, Aug. 1980.

**Popular Mechanics, Aug. 1980.

Apple is a registered trademark of Apple Computer Inc.
TRS-80 is a registered trademark of Tandy Corporation.

A MACHINE LANGUAGE ADDRESS CALCULATOR

by Russ Lavallee

Utilities should be written in assembly language, if at all possible, so that they can be made easily co-resident with the BASIC program for which they will be used. Utilities written in BASIC can work, but they can be cumbersome to use. A convenient location for a machine language utility is in the normally unused page \$300 — only up to \$3CF, because DOS “jumps” begin at \$3D0. Here it does not interfere with or steal memory from the BASIC program and variable areas.

This routine extends the power of Applesoft immediate or calculator mode (I) by allowing Integer hex decimal constants as input into the expression to be evaluated, along with decimal constants, variables, and all Applesoft arithmetic operators. The result will be printed 3 ways:

1. The Integer portion in hex.
2. The Integer portion in unsigned decimal.
3. The full precision result in signed floating point decimal.

More than just a converter, this makes Applesoft a powerful dec/hex calculator. Although it is intended primarily for address calculations, the printing of the FL-PT result serves two purposes:

1. The Integer portion will display negative decimal addresses if 65536 is subtracted at the end of the expression.
2. By comparison with the decimal Integer result, a check of Applesoft truncation and round-up effects can be made to see if the Integer result is the desired one (e.g., try 1/0.1).

On the slightly negative side, besides the Integer limitation on hex input and output, the expression must not exceed 3 lines (about 120 characters), only one expression per call will be evaluated, and it cannot be used in a program. Although the routine could be written without these limitations, no need was foreseen, and the extra code would prevent the page \$300 placement. Thus I will invoke the standard cop-out, “The exercise is left to the reader. . . .”.

Before the first use, the routine must be run once to set up the “&” jump at locations \$3F5,6,7 (*300G from the monitor, or CALL 768 from BASIC). Then in Applesoft, “] < expression > < return >” will print, “\$hhhh D=iiii (FP=iiii.fff)”. Hex numbers in the expression must be preceded by “\$” and a value > \$FFFF will generate an error message for both input and output. A negative decimal address can be displayed by subtracting 65536 at the end of the expression,]&FF69-65536 < return > will print -151, the well known monitor entry. Conversely, the hex equivalent of a negative decimal address is obtained simply by entering it.]&-151 < return > will print \$FF69. If 65536 is to be used often, a variable can be assigned to it, (]Z=65536), and then]&\$FF69-Z < return > will print -151.

If you don't have an assembler and your page \$300 is occupied, hand relocation to work in any page requires only changing location 301, 358, 37C, 386, 390, 39D, 3BD from value 03 to the high-order byte of the new page. To work at \$5000, use 50 for example.

Although the listing is for Applesoft in ROM, the routine works just as well for the RAM Applesoft if the locations in parenthesis in the “EQU” statements are substituted.

The basic idea behind the routine is very simple — use the ampersand to sneak in and modify the expression text to change “\$HEX-ASC” to “DEC-ASC” before sneaking out and letting Applesoft finish its job. However, the expected elegance was somewhat soured by the amount of patching needed to get the FP-BASIC routines to work with the monitor routines (eg., the different signs for ASC characters). Because of Applesoft pre-processing, when the “&” jumps to the routine, the text has been tokenized and sits in page \$200. The “MOVELP” routine scans it and moves it to the second half of the page (\$200) until it encounters the end-of-line token (00), or the “\$” character. The “\$” detection causes a branch to “GETHEX” where the subsequent hex-asc characters will be modified by turning on their sign bit. Then “GOTHEX” will point to the first hex char with the y-reg and call “GETNUM” in the monitor to read the hex-asc and convert it to binary in the A2L/H locations. “CVDEC” then diverts any printing to the “MOVDEC” routine and calls “LINPRT” to print the dec-asc equivalent of the binary number. “MOVDEC” will append decimal text to the previously moved expression text in the second half of \$200. The “MOVDEC” hook is removed by “SETVID” and text scanning continues to “MOVELP” just after the “hex-asc” number that has just been processed.

When the end-of-line token is detected, the “end” routine marks the fact in the second buffer, points to the beginning of the second buffer which now contains only decimal values, and calls “FRMNUM” to evaluate the expression and place the result in the FL-PT acc. “GETADR” converts the FL-PT acc to binary integer in ACL.H which is printed in hex by “PRNTAX”, and in unsigned decimal by “LINPRT”. The modified text buffer is pointed to again, and “FRMNUM” called

(continued on page 32)

```

*****
 4 *
 5 *   MIXED HEX.DEC ADDRESS *
 6 *   CALCULATOR UTILITY  *
 7 *
 8 * FOR USE IN ROM/RAM APPLESOFT *
 9 *   IMMEDIATE MODE      *
10 *
11 *
12 *       BY RW LAVALLEE   *
13 *
14 *   POUGHKEEPSIE NY * 5.10.80 *
15 *
*****

```

again to evaluate it, so that "PRNTFAC" can print the full precision signed decimal result. Control is then returned to BASIC. Through out the printing, "PRMSG" prints identifying characters. Each call to "PRMSG" successively prints the characters at "MSG" until stopped by a positive-ASCII character. The next call will continue with the next character. PRMSG.

```

17 *
18 *           ORG   $0300
19 *           OBJ   $6300
20 *
21 * ROM APPLESOFT: AS LISTED
22 * RAM APPLESOFT: USE ADDRESSES IN PARENTHESES
23 *
24 *

```

```

25 YSAV      EQU   $34      Y-REG SAVE LOC
26 PTR2      EQU   $35      BUFFER2 INDEX
27 CSWL      EQU   $36      COUT VECTOR
28 A2L       EQU   $3E      GETNUM RESULT LOC
29 ACL       EQU   $50      GETADR RESULT LOC
30 PTR1      EQU   $B8      CHRGET INDEX (BUFFER1)
31 CHRGET    EQU   $B1      NEXT CHAR GET ROUTINE
32 AMP       EQU   $3F5
33 BASIC     EQU   $E003    ($0C3C) NO-SCRATCH ENTRY
34 LINPRT    EQU   $E124    ($251B) PRINT A,X REGS AS DECIMAL
35 PRNTFAC   EQU   $E12E    ($2525) PRINT FP ACC AS DECIMAL
36 GETADR    EQU   $E752    ($1F49) CON FP AC TO INT IN ACL/H
37 VALERR    EQU   $F206    ($2A00) PRINT ILLEGAL QTY ERR
38 SYNERR    EQU   $DEC9    ($16CC) PRINT SYNTAX ERROR
39 FRMNUM    EQU   $DD67    ($156A) EVALUATE NUM EXPR AT PTR1
40 INBFR     EQU   $0200    TEXT BUFFER
41 COUT      EQU   $FDED    PRINT A-REG TO SCREEN
42 PRNTAX    EQU   $F941    PRINT A,X REGS AS HEX
43 SETVID    EQU   $FE93    SIMULATE PR#0 CMD
44 GETNUM    EQU   $FFA7    GET HEX NR INTO A2L/H FROM TEXT
45 INIT      LDA   #>PRNTCMD SETUP "&" JUMP VECTOR

```

```

0300: A9 03
0302: 8D F7 03
0305: A9 10
0307: 8D F6 03
030A: A9 4C
030C: 8D F5 03
030F: 60
0310: A0 80
0312: C6 B8
0314: 20 B1 00
0317: F0 50
0319: C9 24
031B: F0 08
031D: 99 00 02
0320: C8
0321: D0 F1
0323: F0 79

```

```

46 STA AMP+2
47 LDA #<PRNTCMD
48 STA AMP+1
49 LDA #$4C
50 STA AMP
51 RTS
52 PRNTCMD LDY  ##80      USE 2ND HALF OF INBFR
53 DEC PTR1      POINT TO '&'
54 MOVELP JSR CHRGET    LOOP OPN INPUT TEXT AND MOVE
55 BEQ END      STOP IF EOL,
56 CMP ##24     OR HEX INDICATOR.
57 BEQ GETHEX
58 STA INBFR,Y  NEITHER, MOVE CHAR TO BFR2
59 INY
60 BNE MOVELP   GET NEXT CHAR
61 BEQ OVFLOW   BUFFER OVERRUN, ABORT!

```

```

0325: 84 35      62  GETHEX  STY  PTR2      HEX ARG, SAVE BFR2 PTR
0327: A4 B8      63          LDY  PTR1      CHRGET PTR
0329: A2 05      64          LDX  #5        (HEX CHAR COUNT)+1
032B: CA         65  HEXLP    DEX           ;LOOP ON HEX-ASC IN BFR1
032C: 30 1F      66          BMI  ILLQTY   VALUE ERROR IF HEXDIGITS>4
032E: C8         67          INY           ;NEXT CHAR
032F: B9 00 02   68          LDA  INBFR,Y  CONV TO NEG ASC,
0332: F0 09      69          BEQ  GOTHEX   TO KEEP MONITOR HAPPY
0334: 30 07      70          BMI  GOTHEX   END IF EOL OR APSFT CMD CHAR.
0336: 09 80      71          ORA  ##80
0338: 99 00 02   72          STA  INBFR,Y
033B: D0 EE      73          BNE  HEXLP    ALWAYS, CONTINUE LOOP
033D: 84 34      74  GOTHEX  STY  YSAV     TEMP TO CHECK GETNUM LENGTH
033F: A4 B8      75          LDY  PTR1     POINTS TO "&" IN BUFFER1
0341: C8         76          INY           ;SKIP IT
0342: 20 A7 FF   77          JSR  GETNUM   GET HEX NR INTO A2L/H
0345: B8         78          DEY           ;ADJ FOR LENGTH TEST
0346: C4 34      79          CPY  YSAV     ERR IF HEXLP & GETNUM DON'T AGREE,

0348: F0 06      80          BEQ  CVDEC    ON LENGTH OF HEX STRING.
034A: 4C C9 DE   81          JMP  SYNERR   HEX SYNTAX
034D: 4C 06 F2   82  ILLQTY  JMP  VALERR   HEX TOO LARGE
0350: B8         83  CVDEC    DEY           ;ADJ FOR NEXT CHRGET,
0351: 84 B8      84          STY  PTR1     AND SAVE IT.
0353: A9 A1      85          LDA  #<MOVDEC VEC APSFT OUTPUT TO MOVDEC RTN.
0355: 85 36      86          STA  CSWL
0357: A9 03      87          LDA  #>MOVDEC
0359: 85 37      88          STA  CSWL+1
035B: A5 3F      89          LDA  A2L+1   GET APSFT TO PRNT DEC ASCII EQUIV
035D: A6 3E      90          LDX  A2L
035F: 20 24 ED   91          JSR  LINPRT
0362: 20 93 FE   92          JSR  SETVID   UNHOOK OUT VECTOR
0365: A4 35      93          LDY  PTR2     BFR2 CURRENT POSN
0367: D0 AB      94          BNE  MOVELP   ALWAYS
0369: 99 00 02   95  END     STA  INBFR,Y  APPEND EOL CHAR
036C: A0 80      96          LDY  ##80    POINT TO BFR2 ORIGIN
036E: 84 B8      97          STY  PTR1     (START OF MODIFIED EXPRESSION)
0370: 20 67 DD   98          JSR  FRMNUM   AND EVALUATE IT INTO FAC
0373: 20 52 E7   99          JSR  GETADR   INTEGER OF IT INTO ACL/H
0376: A0 C5     100         LDY  #<MSG
0378: 84 34     101         STY  YSAV     INIT YSAV TO MSG START
037A: 20 B7 03   102         JSR  PRMSG    PRINT RESULT HEADER
037D: A5 51     103         LDA  ACL+1   PRINT HEX EQUIVALENT
037F: A6 50     104         LDX  ACL
0381: 20 41 F9   105         JSR  PRNTAX
0384: 20 B7 03   106         JSR  PRMSG
0387: A5 51     107         LDA  ACL+1   PRINT DEC EQUIVALENT
0389: A6 50     108         LDX  ACL
038B: 20 24 ED   109         JSR  LINPRT
038E: 20 B7 03   110         JSR  PRMSG
0391: A0 80     111         LDY  ##80    BUFFER2 ORIGIN
0393: 84 B8     112         STY  PTR1     INTO CHRGET INDEX
0395: 20 67 DD   113         JSR  FRMNUM   EVALUATE EXPR INTO FP-ACC AGAIN
0398: 20 2E ED   114         JSR  PRNTFAC  PRINT FL-PT RESULT
039B: 20 B7 03   115         JSR  PRMSG
039E: 4C 03 E0   116  OVFLOW  JMP  BASIC    AND RETURN TO BASIC
03A1: 4B         117  MOVDEC  PHA          ;OUTPUT HANDLER FOR DECIMAL PRINT
          118  *
03A2: 84 34     119         STY  YSAV     SAVE A,Y REGS

```

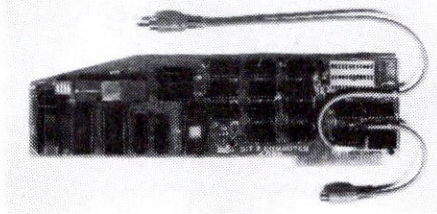
03A4:	29	7F	120	AND	#\$7F	POS ASC TO KEEP APSFT HAPPY.
03A6:	A4	35	121	LDY	PTR2	BFR2 CURRENT POSN
03A8:	99	00 02	122	STA	INBFR,Y	PUT DEC-ASC CHAR IN BFR2
03AB:	C8		123	INY		;NEXT POSN
03AC:	F0	F0	124	BEQ	OVFLOW	BFR2 OVERRUN, ABORT!
03AE:	84	35	125	STY	PTR2	SAVE POSN
03B0:	A4	34	126	LDY	YSAV	RESTORE A,Y REGS
03B2:	68		127	PLA		
03B3:	60		128	RTS		;RETURN TO LINPRT
03B4:	20	ED FD	129	PRMSG1	JSR COUT	PRINT A CHAR OF MSG
03B7:	A4	34	130	PRMSG	LDY YSAV	GET CURRENT MSG INDEX
03B9:	E6	34	131	INC	YSAV	POINT TO NEXT
03BB:	B9	00 03	132	LDA	INIT,Y	MSG CHARACTER
03BE:	30	F4	133	BMI	PRMSG1	LAST CHAR IS POS ASC
03C0:	09	B0	134	ORA	#\$80	
03C2:	4C	ED FD	135	JMP	COUT	
03C5:	24		136	MSG	ASC '\$'	
03C6:	A0	C4 3D	137	DCI	"	D="
03C9:	A0	A8 C6	138	ASC	"	<F"
03CC:	D0	3D	139	DCI	"P="	
03CE:	A9		140	ASC	")"	
03CF:	0D		141	HEX	0D	CAR RTN

--- END ASSEMBLY ---

TOTAL ERRORS: 0 208 BYTES GENERATED THIS ASSEMBLY

FULL-VIEW 80™

by
Bit 3® COMPUTER CORPORATION



DESIGNED FOR THE APPLE II®

80 X 24 DISPLAY CARD

- Permits selection of 80 column or Apple 40 column/graphics on a single monitor via keyboard or program control
- Available character fonts include 7x9, 5x7, or EROM
- Full keyboard editing
- 127 characters with upper/lower case
- Chrystal controlled clock for excellent character quality
- Keyboard Shift-Lock identified by blinking cursor. Un-Shifted keyboard identified by non-blinking cursor
- Light-pen connector and light pen firmware support
- 2K firmware on card
- Lowest power requirement of any 80x24 card on the market
- User definable character sets with the optional EROM Adaptor and a 2716 or 2732 (255 characters!) EROM
- Individually selectable inverse or normal characters
- Real time 1/60 second nonmaskable interrupt clock can be software enabled to permit timing of programs in background mode
- Tab
- 80x24 display memory is contained on card
- 50/60 Hz operation
- 7 keyboard function keys with pressure sensitive adhesive label provides HOME, HOME & CLEAR, CLEAR TO END OF LINE, CLEAR TO END OF SCREEN, CATALOG, LOAD, and RUN
- Works with Apple II, Apple II Plus, Apple Pascal, Z-80 Softcard®, and D.C. Hayes Micromodem TM
- Compatible with all Apple II peripheral cards
- Fast scroll - both scroll up and scroll down
- READ screen capability
- XY cursor positioning via GOTO X, Y command
- Blank screen command to suppress video

List Prices: 5x7 or 7x9 \$395.00
Manual only \$ 15.00

See your Apple Dealer for a demonstration of the *FULL-VIEW 80*. Or contact *Bit 3* for more information.



Bit 3 COMPUTER CORPORATION
1890 Huron Street • St. Paul, MN 55113
Telephone: (612) 926-6997

SOME NOTES ABOUT THE UCSD ASSEMBLER

by Ron Haines

Apple Users Group, New South Wales

One of the criticisms that has been aimed at the Apple Pascal System is that the documentation is often far too terse. This is particularly so in regard to the assembler, which is quite sophisticated, yet only briefly described in the manual.

In this note I'll attempt to explain two aspects of the assembler which I initially found confusing. These are the use of local labels and the addressing of variables declared as public.

1. Local Labels.

In the Apple Pascal manual [1] the use of local labels is mentioned but no explanation of what they are or how to use them is given. Quite by accident I stumbled across a reference to local labels in the manual for the assembler of a large mainframe system. This referred me to Knuth's 'Art of Computer Programming' [2]. It seems that the allowed usage of local labels in the UCSD system is more restricted than in Knuth's assembler, however the motivation for using them is the same. They provide an easy way to address locations that are a short distance away from an instruction. In addition they may be multiply defined, with only the most recent definition being usable.

Local labels consist of a '\$' followed by up to 8 digits. One

should be cautious however, since the local label stack can only hold 21 labels. Local labels cannot be used in an .EQU pseudo-op and so can only be defined by placing them in the label field before a mnemonic. Because they can be redefined, local labels can be used in macros where a normal label couldn't be used, since repeated use of the macro would multiply define a regular label.

An example of local label usage:

```

$01 LDA 00
    .
    .
    .
    JMP $01
$01 LDA 01
    BPL $01
    
```

The second branch to \$01 will be to the second definition of \$01.

Related to local labels is the idea of addressing relative to the location counter. To reference the location counter, use '*'. Thus a branch that would skip a one byte instruction could be written: BCS *+3. The following two examples are equivalent.

```

LDA C000
BPL *-3
$01 LDA C000
    BPL $01
    
```

Note that in the first example the branch was to *-3 since the location counter was pointing to the branch

instruction. Thus three bytes had to be subtracted to point to the LDA instruction. It is also permitted to omit the * altogether. Thus BPL *-3 could be written as BPL -3.

2. Public Variables.

These are data areas that are shared by the Pascal host program and the assembly language routines. Public variables must be declared in the global variable section of the Pascal host, since only then will they be given absolute addresses which the linker can use. Variables declared at deeper levels of the Pascal program are allocated storage as their associated routines are called, and so cannot be absolutely addressed.

Public variables are identified in the assembly language program by the .PUBLIC pseudo-op. The variable name may then be used as though it were a label pointing to the first byte of the variable. Thus if 'A' was declared as an integer in the Pascal host then

.PUBLIC A

.
.
.

LDA A

would load the accumulator with the low order byte of A, while

LDA A+1

would fetch the high order byte of A. Arrays are treated in the same way, with the array name becoming a symbolic label for the starting location of the array in memory. To access the individual elements of the array the indirect indexed addressing mode of the 6502 is convenient. For instance, to access the third element of an integer array named 'VEC', in the Pascal host program one could use:

```

.PUBLIC VEC
LDA VECLABL
STA ZERO1
LDA VECLABL+1
STA ZERO1+1
LDY #04
LDA @ZERO1,Y
    .
    .
    .
    
```

VECLABL .WORD VEC

(continued on page 39)



Find Your Way Around The New Apple® DOS With The Dakin5® Programming Aids 3.3®

Dakin5 Corporation, a Colorado software house, is making available to the public 12 utility programs on one 16 sector diskette, utilizing the new Apple DOS 3.3, which provides 23% more storage.

These menu-driven utilities will facilitate the development of your own microcomputer programs.

All of the **Dakin5 Programming Aids 3.3** programs are also compatible with the Corvus Disk Drive system.

This 12-in-1 set of utility programs accomplishes the following:

The Lister sends BASIC programs to the printer to be listed, utilizing the full line capacity of the printer. Pagination and page headings, including program name and date, are also provided as additional options.

The Line Cross Reference produces a display or a printed listing of all lines referenced by GOTO, THEN, GOSUB, LIST or RUN statements in an Applesoft BASIC program. Cross-referencing of most programs is done in a few seconds. An option allows you to print only the line numbers referenced in GOSUB statements.

The Variable Cross Reference creates a display or a printed listing of all variable names used in an Applesoft BASIC program, showing all line numbers where a given variable name is used.

The Peeker displays or prints either all or selected records from a text file.

The Patcher allows you to display any sector of a given file or program, and then to update any data within that sector. Another option permits you to specify the sector you wish to update such as directory sectors and sectors occupied by DOS.

The Copier copies absolutely ANY type of file or program on a normally formatted diskette from one diskette to another. The name of the program or file is the ONLY information needed.

The Calculator adds, subtracts, multiplies and divides very large numbers using numeric string data. The Calculator subroutine (using twenty place accuracy) is written in Assembler code, and runs much faster than an equivalent BASIC subroutine.

The Diskette Copy is a diskette-to-diskette copy program that does more than just copy. First, the program verifies the input. Then it formats an output disk, copies each track, and checks that the output matches the input. Additional options allow you to either initialize a diskette without DOS, or to create a copy without DOS, thereby increasing storage by 32 sectors. You may even create a copy with a different volume number than the original.

The Array Editor is a simple word processor that allows you to create, modify, print and save your own text or EXEC files.

The Screen Printer permits contents of the text screen to be sent to the printer at any time the keyboard is active (i.e. the cursor is visible). This Screen Printer program remains in effect until you press RESET or "reboot" the system.

The Prompter is a data entry subroutine that handles both string and numeric data. You have the option of using commas, decimal points and leading zeros with right-justified numerics. Alphanumeric data is left justified with trailing spaces added as required. With the Prompter you are also able to specify maximum field length to prevent overflow in both numeric and alphanumeric fields. You can even define your own set of valid characters.

The Cruncher removes REM statements, unreferenced (dead) code, and compresses code in Applesoft programs. This will increase the speed of your programs; memory and disk space savings could be more than 45%.

Many of these utility programs have been developed and tested for in-house use while producing The Controller™ business package for Apple Computer Inc.

Each programming aids package includes a program diskette and very complete documentation, all attractively packaged in a padded, blue print vinyl 3-hole notebook with silver lettering. An identifying tab separates each program for convenient reference.

See your Apple dealer or contact Dakin5 Corporation, P.O. Box 21187, Denver, Colorado 80221. Telephone: 800-525-0463. Visa or MC welcome.

DAKIN5
CORPORATION

FLOAT, FLOAT, FLOAT YOUR POINT (F.P. REPRESENTATION)

by Guy A. Lyle

COPYRIGHT© 1980, Guy A. Lyle ALL RIGHTS RESERVED

Reprinted from "The Harvest" Vol. II, No. 3, Nov. 1980

Northwest Suburban Apple Users Group

"Why can't I store a number much greater than ten to the 38th power?" "What is 'floating point'?" "How are floating point numbers stored inside the computer?" "What's the difference between floating point and integer numbers?" "When should I use floating point? Integer?"

I have heard these questions numerous times. Their central theme is the mysterious 'floating point' type of value. So, let us take a look into this mystery to see if we cannot answer some of these questions.

What is Floating Point?

The primary difference between F.P. numbers and integer numbers is that F.P. numbers may have decimal fractions as part of their values. Numbers such as 23, -100, 0, 6, and 12345 are all integer numbers. F.P. numbers can include such values as 12.1, -16.5, 0.023, -100, and 5000.123. These all possess fractional parts to their values. Note that the value -100 was included in both lists. It may be considered a F.P. value whose fractional part is ".0000....". Therefore, any integer value may also be considered to be a F.P. value, if so desired. The reverse is not always true.

How are they stored differently

Computers store integer and F.P. values differently. Integer variables are stored by the APPLE II's Applesoft Basic in two bytes of memory. The coding technique

used is known as "2's compliment" notation. Values in the range of -32768 through +32767 are legal using this notation. For some reason, Applesoft will not allow the -32768 value, making the lowest value -32767. Integer type variables are denoted by a percent sign (%) following the name of the variable, as in I%.

Applesoft stores F.P. values in five bytes of memory. The coding technique used is one variation of the floating point notations generally used. The first byte of the value represents what is known as the "exponent" of the value. The remaining four are known as the "mantissa". Very crudely, the coded value represents a number using the following formula:

$$\text{value} = \text{mantissa} * 2^{\uparrow \text{exponent}}$$

Remember that the exponentiation is always performed first. I say 'crudely' because there are some rules regarding the storage of the mantissa and the exponent in Applesoft's storage method. Floating point (or 'real') variables have no percent sign or any other sign following their names.

The Technical Details

The exponent byte is stored as an unsigned value, from 0 through 255. It is encoded in "excess 128" notation. That is, the stored exponent is 128 higher than the actual exponent which it represents. Therefore:

$$\text{actual exponent} = \text{exponent byte} - 128$$

This allows the exponent a range of -128 through +127.

The mantissa is stored to 32 bits (four bytes) of precision. This will allow storage of values up to about 4.2 billion without loss of accuracy. The mantissa is stored as a fraction rather than as an integer value, however. It is assumed that the decimal point is immediately to the left of the leftmost bit. Furthermore, the mantissa is always adjusted so that the leftmost bit is a "1". The legal range of values for the mantissa is ".10000000000000000000000000000000" through ".11111111111111111111111111111111" in binary. These represent decimal values of "0.5" and approximately "0.9999999" accordingly.

The process of forcing the leftmost bit of the mantissa to be a "1" bit is called "normalization". If the result of any computation is not already normalized, then it must be normalized before future use. This process involves shifting the bits in the mantissa to the left to bring the first non-zero bit into position. In order to compensate for the changing mantissa, the exponent must be decremented by one for each position which the mantissa is shifted left. This serves to preserve the value which the entire F.P. number represents. Conversely, any operation which requires that the bits of the mantissa be shifted to the right also requires that the exponent be incremented by one for each position shifted.

The mantissa must also have at least one bit which represents its sign, positive or negative. Since the leftmost bit of a properly normalized mantissa is always a one, Applesoft hides this bit and in its place a sign bit is provided. A zero bit represents a positive mantissa while a one bit represents a negative mantissa. Therefore the mantissa, in its final form, consists of a sign bit followed by the 31 least significant bits of the mantissa's value. The most significant bit of the mantissa is a "hidden bit" whose value is always "1".

I should add a note here regarding the sign of the F.P. value. It is the sign of the mantissa which determines the sign of the F.P. value. The sign of the exponent

merely determines the magnitude of the F.P. value — 2 ↑ exponent will always be positive regardless of whether 'exponent' is positive or negative. $2 \uparrow 3$ is +8, $2 \uparrow -3$ is $+\frac{1}{8}$ or +0.125. This point has always been one of confusion for students when studying the similar topic of scientific notation.

The following table demonstrates the storage of some common F.P. values. These were produced using a "F.P. VARIABLE PEEKER" program provided at the end of this article.

value (decimal)	F.P. representation (hex)
6	\$83 40 00 00 00
3	82 40 00 00 00
1.5	81 40 00 00 00
-1.5	81 C0 00 00 00
0.1	7E 4C CC CC CD

Examining the representation for "6", we see an exponent of \$83, or 131 decimal. The actual exponent is 131-128 or just 3. The mantissa shown is .x100000.... The 'x' is the hidden bit, and is always "1". Therefore, the real mantissa is .1100000, or 0.75 decimal. The F.P. value represented is therefore:

$$\begin{aligned} &0.75 * 2 \uparrow 3 \\ &= 0.75 * 8 \\ &= 6 \end{aligned}$$

Note that the representation for "3" simply has an exponent one less than that for "6". And the same for "1.5". Their mantissa are identical, only their exponents differ.

The "-1.5" value demonstrates the setting of the sign bit in the mantissa for the negative value. Note that the only difference between "1.5" and "-1.5" is the setting of this sign bit. This is true for any positive-negative pair. To find the absolute (positive) value of a F.P. number, simply clear its sign bit. (For you Assembly programmers, that is simply an 'AND #\$7F' with the high mantissa byte).

F.P. Operations

The full range of normal operations can be performed on floating point values; addition, subtraction, multiplication, and division. Special rules must be used, however. Those readers who

are familiar with the rules of operations when using scientific notation (also known as powers-of-ten notation) have a decided advantage here. The same rules apply.

Multiplication and division are the easiest to work with. There are three steps involved: (1) The "actual" exponents are added (for multiplication) or subtracted (for division). (2) The mantissas are multiplied. (3) The result is then properly normalized. Overflow will result if the actual exponent exceeds 127. Underflow occurs if the exponent is less than -128. Applesoft generally ignores underflows, turning the result to "0". Incidentally, "0" is a special case — all five bytes are \$00 in representing "0".

Truncation of accuracy is easily seen when looking at multiplication. When two 32-bit values are multiplied, the result is a 64-bit value. Only the most significant (left-most) 32 bits of this result are maintained in the final result. The other 32 bits are generally just chopped off. Sometimes a rounding-off algorithm may be applied, although I do not know whether or not Applesoft applies such an algorithm. If there are any non-zero bits in the 32-bit chopped off portion, then loss of accuracy results. The 32 bits of maintained result represent about $9\frac{1}{2}$ digits of decimal accuracy (1 in 4.295 billion).

Addition and subtraction generally involve a more complex process. Before these operations can take place, both exponents must be made equal. This is generally done by pre-adjusting the mantissa and exponent of the value having the least exponent. For each position the bits of the mantissa are shifted right, the exponent must be increased by one. This process continues until both exponents are equal. At this time the mantissa can be added (or subtracted). At times 32-bit additions can produce 33-bit results. Some subtractions can produce less than 32-bit results. In either case, normalization of the final result must take place.

Addition or subtraction of value

with very large differences in magnitude can produce unusual results. This is due to the pre-adjustment of the least-exponent mantissa prior to the operation. If the difference in the exponents is greater than 31, then the mantissa of the least-exponent number would be complete shifted away, leaving a mantissa of "0"! Adding (or subtracting) two such widely different numbers would produce a result equal to the larger number alone.

F.P. vs Integer

So what's the advantage of F.P. numbers? The primary advantage is the ability to express fractional portions conveniently. There are many things in the world which simply cannot be expressed well as integer values. The nine-digit accuracy of Applesoft's F.P. values also offers more utility to the user, as opposed to the $4\frac{1}{2}$ digit range of integer values. Many other BASICs only offer six or seven digit accuracy in their F.P. values. This is because they only offer a 3-byte mantissa. One could argue that double-precision (say four or five byte) integer values could have solved this problem, but this could not provide for fractional values nor the large range of exponents available in F.P. values.

F.P. values are not without their problems, however. One major problem is the time which it takes to perform F.P. operations. Much more bit-twiddling is involved in F.P. operations as compared to straight, everyday integer operations. Programs using F.P. operations are significantly slower than those involving integer operations. This is one of the primary reasons that APPLE II Integer Basic programs execute so much more rapidly than Applesoft Basic programs do. This also brings up one of the basic failings of Applesoft Basic — It fails to take advantage of easier and quicker integer value operations. In an expression containing Integer type values and variables, all values are converted first to F.P. values before operations with them are effected! (See Applesoft Reference Manual, page 18, second paragraph from

bottom.) This process of integer-to-F.P. conversion adds even more time to evaluating the expression. It is fairly easy to demonstrate, by timing repeated execution loops, that such expressions take longer to evaluate. FOR-NEXT loop execution alone would be significantly increased if the authors had allowed the use of integer-type variables for the control variable. I have used a BASIC which does allow this and the difference can be quite noticeable.

The question becomes one of "Why bother with integer-types?". In terms of Applesoft Basic, there are several things which can be stated in favor of integer-types.

Some BASIC functions expect integer-type arguments. When such functions receive F.P. arguments, time must be taken to convert them to integer values, while integer values would be used directly.

I have also made use of the integer-type variable to denote certain types of values within programs. Memory addresses, for example, may all be maintained in integer variables. As such, these represent more of a programmer's aid and do not offer much of any aid to BASIC.

The most significant area of usage for integer variables is in large numeric arrays. A 100x100 array of F.P. values would require about 50,000 bytes of memory — impossible within the APPLE II. If an integer array were used, only about 20,000 bytes of memory would be needed — possible, depending upon the size of the program. This is a savings of 60%! The only restriction is that the values to be stored must be representable in integer format: -32767 through +32767.

It should be pointed out, however, that there is no space savings in using simple (non-array) integer variables. Each numeric simple variable, F.P. or integer, consumes seven bytes of memory. For integer variables, three of the seven bytes are not used.

FLOATING POINT VARIABLE PROGRAM

A program which shows how

floating point values are stored in the computer's memory has been provided. The user RUNs the program and enters any F.P. value when requested. The program then prints out the hexadecimal representation for that value.

The representation of the value is taken directly from the table which Applesoft keeps of all its simple variables. The variable V is used before any other variables in the program in line 150. This insures that it will be the first entry in the table. Line 160 computes the memory location of the first five bytes which will hold V's value.

The FOR..NEXT loop from line 210 through line 270 scans through these five bytes, PEEKing their values from memory and displaying them on the screen. The subroutine at line 300 is called twice within the loop, printing one hexadecimal digit each time (in line 310). Line 220 computes the high order digit to be printed by the subroutine, line 240 the low order digit. Line 260 simply prints the space between each byte value.

LIST

```

100 REM FLOATING POINT
110 REM VARIABLE PEEKER
120 REM
130 REM BY GUY A. LYLE
140 REM
150 V=0: REM 1ST DEFINED
160 LOC = PEEK (105) + 256 *
    PEEK (106) + 2
170 HOME
180 INPUT "ENTER A VALUE: ";V
190 PRINT
200 HTAB 5: PRINT "$";
210 FOR I = LOC TO LOC + 4
220 :NYBBLE = INT (PEEK (I)/16)
230 : GOSUB 300
240 :NYBBLE = PEEK (I) - 16 *
    NYBBLE
250 : GOSUB 300
260 : PRINT " ";
270 NEXT I
280 PRINT : PRINT
290 GOTO 180
300 REM ** PRINT HEX
    CHARACTER **
310 PRINT CHR$(48 + NYBBLE +
    7 * (NYBBLE > 9));
320 RETURN

```

UCSD ASSEMBLER from page 35

'ZERO1' is assumed to be a zero page location defined elsewhere in the program. The 'WORD' pseudo-op is used to reserve space in the codefile for the linker to later insert the address of 'VEC'. The first thing the program does is to move this to zero page, ready for the indirect indexed addressing mode in the 'LDA' instruction. Note that the Y register is loaded with #04 to access the low order byte of the third element of 'VEC'. Each integer element takes up two bytes. The way Pascal allocates space for variables is described on page 202 of the manual. It is essential to be aware of this information before accessing public variables from assembly language routines. Note also the non-standard notation for the indirect addressing modes used by the UCSD assembler — see page 105 of the manual.

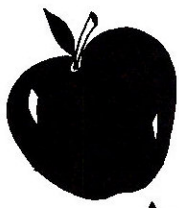
References

- [1] Apple Pascal Reference Manual page 108, Apple Computer, 1979
- [2] D.E. Knuth, The Art of Computer Programming, Vol. 1, P 147, Addison-Wesley, 1973

CREATIVITY LIFE from page 29

Both of these programs were very disappointing and really should not have been included in the package.

In summary, for \$19.95 the package is a good value. You get four programs of which two are not of good quality (Music and Poem Writing) and two of above average quality (Instant Graphics and Instant People). The Poem Writing and Music programs were big disappointments and the quality of these two should be improved as well as the capability of the programs. However, if you are looking for a nice Hi-res screen design package (of which there are many) The Creativity Life Dynamic Package offers many new and interesting aids in the development of Hi-res displays.



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 x 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

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

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.

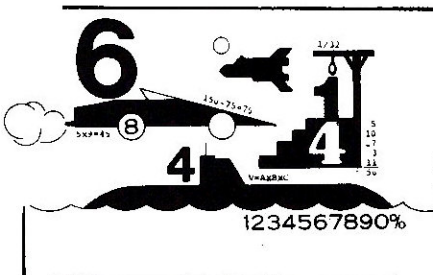
Car Jump—Make your stunt car jump the ramps. Each correct answer will increase the number of buses your car must jump over. These problems involve calculating the areas of different geometric figures.

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



Skybombers

Two nations, separated 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.

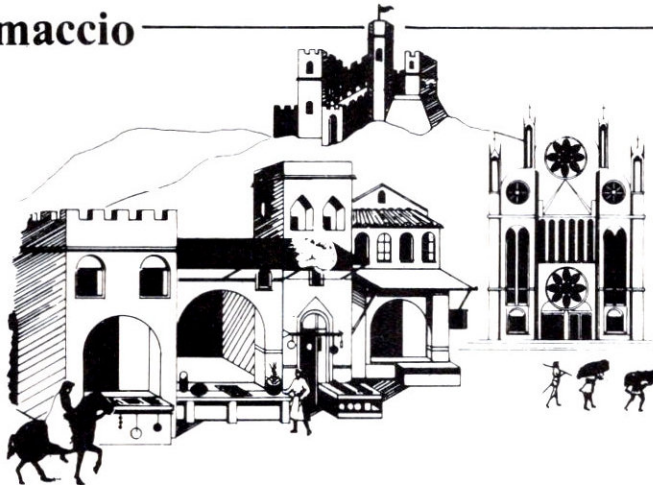
Perhaps I should acquaint you with our little domain. It is not a wealthy area, signore, but riches and glory are possible for one who is aware of political realities. These realities include your serfs. They constantly request more food from your grain reserves, grain that could be sold instead for 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.

108

Name _____

Address _____

City _____ State _____ Zip _____

Check Money Order VISA AMEX Master Charge

Card No. _____ Exp. Date _____

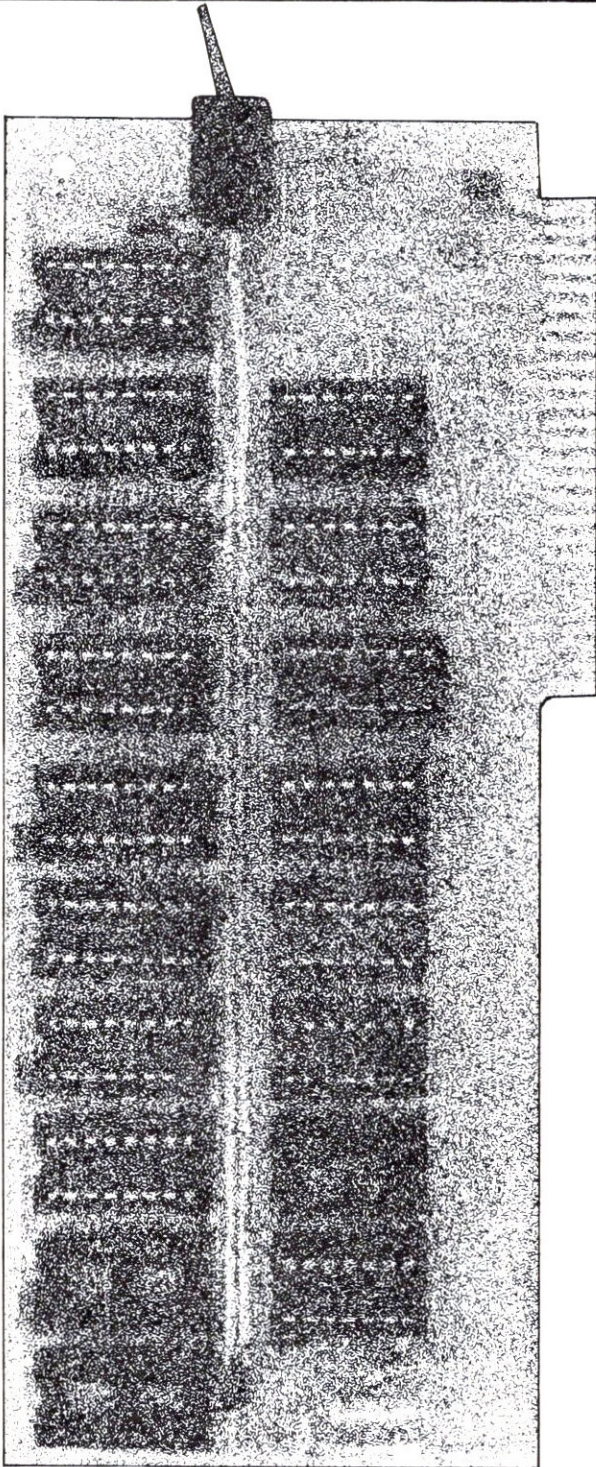
Signed _____ Date _____

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



16K Ram Expansion Board for the Apple II* \$195.00

- expands your 48K Apple to 64K of programmable memory
- works with Microsoft Z-80 card, Visicalc, LISA ver 2.0 and other software
- eliminates the need for an Applesoft* or Integer Basic ROM Card
- switch selection of RAM or mother board ROM language
- includes installation and use manual
- fully assembled and tested



Visa and MasterCard accepted



Shipping and handling will be added unless the order is accompanied by a check or money order

N.C. residents add 4% sales tax

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

ANDROMEDA



INCORPORATED**

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

**Formerly Andromeda Computer Systems

CONTACT



the user group newsletter

INSIDE THE SILENTYPE FIRMWARE

by J.D. Eisenberg
and A.J. Hertzfeld

The current Silentype firmware gives you an easy way to print text and copy graphics screens with variable margins, print intensity, printing direction. Some applications, however, require greater control over the Silentype (for example, to print your own character sets or special graphics). This document will tell you how to access the lower-level routines in the Silentype firmware to get this control. This document is written with the assumption that you have read the Silentype Operation and Reference Manual, and have your Silentype interface card plugged into slot one.

In essence, the Silentype printer consists of a movable print head made of seven wires positioned vertically. Once the printer is en-

abled, the print head produces output by heating the wires and burning dots into the special thermal paper as it moves. The print head can print about five hundred vertical columns. The exact number varies from printer to printer as the Silentype is, after all, a mechanical device.

Thus, to get absolute control over the Silentype, you need to know how to

- * initialize and enable the printer
- * specify the vertical dots to be printed
- * and tell the head which direction to move.

Initializing and Enabling the Printer

From BASIC, you can initialize the printer by executing:

PR#1

In Pascal, booting the system will automatically initialize the printer; executing:

UNITCLEAR(6)

will achieve the same effect.

After the printer is initialized, you must enable the printer ROMs before you take any further action. This is done by accessing memory locations $\$CFFF$ and $\$Cn00$ where n is the number of the slot your interface card is plugged into [in this case, $\$C100$].

In BASIC, you accomplish this via:

**100 POKE -12289,0: REM
\$CFFF**

110 POKE -16128,0: REM \$C100

In Pascal, invoke the procedure

ROMENABLE;

which is listed in Appendix A.

NOTE: Other interface cards used in the Apple may access the same ROM space as the Silentype interface card. Thus, it is a good idea to re-enable the ROMs when you go to use the Silentype after using another device. The Pascal routines in Appendix A will do this automatically; ROMENABLE is included merely for the sake of completeness.

Printing a Column of Dots

Now that the printer is enabled, you would like to print a column of seven dots. To do this, place the dot pattern you want in address \$CF2B, and jump to the firmware subroutine at address \$CBOB. The dot pattern is arranged as follows: bit seven (the high-order bit) of this byte is ignored. Bit six appears as the top of the column, and bit zero (the low-order bit) as the bottom. Thus:

This byte:

```

7 6 5 4 3 2 1 0
X  X X  X X

```

where X represents a bit that is on (a value of \$5D) is printed vertically as:

```

6X
5
4X
3X
2
1X
0X

```

In BASIC, you print a column of dots via:

```

200 POKE -12501,byte: REM
store pattern
210 CALL -13557: REM and
print it

```

and in Pascal via

```
PRINTCOL(byte)
```

where *byte* is the value you wish printed.

Moving Downwards

With this information, you now can print a line of individual columns.

To print another line, you must move the paper downwards. The paper moves down in "steps". The column of seven dots happens to be exactly four paper steps tall. That is, one downward paper step is 7/4 dot tall. [No kidding. Blame it on the Trendcom people, not us!] So, if you move the paper down four steps, the next line will abut exactly with the line above it [great for graphics]. If you want some separation between lines, move the paper downwards more than four steps. Text lines, for instance, in normal printing mode are separated by six steps (ten and a half dots).

To step the paper downwards by *n* steps, load the 6502 accumulator with the number *n*, and then call the firmware subroutine at \$CCAB.

In BASIC, you can't load the accumulator; you'll have to use assembly language for that. In Pascal, invoke:

```
STEPDOWN (n);
```

In BASIC, you can do a text line feed (that is, six steps), by calling the firmware subroutine at address \$CCD9:

```
300 CALL -13095: REM text line
feed
```

and in Pascal, invoke:

```
LINEFEED;
```

Reversing Direction

You have now printed a line of columns, and moved the paper down. At this point you may wish to switch directions and print columns going the other way [right to left].

The firmware routine at address \$CCA1 sets the head direction to move from left to right. The routine at address \$CC98 sets head direction to move from right to left. Thus, in BASIC:

```

400 CALL -13151: REM left to
right
410 CALL -13160: REM right to
left

```

and in Pascal, a call to

```

SETL2R(TRUE); (* set left to
right *)
SETL2R(FALSE); (* set right to
left *)

```

The current print direction is stored in bit seven of address \$CF01.

You may interrogate this value in BASIC by:

```
500 I=PEEK(-12543)
```

If the value you get in *I* is less than 128, the current direction is left to right; otherwise it is right to left.

In Pascal, the function LEFT2RIGHT will return TRUE if you are headed left to right, FALSE otherwise. For example:

```

IF LEFT2RIGHT THEN
WRITE('Going to the right.')
```

```

ELSE
WRITE('Going to the left.')
```

NOTE: The other bits of address \$CF01 contain information for the Silentype's internal use. Don't mess with them.

Physical Left Margin

As stated before, the Silentype is a mechanical device. If you continually reverse direction without ever aligning yourself at the physical left margin of the printer, your output will eventually become misaligned by several dots. In order to return to the true physical left margin to recalibrate, you call the firmware subroutine at address \$CBBC. In BASIC, execute:

```
600 CALL -13380: REM
recalibrate
```

and in Pascal, invoke:

```
RECALIBRATE;
```

It is suggested that you go back to the physical left margin at least once every two lines, and set your direction to left to right in order to avoid this problem.

Locking the Print Head

The Silentype will produce the best printout if the head is kept moving at a steady pace. If you need to stop and start printing (for example, if you need to do some lengthy calculation in between columns), you should lock the print head in order to keep it from coasting onwards in its current direction. This can be done by a call to the firmware subroutine at address \$CBA8. In BASIC,

```
700 CALL -13400: REM to lock
head
```

and in Pascal, invoke:

```
LOCKHEAD;
```

WARNING!

The Silentype draws upon the Apple's power supply in order to heat the wires to burn the paper. It is a bad idea to print mass quantities of black for long periods of time; it shortens print head life and puts a drain on the power supply. *UNDER NO CIRCUMSTANCES SHOULD YOU PRINT ANYTHING (especially not a dark image) WHILE THE DISK IS SPINNING!* The combined power drain may potentially damage the power supply on some Apples. In order to avoid this situation, wait two seconds after a disk access to allow the disk to "time out" before you start to print. The Silentype firmware routine does this before it prints a graphic image, and so should you!

Other Miscellaneous Routines

The firmware subroutine at location \$CD02 will seek the physical left margin, set the direction to left to right, and then move the print head to the software left margin, stored at address \$CF11. In BASIC:

```
800 CALL -13054: REM seek soft
left margin
```

and in Pascal, invoke:

```
SOFTLEFT;
```

You can move up to 256 dots in the current horizontal direction by loading a number between zero and 255 into the accumulator (a value of zero moves 256 dots), and calling the subroutine at location \$CC4E. In BASIC you can't load the accumulator; in Pascal, invoke:

```
SKIPDOTS(ndots)
```

where **ndots** is an integer expression between zero and 255.

A sample program in Pascal that uses some of the routines discussed above may be found in Appendix B.

Using a Different Character Set

The routines given above give you control over the Silentype for arbitrary graphics.

In many cases, however, you simply want to use a different character set, without having to go through all the trouble of stepping the paper downward, keeping account of direction yourself, etc.

The Silentype firmware has been written with these situations in mind. When the Silentype prints text and needs the next column of dots for a letter, it does a JuMP indirect to a subroutine that returns the next column. The address of that subroutine is stored in locations \$CF09 and \$CF0A. Location \$CF0B tells the Silentype how many columns are in each character. Normally, locations \$CF09 and \$CF0A contain the address \$CC1B; the address of the routine that fetches the next column of dots for ordinary text. Location \$CF0B normally contains six (five columns needed per character, plus one blank column).

You may, however, replace the contents of locations \$CF09 and \$CF0A with the address of a routine of your own to provide a column of dots, and the contents of location \$CF0B with a number telling how

many dots wide your special characters are. Then, when the Silentype is printing text and needs a column of dots, it will call your routine. It will send you the following information:

* The text character that it is printing is in the A-register (accumulator).

* Location \$26 tells which column of dots in that letter the Silentype would like to be given. This column is relative to the current direction. When the Silentype is printing from left to right and wants column zero of a character, that's the rightmost column of dots. When printing right to left, column zero calls for the leftmost column of dots. You can determine the direction by looking at the high bit of location \$CF01.

The Silentype gives you this information; you do your calculations or table lookup and return the column of dots that it requested into location \$CF2B.

An example of a routine that does bold letters is in Appendix C.

Appendix A

Pascal Silentype Interface Routines

```
;
; PASCAL SILENTYPE INTERFACE ROUTINES
;
; BY JDEISENBERG
;
; ASSISTED BY ANDY HERTZFELD
;
; JULY 1980
;
;
; .MACRO ENTER
; PLA
; STA RETADR
; PLA
; STA RETADR+1
; .ENDM
;
; .MACRO RETURN
; LDA RETADR+1
; PHA
; LDA RETADR
; PHA
; RTS
; .ENDM
```

```
.MACRO ENABLE
LDA $CFFF
LDA $C100
.ENDM

COLDATA .EQU $CF2B
PRINTIT .EQU $CB0B
STEPDN .EQU $CCAB
TEXTLN .EQU $CCD9
RTLFT .EQU $CC98
LFTRT .EQU $CCA1
PHYSLFT .EQU $CBBC
SOFTLFT .EQU $CD02
LOCKUP .EQU $CBA8
STEPOVR .EQU $CC4E
CURDIR .EQU $CF01

RETADR .EQU 0

; PROCEDURE ROMENABLE;
; .PROC ROMENABLE
; ENTER
; ENABLE
; RETURN
```

```

; PROCEDURE PRINTCOL(VALUE:INTEGER);
  .PROC PRINTCOL,1
  ENTER
  ENABLE          ; ENABLE ROMS
  PLA
  STA COLDATA ; PUT COLUMN IN SILENTYPE LOCATION
  PLA          ; DISCARD HIGH BYTE
  JSR PRINTIT ; AND PRINT IT
  RETURN

; PROCEDURE STEPDOWN(N:INTEGER);
  .PROC STEPDOWN,1
  ENTER
  PLA          ; GET BYTE
  JSR STEPDN  ; AND STEP DOWNWARDS
  PLA          ; DISCARD HIGH BYTE OF PARAMETER
  RETURN

; PROCEDURE LINEFEED;
  .PROC LINEFEED
  ENABLE          ; ENABLE ROMS
  JMP TEXTLN    ; STEP DOWN SIX STEPS

; PROCEDURE SETL2R(WHICH:BOOLEAN)
  .PROC SETL2R,1
  ENTER
  ENABLE          ; ENABLE ROMS
  PLA          ; GET BYTE
  BEQ R2L      ; IF ZERO, SET RIGHT TO LEFT
L2R JSR LFTRT   ; SET LEFT TO RIGHT
  JMP BACK
R2L JSR RTLFT  ; SET RIGHT TO LEFT
BACK PLA      ; DISCARD HIGH BYTE OF PARAMETER
  RETURN

; FUNCTION LEFT2RIGHT:BOOLEAN;
  .FUNC LEFT2RIGHT
  ENTER
  ENABLE          ; ENABLE ROMS
  LDY #01        ; ASSUME LEFT TO RIGHT
  LDA CURDIR     ; GET CURRENT DIRECTION
  BPL NOTSET    ; IF OFF, THEN TRUE: LEFT TO RIGHT
  DEY          ; OTHERWISE RETURN FALSE
NOTSET LDA #00   ; PUSH RESULT HIGH
  PHA
  TYA          ; PUSH RESULT LOW
  PHA
  RETURN

; PROCEDURE RECALIBRATE;
  .PROC RECALIBRATE
  ENABLE          ; ENABLE ROMS
  JMP PHYSLFT   ; RETURN TO PHYSICAL LEFT MARGIN

```


THE APPLE ORCHARD

```
;
PROCEDURE SOFTLEFT;
  .PROC SOFTLEFT ; ENABLE ROMS
  ENABLE SOFTLEFT ; GO TO SOFTWARE LEFT MARGIN
  JMP SOFTLEFT ;
;
PROCEDURE LOCKHEAD;
  .PROC LOCKHEAD ; ENABLE ROMS
  ENABLE LOCKHEAD ; LOCK HEAD
  JMP LOCKUP ;
;
PROCEDURE SKIPDOTS(N:INTEGER);
  .PROC SKIPDOTS,1 ; ENABLE ROMS
  ENTER ; GET BYTE
  PLA ; AND SKIP THE DOTS
  JSR ; DISCARD HIGH BYTE OF PARAMETER
  PLA
  RETURN
  .END
STEPOVR
```

(continued on page 48)

touch the future

reserve your place on the technical frontier!

Apple is looking for 10 of the best technical support people in the country. If you qualify, you'll be there helping personal computer owners make it happen, by providing their dealers with the technical support needed to push Apple products into new applications. While you're at it, you'll acquire the most comprehensive education imaginable in the ways personal computers are used and the opportunities for their future development.


Interested? Write to us about an exciting position as an Apple Field Technical Specialist. Openings are available now in the following areas:

- Sunnyvale, CA
- Irvine, CA
- Westboro, MA
- Carrollton, TX
- Charlotte, NC

You can spend your time solving real problems for real people in one of the most exciting companies in the world. Mail your resume to:

**Apple Computer Inc., 10260 Bandley Drive,
Cupertino, CA 95014, Attn: Kelly Westbrook.**

touch the future at

 **apple computer inc.**

Appendix B

Sample Pascal Silentype Program

```

PROGRAM CHESSBOARD;

VAR
  AFILE:TEXT;
  I,J,K:INTEGER; (* UBIQUITOUS COUNTERS *)

  PROCEDURE ROMENABLE;
  EXTERNAL;

  PROCEDURE PRINTCOL(VALUE:INTEGER);
  EXTERNAL;

  PROCEDURE STEPDOWN(N:INTEGER);
  EXTERNAL;

  PROCEDURE LINEFEED;
  EXTERNAL;

  PROCEDURE SETL2R(WHICH:BOOLEAN);
  EXTERNAL;

  FUNCTION LEFT2RIGHT:BOOLEAN;
  EXTERNAL;

  PROCEDURE RECALIBRATE;
  EXTERNAL;

  PROCEDURE SOFTLEFT;
  EXTERNAL;

  PROCEDURE LOCKHEAD;
  EXTERNAL;

  PROCEDURE SKIPDOTS(N:INTEGER);
  EXTERNAL;

  PROCEDURE DOAROW;
  BEGIN
    FOR J:=0 TO 3 DO BEGIN (* PRINT A ROW L/R *)
      PRINTCOL(127);      (* WHITE SQUARE'S BORDER *)
      FOR K:=0 TO 6 DO
        PRINTCOL(65);    (* WHITE SQUARE *)
        PRINTCOL(127);   (* OTHER BORDER *)
      FOR K:=0 TO 7 DO    (* PRINT A BLACK SQUARE *)
        PRINTCOL(127)
    END
  END;
  BEGIN

    UNITCLEAR(6);
    REWRITE(AFILE,'PRINTER:');
    WRITELN(AFILE,'PICTURE OF CHESSBOARD OR CHECKERBOARD');
    FOR I:=0 TO 3 DO BEGIN

      SETL2R(TRUE);

```

```

IF LEFT2RIGHT THEN
  WRITELN('PRINTING LEFT TO RIGHT...');

RECALIBRATE;
SKIPDOTS(200);

DOAROW;

LOCKHEAD;
STEPDOWN(4);

SETL2R(FALSE);
IF NOT LEFT2RIGHT THEN
  WRITELN('NOW PRINTING RIGHT TO LEFT. ');

DOAROW;

LOCKHEAD;
STEPDOWN(4);

END;

RECALIBRATE;
LINEFEED;
LINEFEED;
LINEFEED;
CLOSE(AFILE);

```

END.

(continued on page 50)

IAC MEMBER CLUB ADDITIONS

The following Apple user groups have joined the IAC since publication of the Fall Apple Orchard.

QUAD CITIES APPLE BYTERS
129 E. Oak Hill Dr.
Florence, AL 35630

TUCSON APPLE USERS GROUP
Pima College — 2202 W. Anklam Rd.
Tucson, AZ 85709
Phone - 602-884-6000

TRI-NETWORK APPLE USERS GROUP
8041 Sadring
Canoga Park, CA 91304
Phone -213-992-4993

HI-DESERT APPLE
537 Sydnor St.
Ridgecrest, CA 93555
Phone - 714-446-2125

SOURCE APPLE USERS GROUP
2525 Beverly Ave., #9
Santa Monica, CA 90405
Phone - 213-396-8668

APPLE BUG
4509 Millbrook Way
Bakersfield, CA 93309
Phone - 805-831-7723

EVAC
250½ W. Center Apt. B
Covina, CA 91723
Phone - 714-332-7690

UCLA APPLE USERS GROUP
17565 Bullock St.
Encino, CA 91316
Phone - 213-825-1944

APPLE-CORP OF SAN DIEGO
279 Satinwood Way
San Diego, CA 92114
Phone - 714-479-6512

APPLE JACKS
4818 Reese Rd.
Torrance, CA 90505

APPLE/VALLEY COMPUTER CLUB
4900 Newcastle
Encino, CA 91316
Phone - 213-345-8507

APPLEHOLICS ANONYMOUS
155 Morse Ave.
Ventura, CA 93003
Phone - 805-647-8945

APPLE JAX
1021 King St.
Jacksonville, FL 32204

CEDAR RAPIDS APPLE USERS GROUP
417 Third Ave.
Cedar Rapids, IA 52404
Phone - 319-366-6327

I/OWA USER GROUP
844 10th N.E.
Mason City, IA 50401

D.A.T.A.
5048 Pebble Creek Trail
Loves Park, IL 61111
Phone - 815-633-1569

SLACO
2445 Cleveland
Granite City, IL 62040
Phone - 618-451-6502

APPLE BITS
6140 Glenwood
Mission, KS 66202
Phone - 913-236-8679

APPLE CORE EXAMINERS
4691 S. Elm Dr.
Bay City, MI 48706
Phone - 517-684-9189

APPLE EYE
25 Morwood Ln.
Creve Coeur, MO 63141
Phone - 314-569-2762

A.M.M.P.L.E.
333 E. Winter
Columbia, MO 65201
Phone - 314-443-0689

(continued on page 80)

Appendix C

Sample Bold Character Routine

```

;
;      BOLD LETTERS PROCEDURE
;
;      BY ANDY HERTZFELD (JUNE 1980)
;
;      ADAPTED TO PASCAL BY JDEISENBERG (JULY 1980)
;
;      PROCEDURE BOLD (WHICH:BOOLEAN)
;
;      SET PARAMETER TO:
;
;          TRUE FOR BOLD LETTERS,
;          FALSE FOR NORMAL LETTERS.
;
;      THE BOLD LETTERS ARE PRINTED BY
;      'OR'ING THE PREVIOUS COLUMN OF DOTS WITH THE
;      CURRENT ROW OF DOTS.
;
;
;
;
;      .PROC      BOLD,1
RETURN  .EQU      0
COLUMN  .EQU      26      ; CURRENT COLUMN BEING PRINTED
;
HOOK    .EQU      0CF09    ; ADDRESS OF COLUMN GETTER ROUTINE
DEFLT1  .EQU      0CC      ; ADDRESS OF DEFAULT COLUMN
DEFLT2  .EQU      01B      ; GETTER ROUTINE
;
NUMCOLS .EQU      0CF0B    ; # OF COLUMNS IN A CHARACTER
STATUS  .EQU      0CF01    ; CURRENT DIRECTION IN HIGH BIT
;
LODOTS  .EQU      0CE0A    ; ADDRESS OF A "MULTIPLY BY 96" TABLE USED
HIDOTS  .EQU      0CE0F    ; BY THE SILENTYPE FIRMWARE FOR
SCRATCH .EQU      02A      ; GETTING COLUMNS OF DOTS
DATAOUT .EQU      0C081    ; BANK-SWITCH ADDRESS
SEND01  .EQU      0CAA3    ; FIRMWARE ROUTINE THAT DOES BANK-SWITCH
;
DOTS    .EQU      0CF2B    ; WHERE TO PUT COLUMN OF DOTS BEFORE PRINTING
;
LDA     0CFFF          ; ENABLE ROMS
LDA     0C100
;
PLA                      ; GET RETURN ADDRESS
STA     RETURN
PLA
STA     RETURN+1
;
PLA                      ; GET PARAMETER
BEQ     OFFBOLD          ; IF FALSE, TURN OFF
;
ONBOLD  LDA     BOLDADR   ; PUT ADDRESS OF OUR ROUTINE

```

```

        STA     HOOK           ; INTO THE HOOK
        LDA     BOLDADR+1
        STA     HOOK+1
        LDA     #07           ; BOLD CHARACTERS ARE 6 DOTS WIDE
        STA     NUMCOLS       ; PLUS ONE BLANK COLUMN BETWEEN.
        BNE     BACK

OFFBOLD LDA     #DEFLT2       ; PUT BACK ADDRESS OF
        STA     HOOK           ; DEFAULT COLUMN GETTER
        LDA     #DEFLT1
        STA     HOOK+1
        LDA     #06           ; WHOSE CHARACTERS ARE 5 DOTS WIDE
        STA     NUMCOLS

BACK    PLA           ; DISCARD HIGH BYTE OF PARAMETER
        LDA     RETURN+1      ; RESTORE RETURN ADDRESS
        PHA
        LDA     RETURN
        PHA
        RTS

DOBOLD  SEC           ; INTERNAL TABLE IS OFFSET BY 32
        SBC     #20
        PHA           ; SAVE CHARACTER INDEX

        LDY     COLUMN
        BEQ     DOLAST      ; COLUMN ZERO IS THE LAST ONE CALLED FOR

        CPY     #05         ; IF FIRST COLUMN
        BNE     DIREC

        LDA     #00         ; SET 'PREVIOUS' TO ZERO
        STA     PREV

DIREC  BIT     STATUS      ; TEST DIRECTION
        BPL     GETDOTS    ; IF LEFT TO RIGHT, GO ON.

        LDA     #05         ; O/W REVERSE DIRECTION OF COLUMN NUMBERS
        SEC
        SBC     COLUMN
        TAY
        INY           ; AND COUNTERACT THE DECREMENT TO FOLLOW

;
; THE SILENTYPE INTERFACE CARD HAS A BANK-SWITCHABLE
; RAM AND ROM AREA. THE FOLLOWING CODE ACCESSES THE SILENTYPE
; ROM TO GET A COLUMN OF DOTS. THE ROM AREA IS ORGANISED
; AS 96 BYTES FOR COLUMN 1 OF ALL CHARACTERS, FOLLOWED BY
; THE 96 BYTES FOR COLUMN 2 OF ALL CHARACTERS (ETC.)
;
;
GETDOTS DEY
        LDA     LODOTS,Y     ; INDEX INTO PROPER SET OF 96 BYTES
        STA     SCRATCH
        LDA     HIDOTS,Y

```

```

        STA      SCRATCH+1

        PLA          ; RECOVER CHARACTER
        TAY
        LDA      #02C          ; BANK IN ROM
        JSR      SEND01
        LDA      (SCRATCH),Y   ; GET COLUMN

        PHA          ; AND STORE AWAY FOR LATER USE
        PHA          ; (WE'LL NEED TWO COPIES)

        LDA      #0C          ; BANK BACK RAM
        STA      DATAOUT,X

        PLA          ; GET COLUMN OF DOTS BACK
        ORA      PREV        ; AND 'OR' WITH PREVIOUS COLUMN
STORDOT STA      DOTS        ; THEN STORE PRIOR TO PRINTING
        PLA          ; AGAIN GET COLUMN OF DOTS,
        STA      PREV        ; AND STORE THIS AS NEW 'PREVIOUS'
        RTS

DOLAST  LDA      PREV        ; FORGET THE 'OR'ING.
        JMP      STORDOT    ; AND FINISH UP BY PRINTING THIS COLUMN ALONE.

PREV    .BYTE    00
BOLDADR .WORD    DOBOLD
        .END

```

BINARY-TO-DECIMAL SHORTCUT (SMALL IS BEAUTIFUL)

by Steve Wozniak

When writing programs in machine language, it is generally preferable to calculate in binary, but to display in decimal. When a page number, game score, or the like must be printed, a 'binary-to-decimal' subroutine can be called. The general technique for converting an integer from one radix (base) to another is to repeatedly divide the argument by the radix of the result, retaining the remainders as successively more significant digits of that result. The process is terminated when the argument is reduced to zero by the divisions.

A better (shorter and faster) binary-to-decimal conversion algorithm can be implemented on processors supporting packed decimal (BCD) operations, such as the 6502. The following algorithm assumes that a 2-byte unsigned binary argument is in variables HEX0 (low order) and HEX1 (high order), and that the result is generated in variables DEC0 (low order), DEC1, and DEC2 (high order).

(1) Clear the result DEC0, DEC1, and DEC2. Set the processor DECIMAL mode, if any.

(2) Shift the binary argument (HEX0 and HEX1) left, the most significant bit into the CARRY.

(3) Perform the calculation

$$\text{RESULT} \leftarrow \text{RESULT} * 2 = \text{CARRY}$$

by adding the result (DEC0, DEC1, and DEC2) to itself (plus the CARRY) in decimal mode.

(4) Perform steps (2) and (3) a total of 16 times.

The algorithm works as follows. The first bit shifted out of the binary argument carries a weight of 2 to the 15th. It is added to the result which is subsequently doubled 15 times (in decimal), restoring the proper weight component. Each bit is treated in this manner.

The high-order result byte, DEC2, can be shifted left at step (3) instead of added to itself in decimal since it will not exceed 9 (\$FFFF = 06 55 35). If this is done, DEC2 need not be initialized to zero, since the original contents will be shifted out during execution. This optimization is implemented in the following 6502 subroutine.

```

2 *****
3 *
4 *   BINARY-TO-DECIMAL *
5 *   CONVERSION SUBROUTINE *
6 *
7 *   27-APR-80   WOZ *
8 *
9 *****
10 *
11 * THIS SUBROUTINE CONVERTS *
12 * A 2-BYTE UNSIGNED BINARY *
13 * ARGUMENT IN HEX0 (LOW *
14 * ORDER) AND HEX1 (HIGH) *
15 * TO A DECIMAL RESULT IN *
16 * DECO (LOW), DEC1, AND *
17 * DEC2 (HIGH). *
18 *
19 *****
20 *
21 HEX0      EQU 0      BINARY ARGUMENT (LOW BYTE)
22 HEX1      EQU 1
23 DECO      EQU 2      (LOW BYTE)
24 DEC1      EQU 3      DECIMAL RESULT
25 DEC2      EQU 4      (HIGH BYTE)
26 *
27 *****
28          ORG  $300
29          OBJ  $6300
30 *
0300: A9 00 31  HEXDEC  LDA  #0
0302: 85 02 32          STA  DECO  CLEAR RESULT (EXCEPT HIGH BYTE).
0304: 85 03 33          STA  DEC1
0306: F8      34          SED  SET      6502 DECIMAL MODE.
0307: A0 10 35          LDY  #16    PREPARE FOR 16 BITS.
0309: 06 00 36  LOOP    ASL  HEX0
030B: 26 01 37          ROL  HEX1  SHIFT BIT OUT OF BINARY ARGUMENT.
030D: A5 02 38          LDA  DECO
030F: 65 02 39          ADC  DECO
0311: 85 02 40          STA  DECO  DOUBLE DECIMAL RESULT.
0313: A5 03 41          LDA  DEC1  (PLUS CARRY)
0315: 65 03 42          ADC  DEC1
0317: 85 03 43          STA  DEC1
0319: 26 04 44          ROL  DEC2  SHIFT IN LAST BIT.
031B: 88      45          DEY          REPEAT 16 TIMES.
031C: D0 EB 46          BNE  LOOP
031E: D8      47          CLD          ;CLEAR DECIMAL MODE.
031F: 60      48          RTS  RETURN.
          49          SYM

```

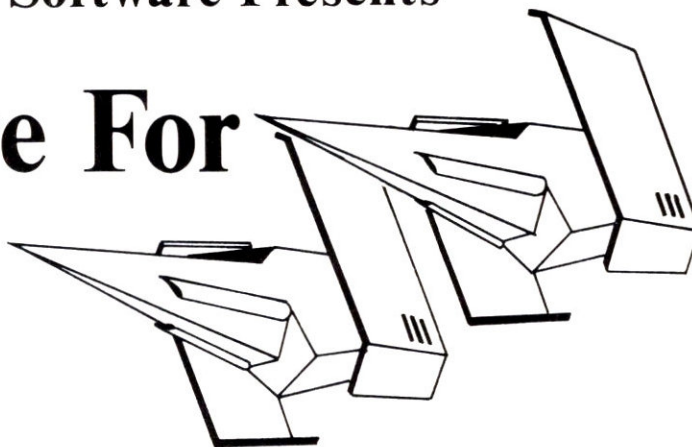
--- END ASSEMBLY ---

TOTAL ERRORS: 0

32 BYTES GENERATED THIS ASSEMBLY

Sirius Software Presents

Software For Your Apple



Cyber Strike

Cyber Strike is brand new for the Apple. A full 48K of assembly language programming with animation and original 3D effects that you haven't seen on the Apple before. INCREDIBLE! Everyone said a game like this wasn't possible on the Apple but we did it; also includes a real time clock (software implemented) and several levels of play. Uses either 13 or 16 sector Apple II, II+, or III. WARNING, THIS GAME REQUIRES PRACTICE AND FAST REFLEXES. Three levels of play and they're all tough.

E-Z Draw

E-Z Draw has not just been updated to 3.3 DOS; we've done a complete rewrite. E-Z Draw now includes the powerful Higher Text character generator written by Ron and Darrel Aldrich. With our new routines the fonts or any part of the picture can be flipped upside down, slanted right or left, rotated 90 or 180 degrees, mirrored, or any combination. Also the fonts or parts of the screen can be expanded in width or height, or compressed in height or width. You can mix portions of pictures together, or save only a portion of the screen on disk. Now fully keyboard controlled for better accuracy. Professional documentation and 20 different and imaginative type styles. Includes commands to print on Silentype printers.

Star Cruiser and Both Barrels

Star Cruiser sold over 2000 copies in the first 3 weeks and is still selling strong. Super Invader..eat your heart out. Star Cruiser is a fast action arcade game that can be played by ages 3 and up and remains a challenge to all. Requires a 32K Apple II or II+. Both Barrels seems to be attracting the younger set. This package features two games; High Noon and Duck Hunt. Fun, cute, amusing, original, and it sells. You'll love the bad guy that falls off the roof and the dogs fighting over the ducks.

Apple is a registered trademark of Apple Computer, Inc. Higher Text is a copyrighted product of Synergistic Software. Star Cruiser, E-Z Draw 3.3, Both Barrels, Duck Hunt, High Noon, and Cyber Strike are all copyrighted products of SIRIUS SOFTWARE. All rights reserved.

 SIRIUS
SOFTWARE

TABBING WITH APPLE PERIPHERALS

by John Crossley
Apple Computer, Inc.

INTRODUCTION:

This driver allows the user to TAB normally without substituting POKE 36,X for TAB(X). IT DOES REQUIRE THAT THE PR#(slot) be replaced with a CALL wherever it occurs in the program and that DOS 3.2 or 3.2.1 is in the system. It is customized for a certain type of interface in a certain slot when it is entered and will not work in any other configuration.

SOFTWARE ENTRY

First you must determine the value for the four variable parameters for the card. Look at the table for the interface that you will be using and determine the value for A, B, C, and D for the slot that the interface will be used in.

PARALLEL INTERFACE

SLOT	1	2	3	4	5	6	7
A	02	02	02	02	02	02	02
B	C1	C2	C3	C4	C5	C6	C7
C	F9	FA	FB	FC	FD	FE	FF
D	07	07	07	07	07	07	07

SERIAL INTERFACE

SLOT	1	2	3	4	5	6	7
A	07	07	07	07	07	07	07
B	C1	C2	C3	C4	C5	C6	C7
C	F9	FA	FB	FC	FD	FE	FF
D	05	05	05	05	05	05	05

SILENTYPE PRINTER

SLOT	1	2	3	4	5	6	7
A	07	07	07	07	07	07	07
B	C1	C2	C3	C4	C5	C6	C7
C	04	04	04	04	04	04	04
D	CF	CF	CF	CF	CF	CF	CF

COMMUNICATIONS CARD PRINT ROUTINE

A = 07 B = 03 C = 02 D = 03

Enter the monitor with CALL -155 and, using the value from the above tables for the items within '<>', type:

```

3B0:A9 < SLOT >
:20 95 FE
:A9 8D
:20 ED FD
:A9 C5
:85 36
:A9 03
:85 37
:4C EA 03
:20 < A > < B >
:48
:AD < C > < D >
:85 24
:68
:60
    
```

To check your typing, type:

3B0L
and compare your listing to the one below for a parallel interface in slot 1.

```

03B0- A9 01      LDA #$01
03B2- 20 95 FE   JSR $FE95
03B5- A9 8D      LDA #$8D
03B7- 20 ED FD   JSR $FDED
03BA- A9 C5      LDA #$C5
03BC- 85 36      STA $36
03BE- A9 03      LDA #$03
03C0- 85 37      STA $37
03C2- 4C EA 03   JMP $03EA
03C5- 20 02 C1   JSR $C102
03C8- 48         PHA
0369- AD F9 07   LDA $07F9
03CC- 85 24      STA $24
03CE- 68         PLA
03CF- 60         RTS
    
```

Now return to BASIC with 3D0G. SAVING THE PROGRAM TO DISK: The driver should be in memory before the printer is used. Save the driver by typing:

BSAVE TABBER, A\$3B0, L\$20

USING THE PRINTER:

NOTE: If the Apple's video is enabled, don't tab past the 40th column. It may ruin your program.

The first time you want to use the printer you must load the driver. From command mode, type:

BLOAD TABBER

This may be done from a program by entering:

100 PRINT D\$;"BLOAD TABBER" assuming that D\$ is a control-D.

The first time the printer is needed, the interface must be initialized. CALL 944 will turn on the printer and send out a carriage return to initialize the interface. Subsequently, initialization is not needed and CALL 954 will return to the printer without printing a carriage return.

Don't use PR#1, use CALL 944 or CALL 954.

Most of these interfaces have parameters that can be POKEd to modify the interface and the Parallel Printer Interface has its ctrl-I commands. These POKEs and commands should be issued just after the CALL 944. They needn't be repeated after the CALL 954.

When you want to switch back to the video monitor for output type PR#0. From within a program this must be in the form of

200 PRINT D\$;"PR#0"

EXAMPLE PROGRAM:

```

100 LET D$= CHR$(4)
110 PRINT D$;"BLOAD TABBER"
120 CALL 944
130 PRINT "THIS WILL BE PRINTED"
140 PRINT D$;"PR#0"
150 PRINT "AND NOW BACK TO THE SCREEN"
160 CALL 954
170 PRINT "NOW FOR A TABBING DEMO"
180 FOR J=1 TO 76 STEP 5
190 PRINT TAB(J);J;
200 NEXT J
210 PRINT
220 PRINT D$;"PR#0"
230 END
    
```

APPLE WRITER AND THE TELEPHONE

by Jim Hoyt

Apple Computer, Inc.

Here's a quick and easy method for transmitting your Apple Writer files from your Apple to another using the telephone. As written, the programs use the Apple Communications Card and an acoustic modem, although not much programming effort would be required to make them work with an auto-answer/auto-dial modem.

As a little history, these programs were written for an Apple dealer in London. It seems that some method to move files from several widely separated Apples in solicitors (attorneys) offices to a central printing station was a good idea. Prices of letter quality printers being what they are in England, what with V.A.T., import duties, etc., it makes good sense to provide one office rather than equip all offices with expensive printers. These programs fill that need quite nicely.

These programs will execute in a 48K system with Applesoft in ROM. They could easily be re-written in Integer Basic. Here's what they do: PROGRAM ONE (HELLO)

This is the "HELLO" program on the diskette. It allows a choice of transmitting, receiving or exiting the system. When the third option is chosen, a DOS "FP" command is issued to reset the system to its start-up parameters. (Too few programs do this and it's always a nuisance to get a "PROGRAM TOO LARGE" error because HIMEM, LOMEM or program pointers have been changed by some previous program run.)

```

LOAD HELLO
JLIST
1 GOTO 1000
2 PRINT CHR$(4)"SAVEHELLO"
: END
1000 REM
HELLO (MENU PROGRAM)
1010 TEXT : HOME : PRINT "-----
-----
-";
1020 PRINT "APPLE COMPUT—
ER APPLE WRITER MOVER
": PRINT "-----
-----"
1030 HIMEM: 6400:D$ = CHR$(4)
:Q$ = CHR$(34)
1040 VTAB 7: PRINT "YOU MAY
CHOOSE FROM THE FOLL
OWING:"
1050 VTAB 11: PRINT "1. TRAN
SMIT APPLE WRITER
FILES": PRINT
1060 PRINT "2. RECEIVE APPLE
WRITER FILES": PRINT
1070 PRINT "3. QUIT THIS PRO
GRAM"
1080 VTAB 19: HTAB 1: PRINT "E
NTER YOUR CHOICE BY
NUMBER AND PRESS":
PRINT "THE "Q$"RETURN"
Q$" KEY: ";: CALL - 958: IN
PUT "";A$
1090 LET A = VAL (A$): IF A < 1
OR A > 3 THEN VTAB 23:
HTAB 12: PRINT "INVALID
RESPONSE": FOR I = 1 to
1000: NEXT : GOTO 1080
1100 ON A GOTO 1110, 1120,
1130
1110 PRINT D$"RUNTRANS
MIT"
1120 PRINT D$"RUNRECEIVE"
1130 HOME : VTAB 22: PRINT D$
"FP"

```

PROGRAM TWO (TRANSMIT)

A little customizing is needed here. Set the variable "MS" in line 1030 to the slot number in which the Comm Card is inserted. Line 1040 prompts for the name of the file to transmit and if a Carriage Return is entered will provide a Catalog of the diskette. Line 1050 BLOADS the file. Note that all Apple Writer files start at the same address. Lines 1060 through 1090 prompt you to go ahead and give you a chance to change your mind. Line 1100 delays a bit to give the person on the other end time to get the phone handset into the modem, initializes the interface card, then POKES Apple Writer's Beginning of File mark (131). All the work is done in line 1110. The PEEKed byte is printed both on the screen and to the modem, if the End Of File mark is not found, the address counter (AD) is incremented and we go back for another byte. That's it.

```

LOAD TRANSMIT
JLIST
1 GOTO 1000
2 PRINT CHR$(4)"SAVETRANS
MIT": END
1000 REM
TRANSMIT PROGRAM
1010 TEXT : HOME : PRINT "-----
-----
-";
1020 PRINT "APPLE COMPUTE
R TRANSMIT": PRINT "----
----"
1030 HIMEM: 6400:D$ = CHR$(4)
MS = 2:Q$ = CHR$(34): POK
E 34,3: HOME : VTAB 12
1040 PRINT "FILE TO SEND: TEX
T.": INPUT "";A$: IF NOT
LEN (A$) THEN HOME : PRI
NT D$"CATALOG": PRINT :
GOTO 1040
1050 PRINT D$"BLOADTEXT. "A
$",A$1900"
1060 HOME : VTAB 12: HTAB 9:
PRINT "PRESS "Q$"RETUR
N"Q$" TO TRANSMIT": PRI
NT : HTAB 15: PRINT Q$"ES
C"Q$" TO QUIT..."
1065 VTAB 22: PRINT "ASSURE
THAT MODEM IS IN "Q$"
ORIGINATE"Q$" MODE"

```

```

1070 VTAB 14: HTAB 11: GET AS:
PRINT : IF ASC (A$) = 27
THEN PRINT D$"RUN
HELLO": END
1080 IF ASC (A$) = 13 THEN 1100
1090 PRINT CHR$ (7):: GOTO
1070
1100 HOME : VTAB 12: HTAB 14:
PRINT "SETTING UP...":
FOR I = 1 TO 10000:
NEXT : HOME : VTAB 12:
PRINT " NOW SENDING
DATA:": PRINT D$"PR#"MS
:AD = 6400: POKE 6400,131
1110 VTAB 12: HTAB 28: PRINT
PEEK (AD): CALL - 868: IF
PEEK (AD) < > 96 THEN
AD = AD + 1: GOTO 1110
1120 PRINT D$"PR#0": PRINT
CHR$ (7): HOME
1130 VTAB 12: PRINT "SEND
ANOTHER FILE? "": CALL -
958: INPUT "";A$: IF LEFT$
(A$,1) = "Y" THEN GOTO
1030
1140 IF LEFT$ (A$,1) = "N" THEN
PRINT D$"RUNHELLO":
END
1150 GOTO 1130

```

PROGRAM THREE (RECEIVE)

Again, set "MS" to your modem slot. The important line in this module is 1140. POKE 50,128 resets the prompt character from a "?" to a space. The NORMAL command re-sets it. When the EOF mark (96) is found, the user is prompted for a file name to save and the program re-executes.

LOAD RECEIVE

```

JLIST
1 GOTO 1000
2 PRINT CHR$ (4) "SAVE
RECEIVE": END
1000 REM
RECEIVE PROGRAM
1010 HIMEM: 6400: TEXT : HOME
: PRINT "-----";
1020 PRINT "APPLE COMPUT
ER RECEIVE": PRINT "--
----"
1040 LET MS = 2:D$ = CHR$ (4)
1050 VTAB 12: HTAB 8: CALL -
958: PRINT "PRESS" CHR$
(34) "RETURN" CHR$ (34)"
TO BEGIN OR"

```

```

1060 PRINT : HTAB 14: PRINT
CHR$ (34) "ESC" CHR$ (34)
" TO QUIT..."
1065 VTAB 22: PRINT " ASSURE
THAT MODEM IS IN "Q$"
ANSWER"Q$" MODE"
1070 VTAB 14: HTAB 10: GET AS:
IF ASC (A$) = 13 THEN 1100
1080 IF ASC (A$) = 27 THEN
PRINT D$"RUNHELLO"
1090 PRINT CHR$ (7):: GOTO
1070
1100 PRINT : VTAB 12: CALL -
958
1110 VTAB 13: PRINT " NOW
RECEIVING INFOR:"
1120 LET AD = 6400
1130 PRINT D$"IN#"MS
1140 POKE 50,128: VTAB 13:
HTAB 28: INPUT A$:
NORMAL : POKE AD, VAL (A
$) : AD = AD + 1: IF VAL (A$)
= 96 THEN 1160
1150 GOTO 1140
1160 PRINT D$"IN#0": PRINT
CHR$ (7): VTAB 13: CALL -
958: PRINT "FILE NAME:
TEXT.": INPUT"";A$
1170 PRINT D$"BSAVETEXT."A
$,A$1900,L"AD - 6399
1180 GOTO 1040

```

DOW JONES NEWS & QUOTES REPORTER

Selected Business News at the Touch of a Key

The Dow Jones News & Quotes Reporter is a powerful business software package, designed especially for investors, busy managers, and executives who need fast access to stock market news and information. It allows users to retrieve — over telephone lines — past and current news stories and headlines from The Dow Jones News/Retrieval Service, The Wall Street Journal, and Barron's, as well as quotations for more than 6000 securities traded on the major exchanges.

News & Quotes Reporter can also be used in conjunction with Portfolio Evaluator, another Dow Jones program from Apple. Portfolio Evaluator allows you to store, modify, and update approximately 100 individual

portfolios of up to 50 stocks each on a single diskette. This capability — coupled with News & Quotes Reporter's access to late-breaking financial news and stock quotations — provides the tools and information needed for sophisticated portfolio creation and management.

The Dow Jones News & Quotes Reporter is your personal key to making sound investment decisions in a rapidly changing world.

BENEFITS

The Dow Jones News & Quotes Reporter package...

- Saves time spent skimming and searching through business media, by putting financial headlines and news stories at your fingertips...

- Keeps you informed of stock price fluctuations, by providing fast access to stock quotations for more than 6,000 securities on all the major stock exchanges...
- Renders concise records for business or tax purposes through a monthly Dow Jones billing.

DOW JONES NEWS & QUOTES REPORTER — A CLOSER LOOK

The main menu of the Dow Jones News & Quotes Reporter contains five program selections. With News & Quotes Reporter, you can access, display, and print news headlines and entire stories from the worldwide network of the Dow Jones News/Retrieval Service, The Wall Street Journal, and Barron's. You can also access timely stock and composite quotes on more than 6,000 corporate stocks and bonds, options, mutual funds, and treasury notes and bonds on the New York, the American, the Mid West, and the Pacific stock exchanges, plus the over-the-counter market (OTC NASDAQ).

You can "log on" to the Dow Jones News/Retrieval Service over telephone lines, using either an autodial or acoustical modem and a special password, obtained from your dealer when you purchase Apple's Dow Jones News & Quotes Reporter. To access news items, select NEWS RETRIEVAL SERVICE from the main menu. Your Apple II System automatically initiates the log-on procedure, even automatically dialing the correct phone number if you have an auto-dial modem.

You can obtain news either by category or company. For example, you may be interested in up-to-the-minute foreign news — say, from the Mid-East — or in current stories concerning aerospace, mining, or a particular company in which you're interested. Using NEWS RETRIEVAL SERVICE, simply enter the appropriate symbol. Then choose the most recent news story, or the first page of subject-related headlines. There may be less than a page or many pages of headlines, listing some stories as far back as three months.

Using one of several suggested printers (see System Configuration), you can also print news stories in their entirety, or the full list of headlines (one page at a time). Simply type the "!" symbol, and the news item or list of headlines that you're reading on the video monitor will be printed.

Another selection option available from the main menu is STOCK QUOTE SERVICE, which allows you to access Dow Jones' securities quote data base and obtain current stock quotations. (To conform with federal regulations, stock quotations are delayed 15 minutes.) As it does with NEWS RETRIEVAL SERVICE, News & Quotes Reporter leads you step-by-step through the STOCK QUOTE SERVICE procedures.

Suppose, for example, that you're reading a current news story about a company in which you own stock, and you encounter a competitor's name. With the News & Quotes Reporter, it takes but a few keystrokes to return to the menu, select the Stock Quote

Service Program, and find the trading price of the competitor's stock.

The third option on News & Quotes Reporter's main menu, CUSTOMIZING FEATURES, allows you to print news stories either at 40 or 80 columns wide. You can switch easily from 80-column hardcopy for news stories, to 40-column hardcopy for the Dow Jones Industrial Averages. CUSTOMIZING FEATURES also lets you enter and maintain your password (supplied by Dow Jones), and enter and maintain two telephone numbers for use with an auto-dial modem.

Selection 4, DISCONNECT DOW JONES, simply disconnects your line from the Dow Jones phone number. With this option you "hang up" and don't continue paying connect-time charges, but your system is left still running, connected, and ready to log back on, as soon as you need News & Quotes Reporter again. Selection 5, EXIT NEWS & QUOTES REPORTER, terminates the News & Quotes Reporter program. If you're using an autodial modem, you will be reminded to unplug your line and reconnect it to your phone; with an acoustical modem, you'll be reminded to hang up your phone.

SYSTEM CONFIGURATION

To use the Dow Jones News & Quotes Reporter you will need:

- Apple II or Apple II Plus, each with a minimum 48K of memory; or
- Apple II with the Apple Language System;
- Apple Disk II with controller and 16-sector PROMs;
- Apple Modem IIB, with Apple Communications Interface Card;
- A video monitor or television;
- A standard, working telephone;
- A printer and interface card* (optional);

* Note: The Apple Computer System works with several printers and appropriate interface cards, including the following:

- Apple Silentype Printer Card: Silentype Interface Card (supplied with printer)
- Printer IIA Card: Centronics 779 Printer Interface Card (Apple Product A2B0007, included if printer is purchased from Apple Computer Inc.)
- Qume 5/45 Card: High Speed Serial Interface Card (Apple Product A2B0005) with P8-02 PROM

TECHNICAL SPECIFICATIONS

Language: Written in Pascal (Run Only)

Dow Jones Access: 15-minute tape delay; exchanges include NYSE, AMEX, Mid West, Pacific, Composite, and OTC NASDAQ.

NOTE: The cost of the Dow Jones News & Quotes Reporter package includes the one-time password fee necessary to use the Dow Jones News/Retrieval Service. In addition, there are also Dow Jones connect-time charges. If you already own Apple's Portfolio Evaluator program, then you've paid for and received a password that's usable with News & Quotes Reporter. Though you have to pay the password fee again as part of the News & Quotes Reporter package, the amount will be credited to your Dow Jones account. Your dealer can further explain these arrangements to you.

THE DOW JONES NEWS & QUOTES REPORTER PACKAGE Order No. A2D0030

With your News & Quotes Reporter order, you will receive:

- Apple's Dow Jones News & Quotes Reporter master diskette;
- Apple's Dow Jones News & Quotes Reporter back-up diskette;
- Apple's Dow Jones News & Quotes Reporter Instruction Manual;
- Dow Jones News/Retrieval and Stock Symbol Guide;
- Dow Jones News/Retrieval Service contract and password;
- One hour of free training time, to be used during non-prime time hours, within 15 days of purchase of the News & Quotes Reporter package.

CONVERTING STRINGS TO NUMERIC VARIABLES

by Jo and Charlie Kellner

Apple Pascal doesn't allow input editing for integers and reals. This can cause a lot of trouble if the wrong data is entered by mistake: either the program is stuck with bad data, or worse yet, the system may crash.

STRINGSTUF is an intrinsic unit which is designed to avoid this problem. All data is entered in the form of strings, then converted to the appropriate data format by the unit. This allows editing while the data is in string form.

Note: The STRINGSTUF unit and accompanying demo program are designed to run in the new release of Pascal, version 1.1 only. Long integers are not supported in STRINGSTUF.

THE UNIT

STRINGSTUF may be located in any unused segment number 17-31. Install in SYSTEM.LIBRARY, following the instructions in the Pascal reference manual.

```
(*S+*)
(*$LPRINTER:*)

UNIT STRINGSTUF; INTRINSIC CODE 26;

(* Copyright Apple Computer Inc. 1980 *)

INTERFACE

    TYPE STRING255=STRING[255];

    FUNCTION STRFP (VAR STR:STRING255; VAR FP:REAL): BOOLEAN;
    FUNCTION STRINT (VAR STR:STRING255; VAR INT:INTEGER): BOOLEAN;

IMPLEMENTATION

FUNCTION STRFP; (* String to Real *)

CONST MAXREAL=1.70E37; (* Max/10 *)
      MINREAL=1.2E-37; (* Min/10 *)

VAR DEC,DEX,EDP,INX,LEN: INTEGER;
    DP,EX,IM,MN,MX,SN: BOOLEAN;
    CH: CHAR;
    NUMERIC,EXPONENT,MODIFIER: SET OF CHAR;

PROCEDURE TERMINATE;
VAR I: INTEGER;
BEGIN
    IF MX THEN DEX:=-DEX;
    EDP:=EDP+DEX-DEC;
    IF EDP<0
    THEN FOR I:=1 TO -EDP DO
        IF FP>=MINREAL THEN FP:=FP/10.0
        ELSE FP:=0 (* Underflow => 0 *)
    ELSE FOR I:=1 TO EDP DO
        IF FP<=MAXREAL THEN FP:=FP*10.0
        ELSE EXIT (STRFP); (* Overflow *)
    IF MN THEN FP:=-FP;
    STRFP:=TRUE;
    EXIT (STRFP) (* Successful conversion *)
END;
```

```

PROCEDURE SEARCH;
BEGIN
  WHILE INX<=LEN DO
    IF STR[INX] IN NUMERIC
      THEN BEGIN
        (*$R-*)
          WHILE (INX>1) AND (STR[INX-1] IN EXPONENT+MODIFIER)
            (*$R+*)
              DO INX:=INX-1;
              EXIT(SEARCH) (* Found start of number *)
            END
          ELSE INX:=INX+1;
          EXIT (STRFP) (* Non-numeric string *)
        END;
      END;
    BEGIN (*STRFP*)
      NUMERIC:=[ '0'..'9' ];
      EXPONENT:=[ 'E', 'e' ];
      MODIFIER:=[ '+', '-', '.', ', ' ];
      DP:=FALSE; EX:=FALSE; IM:=TRUE;
      MN:=FALSE; MX:=FALSE; SN:=FALSE;
      DEC:=0; DEX:=0; EDP:=0; INX:=1;
      LEN:=LENGTH(STR); FP:=0;
      STRFP:=FALSE;
      SEARCH; (* Find start of number *)
      WHILE INX<=LEN DO BEGIN
        CH:=STR[INX];
        IF CH IN NUMERIC+EXPONENT+MODIFIER
          THEN BEGIN
            IF CH IN NUMERIC
              THEN IF EX
                THEN BEGIN
                  IF DEX<1000 THEN
                    DEX:=DEX*10+ORD(CH)-ORD('0'); (* Exponent *)
                  SN:=TRUE
                END
              ELSE BEGIN
                IF FP<1.0E8
                  THEN FP:=FP*10+ORD(CH)-ORD('0') (* Mantissa *)
                ELSE EDP:=EDP+1;
                IF DP THEN DEC:=DEC+1; (* Digits to right of DP *)
                IM:=FALSE;
                SN:=TRUE
              END
            ELSE CASE CH OF
              '+': IF SN THEN TERMINATE (* Duplicate '+' sign *)
                ELSE SN:=TRUE;
              '-': IF SN THEN TERMINATE (* Duplicate '-' sign *)
                ELSE BEGIN
                  IF EX THEN MX:=TRUE
                    ELSE MN:=TRUE;
                  SN:=TRUE
                END;
            END;
          END;
        END;
      END;
    END;
  END;

```

```

    '.': IF DP OR EX THEN TERMINATE (* Duplicate '.' *)
        ELSE DP:=TRUE;
    'E','e': IF EX THEN TERMINATE (* Duplicate 'E' *)
        ELSE BEGIN
            IF IM THEN FP:=1.0; (* Implied mantissa *)
            EX:=TRUE;
            SN:=FALSE
        END;
    END; (*CASE*)
    INX:=INX+1
    END
    ELSE TERMINATE (* End of number *)
    END;
    TERMINATE (* End of string *)
END;

```

```

FUNCTION STRINT; (* String to Integer *)

```

```

VAR FP: REAL;

```

```

BEGIN

```

```

    STRINT:=STRFP (STR,FP); (* First convert to real *)

```

```

    IF ABS(FP)<=MAXINT

```

```

        THEN INT:=ROUND(FP) (* then round to integer *)

```

```

    ELSE BEGIN

```

```

        STRINT =FALSE; (* Integer out of range *)

```

```

        INT:=0

```

```

    END

```

```

END;

```

```

BEGIN (* Unit Initialization *)
END.

```

```

PROGRAM STRINGTEST;

```

```

USES STRINGSTUF; (* tests StringStuf library unit *)

```

```

VAR INPUT,STR: STRING;

```

```

    INT: INTEGER;

```

```

    FP: REAL;

```

```

BEGIN

```

```

    PAGE (OUTPUT);

```

```

    WRITELN ('STRINGSTUF STRING => NUMERIC CONVERSION:');

```

```

    REPEAT

```

```

        WRITELN;

```

```

        WRITE ('STRING : ');

```

```

        READLN (INPUT);

```

```

    (*$V-*)

```

```

    IF STRFP (INPUT,FP) THEN

```

```

        BEGIN

```

```

            WRITELN (' REAL: ',FP);

```

```

            IF STRINT (INPUT,INT)

```

```

                THEN WRITELN ('INTEGER: ',INT)

```

```

                ELSE WRITELN('INTEGER: OUT OF RANGE.')

```

```

            END

```

```

            ELSE WRITELN('NO NUMERIC VALUE IN STRING.');
```

```

    (*$V+*)

```

```

    UNTIL INPUT=''

```

```

END.

```

THE DEMO

This program illustrates the use of STRINGSTUF. The compiler \$V-option is required to override the normal string length checking.

PASCAL RUN-TIME ERRORS

by Jo Kellner
Apple Computer, Inc.

Run-time errors generate a message plus a set of numbers that indicate which instruction was being executed when the error occurred. "S" stands for "SEGMENT", "P" for "PROCEDURE", and "I" for "INSTRUCTION COUNT". These can be correlated with the textfile by using the System List option (*\$L<filename>*) when doing a compile. This will produce an annotated listing of the text, with the S, P, and I numbers included.

For example, here is a very simple program which has been compiled with the listing option:

A word of warning: do NOT use L+ as the option, as this will result in the loss of your code file and possibly your operating system. Instead, name a disk file that will be placed on a different volume from the destination of the code file, or better yet, use (*\$LPRINTER:*) to put the listing directly onto your printer.

(1)	(2)	(3:4)	(5)<--TEXT----->
1	1	1:D	1 (*\$LPRINTER:*)
2	1	1:D	1 PROGRAM EXAMPLE;
3	1	1:D	3
4	1	1:D	3 VAR S:STRING;
5	1	1:D	44
6	1	1:0	0 BEGIN
7	1	1:1	0 READLN(S);
8	1	1:1	21 WRITELN(S)
9	1	1:0	40 END.

KEY:

- (1) Line number of text
- (2) Segment number (S#) — S# values which do not appear in your listing indicate that the error has occurred in that segment of the operating system. S#0 is SYSTEM. PASCAL, and 17-31 are usually SYSTEM.LIBRARY segments.
- (3) Procedure number (P#) — the procedure number within the segment designated by the S#.
- (4) Nesting level (D=declaration)
- (5) Instruction count (I#) — this is the instruction count from the beginning of the procedure. The number indicates the count at the beginning of each line, so a value between two lines simply means that the error occurred in the middle of the line.



APPLE ORCHARD SUBSCRIPTIONS

P. O. BOX 2227 SEATTLE, WASHINGTON 98111, USA

The International Apple Core will make individual subscriptions to "The Apple Orchard" available commencing with Volume I, Number 2 to be published in September, 1980.

NAME _____

STREET _____

CITY _____ STATE _____ ZIP _____

COUNTRY _____

Annual Subscription Rate: \$10.00 per year
First Class Postage: \$5.00 per year additional (required for Canada, Mexico, APO, and FPO addresses)
Overseas and other foreign air mail postage (required): \$10.00 per year additional

TOTAL REMITTANCE ENCLOSED: \$(USA) _____

Make check or money order payable to "International Apple Core" and return with this form to:
Apple Orchard Subscriptions

P.O. Box 2227

Seattle, Washington, USA 98111

Saving and Loading Arrays in Applesoft

Have you ever encountered an application in which you've had a large array filled with valuable data and then you wanted to save that array to the disk? Those of you who have tried to save the array, element by element, using Applesoft BASIC, have probably wondered if there might be a faster way to get the array on and off the disk. This article will discuss a method for using BSAVE and BLOAD to store arrays on disk and retrieve them for use with Applesoft programs, without trashing your variables or the program.

Numeric Arrays

The following subroutine will BSAVE or BLOAD an array of real numbers in Applesoft. It uses two variables (A and N\$) which will contain the starting address of the array and the name of the array to be saved to disk.

```

1000 A=0: GOSUB 1060
1010 PRINT CHR$(4); "BSAVE
      ARRAY";N$;" ,A";A;" ,L";PEEK
      (A+2) + PEEK (A+3) * 256
1020 RETURN
1030 A=0: GOSUB 1060
1040 PRINT CHR$(4); "BLOAD
      ARRAY";N$;" ,A";A
1050 RETURN
1060 A=PEEK(107) + PEEK(108)
      *256
1070 IF LEN (N$) = 1 THEN 1090
1080 IF PEEK(A+1)<>ASC
      (MID$ (N$,2,1)) THEN 1110
1090 IF PEEK(A)<>ASC (LEFT$
      (N$,1)) THEN 1110
1100 RETURN
1110 A = A + (PEEK(A+2) +PEEK
      (A+3) *256)
1120 IF A < (PEEK(109) + PEEK
      (110) *256) THEN 1070
1130 PRINT N$;"NOT FOUND":
      RETURN
  
```

To BSAVE an array of real numbers from your program you would set N\$ equal to the name of the array you want to save, then GOSUB 1000. Line 1000 sets the variable 'A' to zero to insure that space is allocated for the variable

'A'. If this is not done, then when line 1060 is executed, the starting address of the array space will be seven bytes off. This is because the value for the variable is calculated before memory is allocated for the variable. After memory is allocated for the variable the value is stored in memory. But by this time Applesoft memory has been shifted around to make room for the new variable. Therefore, 'A' is made equal to zero so that space is allocated for the variable before the starting address of the array variable space is calculated.

Line 1060 calculates the starting address of the variable array space in memory and stores this value for future reference in the simple variable 'A'. A simple variable is one that contains a single numeric value or, in the case of simple string variables, one string of characters.

Applesoft arrays have well defined structures in memory. A description of this structure can be found on page 137 of the "Applesoft II BASIC Programming Reference Manual". The address calculated in line 1060 is the starting address for all of the arrays in memory. To find the starting address of the array named in N\$, the above subroutine must look through memory to find the name that matches the name in N\$. All variable arrays in Applesoft are stored in one continuous chunk of memory, starting at the address contained in location 107 and 108. Each array in this continuous memory space is stored one right after another. When a new array is DIMensioned, this continuous chunk of memory is expanded to make room for the new array. Each array in memory contains information about itself. All the information our routine needs is in the first four bytes of any Applesoft array. The first two of these four bytes contains the name of the array, and the next two bytes contain the length of that array.

Line 1070 through 1090 compare the first two characters of the name in N\$ to the array name at the memory location stored in the variable 'A'. Line 1070 ensures that only the necessary characters in N\$ are compared with their corresponding values in the first two bytes of the array in memory. If either of the characters in N\$ do not match the ASCII values in the first two bytes of the array, then the array we are looking for has not been found. This is when the length bytes of the array in memory come in handy. To find the length of the array the subroutine PEEKs the third and fourth memory locations beyond the address in the variable 'A'. Line 1110 uses the following function to find the length of the array.

PEEK(A+2) + PEEK(A+3) *256

This result is then added to the value in 'A' to yield the address of the first byte of the next array in memory.

If the name of the array at the address in 'A' does not match the name in N\$ then control of the routine is transferred to line 1110. Line 1110 changes the address in 'A' to the first byte of the next array in memory. Then the name of this array can be compared to the name in N\$ and the search will continue until all of the arrays in memory are examined or the array in N\$ is found.

The chunk of memory for array storage has a limit, and it doesn't do any good to check for arrays beyond this limit. As it turns out, the last address of the array space in memory is stored in zero page locations 109 and 110. If the address in the variable 'A' is compared to the address in 109 and 110 then we have a way to stop the routine when an array name is not in memory. This in fact is the reason for including line 1120. As long as the address in 109 and 110 is greater than the array address stored in 'A' the routine continues to search for the array name in N\$.

When the array is found in memory, program control is transferred to the next statement following the GOSUB that called the search routine to start with. If a GOSUB 1000 was performed then the array in memory is BSAVED. If a GOSUB 1030 is performed then a

BLOAD overwrites the array in memory.

Assumptions and Cautions:

The above routine assumes that the name of the array has been placed in N\$ before a GOSUB 1000 or GOSUB 1030 is performed. If the array is to be BLOADED, then it must be DIMensioned (using the DIM statement) to the same size as the array that was BSAVED. In other words if the array A(10,20) was saved to disk and it is to be loaded into memory again, then the array must be DIMensioned as A(10,20) in the program before loading the array. The routine listed above also assumes that the array being loaded or saved is a real number array. To load and save integer arrays, make the following modification to lines 1080 and 1090.

```
1080 IF PEEK(A+1)<>(ASC
(MID$(N$,2,1)))+128
THEN 1110
1090 IF PEEK(A)<>(ASC (LEFT$
(N$,1)))+128 THEN 1110
```

With the above changes, the routine will load and save integer arrays only.

The following program illustrates the use of this subroutine for BLOADing arrays.

```
10 DIM A(10,20),CV(50)
20 FOR I = 0 TO 10
30 FOR J = 0 TO 20
40 A(I,J) = J + I
50 NEXT J
60 NEXT I
70 FOR I=0 TO 50
80 CV(I) = I
90 NEXT I
100 N$ = "CV"
110 GOSUB 1000: REM SAVE
THE ARRAY CV(50)
120 N$ = "A"
130 GOSUB 1000: REM SAVE
THE ARRAY A(10,20)
140 END: REM REAL NUMBER
ARRAY SUBROUTINE
STARTS HERE
1000 A = 0: GOSUB 1060
1010 .
```

```
1130 PRINT N$; " NOT FOUND":
RETURN
```

When this program is run, the arrays 'A' and 'CV' will be saved to

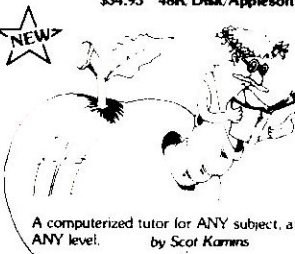
the disk. To load the real number arrays back into memory, enter the following program.

```
10 DIM CV(50), A(10,20)
20 N$ = "A"
30 GOSUB 1030: REM LOAD
ARRAY "A"
40 FOR I = 0 TO 10
50 FOR J = 0 TO 20
60 PRINT A(I,J)
70 NEXT J
80 NEXT I
90 N$ = "CV"
100 GOSUB 1030: REM LOAD
ARRAY CV
110 FOR I=0 TO 50
120 PRINT CV(I,J)
130 NEXT I
140 END: REM REAL NUMBER
ARRAY STARTS HERE
1000 A = 0: GOSUB 1060
1010 .
.
.
1130 PRINT N$; " NOT FOUND":
RETURN
```

When this program is run, it will print the contents of the real number arrays CV and A.

STONEWARE MICROCOMPUTER PRODUCTS
1930 Fourth Street, San Rafael, CA 94901 (415) 454-6500

Aristotle's Apple
\$34.95 48K Disk Applesoft



- 2 modes of instruction tutor and test.
- 3 quiz types fill in, multiple choice, and matching, including alternate answers for fill-in questions.
- Stores quizzes on disk for fast, easy access.
- Multi-level learning reinforcement. Written by a specialist in Computer Aided Instruction (CAI).
- Highly interactive no programming knowledge necessary.
- Good for students, home study and correspondence courses, government and ham radio exams, etc.

A computerized tutor for ANY subject, at ANY level.
by Scot Kamms

MICRO MEMO \$39.95 48K DISK Applesoft
by Barney Stone
A powerful, easy to use appointment calendar.



- Includes one-time, weekly, monthly, semi-annual and annual memos.
- Will remind you one week, two weeks or a month in advance to prepare for meetings, make reservations, buy birthday presents, etc.
- Display or print any day's or week's reminders.
- A "perpetual" calendar holds one full year, beginning with any month. Automatically posts birthdays, etc., into new months.
- Knows most major holidays.
- Supports Mt. Hardware Apple Clock (not required).

Calif. Residents Add 6% Sales Tax. No C.O.D.'s. Add \$2.00 for Shipping & Handling. Use Check, Money Order, VISA or MASTERCARD. (We need expiration date on charge card.) DEALER INQUIRIES INVITED. APPLE II is a registered trademark of Apple Computer, Inc.

Serious SOFTWARE for your APPLE!

EASYWRITER , the best-known, best-loved Word Processor by CAP'N SOFTWARE	\$99
EASYWRITER , the Professional Word-Processing System with 80-column Screen-Hardware by INFORMATION UNLIMITED SOFTWARE/CAP'N SOFTWARE	\$250
EASYMAILER , the mailing-list and record-management system Use it to interface and merge lists with the Word-Processors above	\$69
WHATSIT? the Self-Indexing Query System by COMPUTER HEADWARE	\$135
VISCALC , the numerical-modeling and calculating tool by PERSONAL SOFTWARE	\$135
CCA , the complete Data Management System by PERSONAL SOFTWARE. contains mailing-lists, report-generation & program-interfacing features	\$99
STOCK MARKET ANALYSIS by GALAXY	\$49
SUPER-CHECKBOOK does check reconciliation and analysis, by POWERSOFT	\$25
APPLE-DOC , documents your BASIC programs, by SOUTHWEST DATA SYSTEMS	\$25
MULTI-MESSAGE , allows you to create large, colorful, hi-res messages on multiple TV sets. Broadcasts ads from shop-windows & trade-show booths. By CONNECTICUT INFO. SYSTEMS	\$35
APPLE DATA-GRAPH , plots up to 3 hi-res curves, 40 points each, on a graph with dimensioned axes. Graphs can be saved to disk & recalled instantly! By CONNECTICUT INFO. SYSTEMS	\$25
VERBATIM , 5 1/4-inch diskettes, per box of 10	\$29

* Either MULTI-MESSAGE or APPLE DATA-GRAPH are FREE with each \$100 of any merchandise ordered. *

HARDWARE for your APPLE!

D.C. HAYES MICROMODEM	\$300
MICROSOFT Z-80 SOFTCARD , lets you run CP/M programs on APPLE!	\$275
MOUNTAIN HARDWARE: APPLE CLOCK	\$225
ROMPLUS	\$169
CALIFORNIA COMPUTER SYSTEMS: PARALLEL INTERFACE	\$119
ASYNCHRONOUS SERIAL INTERFACE	\$159
SSM AIO SERIAL-PARALLEL INTERFACE works with almost all popular printers!	\$175
APPLE LANGUAGE SYSTEM with PASCAL	\$397

Contact us for help in choosing items. We like to assist. C.O.D.'s Welcome!

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

THE KEYPRESS FUNCTION

KEYPRESS is a widely used function which resides in the unit APPLESTUFF. In many cases, it is the only routine being called from the unit, so it would be more efficient to refer to the routine without also "using" the rest of the unit.

The following listing is the function "KEYPRESS", which can be assembled for linkage to your host program as an "external" function. Follow the instructions provided with the example listed on pages 100 - 104d of the Pascal reference manual.

```

        .FUNC KEYPRESS,0      ;0 words of parameters passed
;*****
;*
;*  FUNCTION KEYPRESS: BOOLEAN; EXTERNAL
;*
;*****
;
RETURN  .EQ 0                ;Storage for return address
CONCKVEC.EQ 0BF0A           ;Fixed address in BIOS
RPTR    .EQ 0BF18           ;Fixed buffer pointer
WPTR    .EQ 0BF19           ;Fixed buffer pointer
VERSION .EQ 0BF21           ;System version number
KEYBOARD.EQ 0C000          ;Keyboard hardware
CONCK   .EQ 0FF5C           ;Way to get CONCK in old system

        PLA
        STA RETURN
        PLA
        STA RETURN+1
        PLA
        PLA
        PLA
        PLA
        PLA                ;Pop 4 bytes stack bias for function
        LDA #0
        PHA                ;Return MSB zero
        LDA KEYBOARD
        BMI TRUE
        LDA VERSION
        BNE $1              ;Jump if not original Pascal version
        JSR CONCK
        JMP $2
$1      JSR CONCKVEC        ;Check console
$2      LDA RPTR
        CMP WPTR           ;Char in buffer?
        BEQ EMPTY
TRUE     LDA #1             ;Yes, return KEYPRESS=TRUE
        BNE KPDONE        ;Always taken
EMPTY    LDA #0            ;No, return KEYPRESS=FALSE
KPDONE   PHA               ;Push LSB result
        LDA RETURN+1
        PHA                ;Restore return address
        LDA RETURN
        PHA
        RTS
        .END

```

THE DEMO

This brief program illustrates the use of KEYPRESS as an externally linked routine. Follow the instructions given on pages 100 - 106 of the Pascal reference manual for assembling and linking external code.

```

PROGRAM PRESSTEST;
VAR I: INTEGER;
FUNCTION KEYPRESS:
    BOOLEAN; EXTERNAL;
BEGIN
I:= 0;
REPEAT
    WRITELN (I);
    I:=I+1;
UNTIL KEYPRESS
END.

```

THE TAX PLANNER — A Personal Financial Planning Tool

The Tax Planner is an innovative program designed for use with Apple II or Apple II Plus computers. It allows you to determine the federal income tax advantages or liabilities that result from personal financial decisions. You can construct various income scenarios and compare the tax impact of each. This means you can determine the best time to make a financial move (for example, sell property or take gains or losses on investments). And, you can print out your tax strategies, or store them on a diskette, ready for fast retrieval and modification.

In short, the Tax Planner is a tool that can help you manage your personal income through better planning of your tax liability.

BENEFITS

The Tax Planner. . .

- Provides you with the capability to optimize the federal income tax advantages available to you, because it lets you quickly compare the tax consequences of various financial decisions.
- Assists you in choosing the best time to make certain financial moves by instantly computing their impacts on income taxes in current and future years.
- Helps clarify current federal tax regulations, because you can experiment with various financial scenarios, and observe how the tax mechanisms apply to each.
- Increases your productivity in conducting financial analysis and making financial decisions, by allowing you to quickly develop or change tax strategies.

THE TAX PLANNER — A CLOSER LOOK

The Tax Planner main menu offers four options:

1. TAX PLANNING - by which you plan your tax strategy.
2. FILE CLERK - by which you manage the diskette storage of your various income and tax scenarios.
3. PRINTER - by which you print

the results of a tax planning scenario.

4. QUIT - by which you exit from the Tax Planner.

To begin using the program, you'd select TAX PLANNING. Before tax computations can be performed, you will be asked to provide some information, including how far into the future you wish to plan (up to nine years), and your base period income for each of the four preceding years. If you choose to plan only for next year, you may specify the number of alternative scenarios to be examined for that year. After you've provided this information, your screen will display the Tax Matrix, a skeletal array of the items which affect your federal income tax situation. These include your tax filing status, income, long- and short-term capital gains or losses, deductions, and adjustments.

Suppose you were planning your 1980 tax situation, plus two years into the future. On the screen, the Tax Matrix will display three columns, labelled 1980, 1981, and 1982. You then provide input information on your projected income and deductions for these years. Simply enter the information into the Tax Matrix on the screen. Once you're satisfied with your projections, request that the Tax Planner compute adjusted gross income, estimated tax, most advantageous tax computation method, and tax bracket — for each of the three years. At any point you may alter your projections and instantly see their consequences.

Setting up the Tax Matrix usually requires some time and careful thought. But once it's complete, the Tax Planner does all the rest of the work — the tedious tax computations — for you. The Tax Planner automatically computes your tax by income averaging as well as by the regular method. It also computes the maximum tax on personal service income, the minimum tax on preference items,

and the alternative minimum tax. In addition, your tax bracket and the tax computation method most advantageous to you are determined for each year or alternative within a single year.

With Tax Planner, you can use the Tax Matrix to quickly perform "what if" experiments that otherwise would consume hours of calculating. Suppose you're thinking of selling your home, and wonder if it would be more advantageous in tax savings to sell it this year rather than next. Simply include the expected sale price in this and next year's cell of the Tax Matrix, and let Tax Planner compute the results in each case. A quick comparison will reveal the more advantageous time for selling.

The Tax Planner incorporates current federal tax laws. In addition, if you fill out and mail the Registration Card that accompanies the program, Apple will keep you informed of any product changes corresponding to tax law revisions.

SOME TECHNICAL INFORMATION

The Tax Planner is written in the Pascal language, but your Apple does not require the Language Card because the program is written in "run-time" Pascal.

For most users, a single Tax Planner diskette can store about 40 tax planning scenarios. Of course you may change or delete any stored scenario whenever you wish.

THE TAX PLANNER PACKAGE

- Order No. A2D0040

With your Tax Planner order, you will receive:

- Two (2) Tax Planner diskettes (a master and a backup).
- The Tax Planner manual.
- User registration card.
- Customer license agreement.

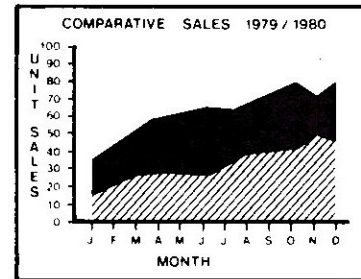
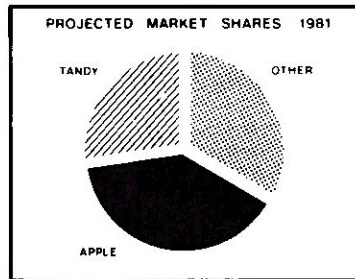
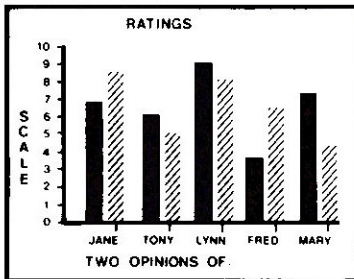
SYSTEM CONFIGURATION

- An Apple II or Apple II Plus, each with a minimum of 48K of memory;
- one Apple Disk II with controller and 16-sector PROMs;
- a video monitor or television;
- a printer and interface card (optional — you can still use the Tax Planner without using the PRINTER menu option).

Business & Professional Software, Inc.

introduces

appleGraph™



for use with Apple* Computer Systems

appleGraph is a high-quality microcomputer software package for general purpose plotting of data in a variety of formats for use by the business, professional, and research decision maker.

appleGraph yields high-resolution, multicolor graphics for video display and hardcopy output. Featured are pie charts generated separately or bar graphs, area plots, points, and solid or dashed lines produced in any combination of overlays.

appleGraph features *easy-to-use* English language commands, making it readily accessible to those unfamiliar with computers, as well as those who wish to add to their selection of decision-making tools.

Commands may be entered interactively or in advance for automatic presentation of an entire data analysis complete with mathematical manipulation, curvefitting, smoothing, and simple statistics.

Data may be entered directly or supplied from other programs.

appleGraph's many applications include:

- a forecasting tool for business and professional decision makers
- a teaching aid for educators
- a data analysis tool for researchers
- a visual aid for presentations

appleGraph will be available in January 1981.

For further information contact your local Apple dealer, or write:

Business & Professional Software, Inc.
 238 Main Street, Cambridge, MA 02142
 Telephone (617) 491-3377

Dealer and OEM inquiries invited.

*Apple is a trademark of Apple Computer, Inc.

APPLEWRITER AND DOS TEXT FILES

by John Crossley and Jim Hoyt
Apple Computer, Inc.

Now that Applesoft has been out awhile it occurs to me that it could be used for more than just writing letters. As a first step in this direction I thought that it would be nice to use Applesoft to edit text files and print them out nicely formatted.

When Applesoft saves a file to the disk, it saves an image of what's in memory with a BSAVE. This format is totally incompatible with DOS text files. So the easiest way to transfer is to READ the STET file one character at a time into memory and BSAVE it the way Applesoft does.

The Apple II slightly modifies the ASCII for its internal use by having the most significant bit on. Applesoft modifies this even more. The upper case letters are moved down to the inverse range, lower case letters are moved to where upper case letters belong and numbers and punctuation are moved to above either upper or lower case letters. Most text files contain straight or normal Apple ASCII so there is some code conversion required during the transfer. Since the transfer has to take place one character at a time, this doesn't cause any problems.

The easiest way to work with DOS and put up pretty screens for the user is with a BASIC program. The easiest and quickest way to re-map character sets is in machine language. So I chose to use a mix of the two. The Applesoft portion does the one-time job of the appropriate DOS commands and user prompts, and the machine language portion does the character transactions.

The normal way to read a text file into a BASIC variable is to OPEN, READ, and then INPUT the variable. That's exactly what I've done in the first program except instead of using the INPUT statement, I have substituted a machine language routine to convert the ASCII to Applesoft ASCII and store the converted character in memory. This continues until the routine requests one more character than there is in the file and DOS gives an OUT OF DATA error.

One of the first things the Applesoft portion does is to set up an ONERR routine. That routine gets control when the OUT OF MEMORY error happens. The ONERR routine CLOSEs the text file and POKEs an Applesoft end of file marker (\$60.96 decimal) behind the last character in memory and BSAVEs the entire file to the disk. The Applesoft file is built where Applesoft would normally put it so you can quit at this point, run Applesoft, and start editing.

The program, as written, names the Applesoft the same name as the text file with 'TEXT.' prefixed.

This makes it easy to tell what files got converted to what new files.

```

100 HIMEM: 4096:TEXT : HOME
110 LET D$ = CHR$(4)
120 PRINT "**** ASC TEXT TO
    TEXT EDITOR MOVER"
130 PRINT "*****": PRINT "*****
    COPYRIGHT 1979, APPLE
    COMPUTER"
140 PRINT "*****": PRINT "*****
    DEVELOPED 1979, (J.H. &
    J.C.)
150 GOSUB 1000
160 VTAB 10: INPUT "TEXT FILE
    TO TRANSFER: ";N$: IF LEN
    (N$) = 0 THEN VTAB 22: END
190 VTAB 12: CALL - 958: INPUT
    "FROM WHICH DRIVE:
    ";IN$:IN = VAL (IN$): IF IN > 2
    THEN 190
200 IF (IN) THEN PRINT D$"
    OPEN "N$",D":IN: GOTO 220
210 PRINT D$"OPEN "N$
220 ONERR GOT 320
240 PRINT D$"READ "N$
260 CALL 768
320 PRINT D$"CLOSE "N$
330 POKE 216,0
335 POKE 6400,131
340 LET E = PEEK (30) + PEEK
    (31) * 256
345 POKE E,96
350 LET A$ = "" + N$ + "
    TRANSFERRED "
360 VTAB 18: HTAB 20 - INT (
    LEN (A$) / 2): INVERSE :
    PRINT A$:NORMAL
370 PRINT
400 PRINT D$"BSAVE TEXT."N$;
    ",A$1900,L"E - 6400 + 1
420 VTAB 18: CALL - 958
430 PRINT " T)RANSFER
    ANOTHER, E)DIT, Q)UIT"
440 PRINT
445 PRINT "WHICH?";
450 GET A$
460 IF A$ = "T" THEN RUN
470 IF A$ = "E" THEN PRINT :
    PRINT D$"RUN HELLO,D1"
480 IF A$ <> "Q" AND ASC (A$)
    <> 13 THEN 450
485 PRINT
490 END
1000 FOR I = 768 TO 808
1010 READ J: POKE I,J: NEXT :
    RETURN
1020 DATA 169,1,133,30,169,25
1030 DATA 133,31,32,12,253,160
1040 DATA 0,9,128,201,141,240,
    12
1050 DATA 201,160,144,241,201
1060 DATA 224,144,2,233,97,105
1070 DATA 64,145,30,230,30,208
1080 DATA 227,230,31,208,223
    
```

- UC Upper Case letters
- lc Lower Case letters
- (#) Lower Case letters in the Apple display as numbers
- # Numbers and punctuation
- ctrl Control characters
- cur The Applesoft cursor

	0,1	2,3	4,5	6,7	8,9	A,B	C,D	E,F
NORMAL ASCII	ctrl	#	UC	lc				
APPLE ASCII					ctrl	#	UC	(#)
APPLEWRITER	UC	UC#		cur	CR		lc	lc#
UC	Upper Case letters							

POKE SALAD

by Tom Brown
Birmingham Apple Peel

Using Applesoft ROM routines.

The first issue of Apple Orchard contained something I had been waiting for for a long time—Applesoft internal entry points. These are entry points to machine language routines within Applesoft (ROM card) for floating point operations, string manipulations, HI-RES graphics to name a few. After all, Applesoft is simply a giant machine language program. Most of these routines end in RTS (similar to RETURN in BASIC) so they can be used as subroutines from your own machine language programs. The following program is a result of my exploring these routines: a machine language dollar formatter.

When you want to print a number in dollar format from BASIC, you simply precede it with an "&": eg. 10 &X : &Y : &Z. The numbers are printed out as a block of 10 characters and/or spaces, right justified, with a dollar sign and leading/trailing zeros as necessary. Negative numbers are also allowed.

Here's how it works. When Applesoft encounters an '&', it causes an immediate jump to location \$3F5. At this location we have stored the instruction "JuMP \$300" which is the address of the formatter routine. This instruction is placed there by line 10 of the BASIC demo program. (This needs to be done only once). Now the formatter has control. FRMNUM is a routine which evaluates the number or expression (eg. X+3/A) and places it in the Applesoft floating point accumulator. (By the way, the beauty of using these subroutines is that you don't have

to know how they work. You just set things up, do a JSR-Jump SubRoutine—and like magic, the operation has been performed). Next, we round off to two decimal places: multiply by 100, add 0.5, take the integer, and divide by 100 (lines 25-30). Note the routines used: MUL10 (mult. by 10), FADDH (add 0.5), INT (INTEGER), and


DIV10 (div. by 10). Next, FOUT converts the floating point representation of the number to an ASCII string and stores it beginning at \$100. This happens any time you print a number from Applesoft. Lines 32 to 90 format the string, and finally at line 91 we JSR STROUT ("string output") which is a routine that prints a string pointed to by Y,A. Note that in lines 95 and 96 we load Y with \$1 and load A with \$00 which will point STROUT to \$100 — the beginning of our string.

Of course, you could do all of this from within BASIC, but this routine is *much* faster (in some cases, faster than BASIC can print them unformatted!) and causes no "garbage" string buildup.

Try the following BASIC demo:

```
10 POKE 1013,76: POKE
1014,0: POKE 1015,3:REM
SET UP'&VECTOR
20 FOR I=-10 TO 10 STEP .1
30 &I
40 NEXT
```


(Note: the numbers may range from 999,999.99 to -99,999.99)



1930 Fourth Street, San Rafael, CA 94901 (415) 454-6500


Finally... The Hi-res Baseball that's as good as the Apple!
by Arthur Wells

\$24.95, 32K Disk, Applesoft or Integer



Micro-League
Baseball

- 8 different pitches, 6 different swings
- 3 D effect on fly balls
- Player controlled fielding and throwing
- Vocal umpire
- Complete electronic score board
- Beautiful stadium in full color




A great hi-res lunar lander, just like the arcade game!
by Bill Budge creator of Trilogy and Penny Arcade

\$24.95, 48K Disk, Applesoft or Integer

- Landscape scrolling
- Auto-zoom for landing site close-up
- Player control of 360° craft rotation
- Spectacular crashes
- Always challenging

Improve your scores as you improve your skill!



Calif. Residents Add 6% Sales Tax. No C.O.D.'s. Add \$2.00 for Shipping & Handling. Use Check, Money Order, VISA or MASTERCARD. (We need expiration date on charge card.) DEALER INQUIRES INVITED.

APPLE II is a registered trademark of Apple Computer, Inc.



presents

SOFTWARE YOU CAN COUNT ON

You depend on good software to save you time and to have your computer help you do a job more efficiently. Our software is designed to do just that. We are one of the oldest companies supplying software for the Apple II*, and one of the very few that offers an unconditional guarantee of satisfaction or your money back! Here are a few that you'll want to add to your library:

Super Terminal Software

ASCII EXPRESS II: The most complete communications package available for the Apple II. Designed for the most efficient transfer of data to or from practically **any** online computer. Fully supports upper/lower case, including characters normally unavailable: underscore, rubout, break, and most others. Keyboard **macros** allow you to define dual keystrokes as entire strings for fast sign-ons, sign-offs, and system commands. A 20K data buffer allows for large files, and a convenient line editor means easy editing before and after transfer. Buffer can be output to printer, disk, or viewed at any time. Supports Micromodem II* and most other communication devices.

Price: \$64.95 on Disk. (Pre-Jan. 1 Price: \$59.95)

And for the Z80 Apple...

Z-TERM: A flexible communications package for the Apple II equipped with Z80 Softcard* and the CP/M* environment. Allows file transfers to or from all types of dial-in systems. Fully supports Micromodem II and most other communication devices, as well as 80 column display boards and external terminals! Utilizes standard CP/M sequential text files, with up to a 40K internal buffer (using additional RAM or Language Card.) Supports multiple modes of data transfer and includes keyboard **macros**, auto-dial (with Micromodem II), and upper/lower case.

Price: \$74.95 on 16 sector diskette. (Available Jan. 1, 1981)

Also available...

APPLE-DOC: A set of several utilities to speed up software development and customization. **Vardoc** makes a list of all the variables in a program and every line on which they occur. Also allows you to create a list of descriptors of what each one does. **Linedoc** makes a similar list for each line/subroutine called by a GOTO, GOSUB, etc. **Condoc** is similar but documents all numeric constants — great for scientific & business uses! **Replace** is a powerful replacement editor which makes changing any occurrence of a variable or group of statements a breeze!

Price: \$34.95. Disk. (Available Jan. 1, 1981)

THE CORRESPONDENT: An extremely versatile program! Designed primarily for writing letters and other documents in a very visual way. The Apple screen acts as a "window" onto a 40-80 column page. 4-directional scrolling lets you see any part of the page just as it will be printed. Editor functions include full upper/lower case & control chars., block move/copy, split screen option, even math functions! Additional utilities & uses include printing form letters, a free-form database, putting bi-directional scrolling in your own programs, single-disk copy program, DOS remove for greater storage on diskettes, and more!

Price \$44.95 on disk.

All programs require 48K and Applesoft in ROM or language card. Specify DOS 3.2 or 3.3. California residents add 6% to all prices.

See these and other S.D.S. products at your local dealer, or for more information, write or call:

southwestern data systems
P.O. Box 582-I • Santee, CA 92072 • (714) 562-3670

*Apple II is a registered trademark of Apple Computer Co
*Micromodem II is a registered trademark of Hayes Microcomputer Products, Inc.
*Z80 Softcard is a registered trademark of Microsoft Consumer Products, Inc.
*CP/M is a registered trademark of Digital Research, Inc

```

*****
4  *
5  * DOLLAR FORMATTER & T A BROWN *
6  *
7  *   COPYRIGHT (C) 1980   *
8  * ALL COMMERCIAL RIGHTS RESV'D *
9  *
*****
11 *
12 * PROGRAM IS RELOCATABLE
13 * REMEMBER TO SET "&" VECTOR TO
14 * FIRST BYTE OF THIS ROUTINE.
15 *
16         ORG $300
17         OBJ $6300
18 *
19 *
20 SIGN     EQU $6
21 STROUT   EQU $8B3A
22 FROMRM   EQU $8D67
23 FADDH    EQU $E7A0
24 MUL10    EQU $EA39
25 DIV10    EQU $EA55
26 INT      EQU $EC23
27 FOUT     EQU $EB34
28 *
29 * ROUND OFF
30         JSR FROMRM
31         LDA $A2
32         STA SIGN
33         JSR MUL10
34         JSR MUL10
35         JSR FADDH
36         JSR INT
37         JSR DIV10
38         JSR DIV10
39         JSR FOUT
40         LBA $600
41         STA $10A
42 *
43 * FORMAT
44         TAX
45         LDA $100,X
46         CMP $630
47         BEQ ZERO
48         LDY $FF
49         LDX $FF
50 LOOP3    INX
51         LBA $100,X
52         BEQ PAD
53         CMP $62E
54         BNE LOOP3
55         DP
56         INX
57         LDA $100,X
58         BNE DP

```



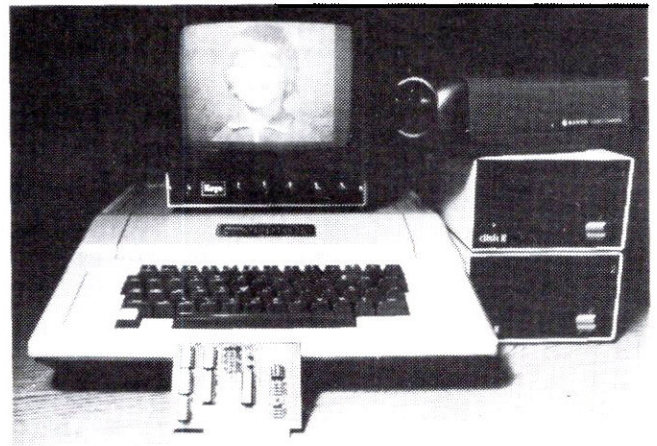
```

033E: 88      59  PAD      DEY
033F: F0 13   60          BEQ  PAD1
0341: 10 1A   61          BPL  MOVE
0343: A9 30   62  PAD2    LDA  #$30
0345: 8D 09 01 63          STA  $109
0348: 8D 08 01 64          STA  $108
034B: A9 2E   65          LDA  #$2E
034D: 8D 07 01 66          STA  $107
0350: A0 06   67          LDY  #$06
0352: 10 12   68          BPL  MOVE1
0354: A9 30   69  PAD1    LDA  #$30      #PADO
0356: 8D 09 01 70          STA  $109
0359: A0 08   71          LDY  #$08
035B: 10 09   72          BPL  MOVE1
73      * MOVE TO END OF BUFFER
035D: A0 0A   74  MOVE    LDY  #$0A
035F: BD 00 01 75  LOOP4   LDA  $100,X
0362: 99 00 01 76          STA  $100,Y
0365: 88      77          DEY
0366: CA      78  MOVE1   DEX
0367: 10 F6   79          BPL  LOOP4
80      * FILL WITH * AND LEADING BLANKS
0369: C0 06   81          CPY  #$06
036B: D0 06   82          BNE  FILL
036D: A9 30   83          LDA  #$30
036F: 99 00 01 84          STA  $100,Y
0372: 88      85          DEY
0373: A9 24   86  FILL    LDA  #$24      * "$"
0375: 99 00 01 87          STA  $100,Y
0378: 24 06   88          BIT  SIGN
037A: 10 06   89          BPL  FILL2
037C: A9 2D   90          LDA  #$2D      * " "
037E: 88      91          DEY
037F: 99 00 01 92          STA  $100,Y
0382: A9 A0   93  FILL2   LDA  #$A0
0384: 88      94  LOOP5   DEY
0385: 30 05   95          BMI  OUT
0387: 99 00 01 96          STA  $100,Y
038A: 10 FB   97          BPL  LOOP5
038C: A0 01   98  OUT     LDY  #$01
038E: A9 00   99          LDA  #$00
0390: 20 3A DB 100         JSR  STROUT
0393: 60      101         RTS
0394: EB      102  ZERO    INX
0395: 10 AC   103         BPL  PAD2
    
```

— END ASSEMBLY —

TOTAL ERRORS: 0

151 BYTES GENERATED THIS ASSEMBLY



COMPUTER STATION proudly offers a high-speed binary video digitizer for the Apple II called the DITHERIZER II. The peripheral board uses a video camera with external sync to load the hi-res page of the Apple with any image the camera can capture. The DITHERIZER II is a frame grabber, DMA type digitizer requiring only 1/60th of a second to capture a binary image. Software supplied with the board enables building dithered images and capturing image intensity contours. Intensity and contrast are user controllable via the game paddles. Matrix size for dithering changeable with one keystroke. Requires video camera with external sync; recommended model, Sanyo VC1610X.
 DITHERIZER II, \$300; B/W SANYO VIDEO CAMERA, \$410
 PACKAGE OF DITHERIZER II AND CAMERA, \$650.

GRAPHICS DUMPS:

Computer Station offers the highest degree of human engineering on the market for hard copy graphics from the hi-res pages of the Apple. The following machine language dump routines are available for BASIC:
 IDS440G/445G * \$44.95
 IDS460G * 44.95
 ANADEX 9501 44.95
 NEC SPINWRITER 5510 44.95
 NEC SPINWRITER 5520 44.95
 * = Also available for use with Pascal, \$44.95

APPLEWRITER GRAPHICS:

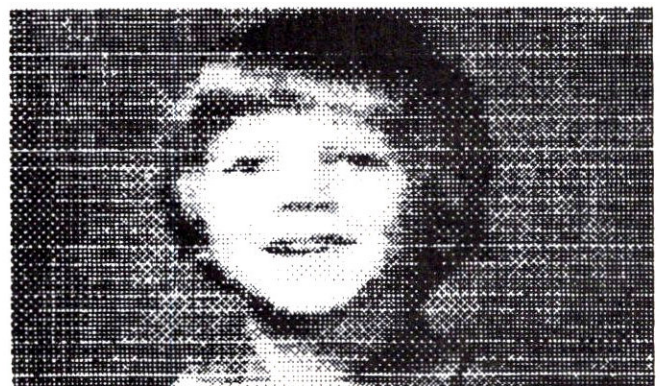
Hard copy of character sets found in DOS Tool Kit for use with Applewriter or print statements in your own programs. Requires DOS 3.3, DOS Tool Kit, one of graphic printers below:
 Silentype \$34.95
 IDS440G/445G 34.95

VISICALC

Get hard copy of the FORMULAS used in VISICALC models. Prints grid location, contents (formulas or labels), and global parameters. Handy utility for all VISICALC users.

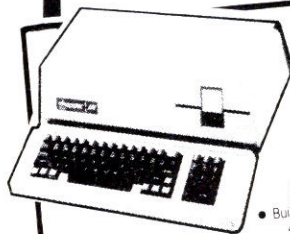
COMPUTER STATION
 12 CROSSROADS PLAZA
 GRANITE CITY, IL. 62040
 (618) 452-1860

Apple II is a registered trademark of Apple Computer, Inc.
 HIPLDT is a registered trademark of Houston Instrument, Inc.
 VISICALC is a registered trademark of Personal Software, Inc.
 DITHERIZER II is a registered trademark of Computer Stations, Inc.



APPLES APPLES APPLES

WE HAVE EVERYTHING YOU NEED FOR YOUR APPLE COMPUTER



apple III The Most Powerful Professional Computer In Its Class!

The "Information Analyst" **\$4340.00**

- 96K Apple III
- SOS
- VisiCalc III
- Apple Business BASIC
- Mail List Manager
- 12" Black and White Video Monitor

FEATURES:

- Built in RS232 Interface, Silentype Interface, 4 Channel A/D
- Calendar Clock with Battery Backup
- Memory Management for up to 128 Kbytes
- 80 Columns x 24 Lines Display
- Numeric Keypad
- 2 MHz 6502A Surrounded by LSI for instruction Set Enhancement
- Up to 560 x 190 Graphics Mode
- 16 Colors or True Grey Scale in 280 x 192 Mode
- 3 Video Outputs: B/W NTSC, Color NTSC, RGB Video

NEW

CALL FOR AVAILABILITY AND PRICE.

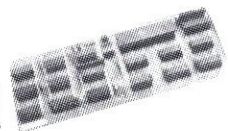
The "Word Processing System"

- 96K Apple III
- SOS
- Apple Business BASIC
- Word Painter™ Word Processor
- Audio/Visual Training Course on using Word Painter
- 12" Black and White Video Monitor
- One expansion disk drive
- Either an Silentype™ or Letter Quality Printer

\$7,800.00 W/QUME **\$5,400.00** W/SILENTYPE

★ 280 MICROSOFT Z80 SOFTCARD

- Z80 CPU on Apple Card
- CP/M 2.2 by Digital Research
- Microsoft BASIC MBASIC 5.0
- GBASIC 5.0 includes Apple Graphics
- File transfer functions for reading 13 or 16 sector Apple diskettes
- Will use 80 x 24 cards & terminals
- Can use Language Card for 56K CP/M



\$349.00 List **\$299**

Centronics 737-1

The new 9 x 11 matrix printer with proportional spacing, friction feed, pin feed and bidirectional forms control.

SAVE \$100.00

Nationally advertised at \$995.00

ACP **PRICE \$895.00**

Apple Language System

This system allows Apple users to take immediate advantage of the powerful Pascal language as well as the Integer and Applesoft BASIC interpreters. It does this by means of the Language Card, which provides 16K bytes of RAM memory that electrically replace the ROM firmware built into each Apple. Upon start-up, this RAM-memory is automatically loaded from disk then protected from change. This technique gives both Apple II and II Plus owners access to all available languages, as well as the hardware needed to run future language processors as they appear. Equally important, this product comes with a set of conversion PROMs that allow for a 20% increase in disk capacity by implementing a compatible 16-sector (143K byte) disk format.

- With your Apple Language System order, you will receive
- Apple Language Card containing
 - 16K bytes of RAM on a plug-in card
 - Auto-start monitor ROM
 - Four (4) Pascal System Diskettes
 - BASICS diskette
 - Two (2) 16-sector PROMs for the Apple Disk II Controller
 - IC puller
 - Reference manuals for Pascal, Applesoft BASIC, Integer BASIC, and the Language Card

\$469.00



DISK II FLOPPY DISK SUBSYSTEM

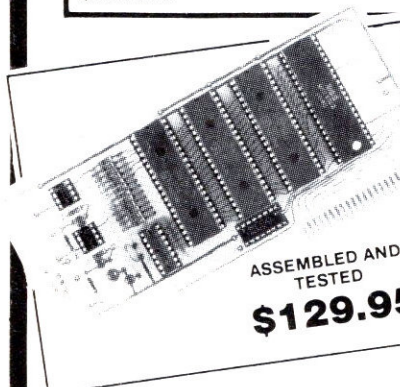
- Disk II disk w/controller \$595.00
 - Disk II additional disk 459.00
- Includes powerful disk operating system, and adds high speed random access storage to your system

APPLE II PLUS

- Apple II + w/16K \$990.00
 - Apple II + w/32K 1050.00
 - Apple II + w/48K 1100.00
- Apple II Plus systems come with floating point Basic Card in ROM. Order with integer ROM Card or Language Card for multi-language operation.

APPLE II

- 16K Upgrade Kit \$44.95
 - Apple II w/16K \$990.00
 - Apple II w/32K 1050.00
 - Apple II w/48K 1100.00
- Apple II systems come with integer BASIC in ROM. Order with Applesoft ROM Card or Language system for Multi-Language operation.



Vista™ MUSIC MACHINE 9[©] WITH 9 VOICES!

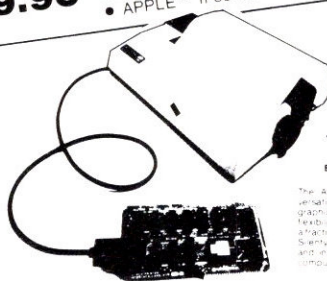
- NEW! Uses latest State of the Art LSI Technology.
- Requires only one slot for 9 voices!
- Uses three Ay3-8910's to produce nine voices. (Other competitive models have only 3 voices).
- Includes conversion software.
- Simulates three ALF Boards.
- Plays music generated by the ALF Board.
- APPLE™ II compatible.

ASSEMBLED AND TESTED **\$129.95**

AVAILABLE SOON

CRYPTEXT for Apple II data encryption for disk files.

\$449.00



Silentype Thermal Printer

Eliminating the High Cost of Hardcopy
The Apple Silentype is a quiet, portable and compact thermal graphics printer. Offers increased flexibility over other printers. An a trademark of Microsoft. Requires the Silentype receiver, D007, its power and intelligence from your Apple computer.

SAVE \$50.00
LIST \$599.00
ACP PRICE **\$549.00**

SPECIAL LIMITED TIME OFFER!

Visi-Calc

The Professional Planning Tool used by Thousands of Management Personnel across the Country.

\$119.95*

*SUBJECT TO QUANTITY ON HAND

DC HAYES MICROMODEM II

SAVE **\$50.00** LIST \$399.00
ACP PRICE **\$349.00**

SEND FOR OUR 150 PAGE 1980 CATALOG, ONLY \$2.00, FREE WITH ANY ORDER

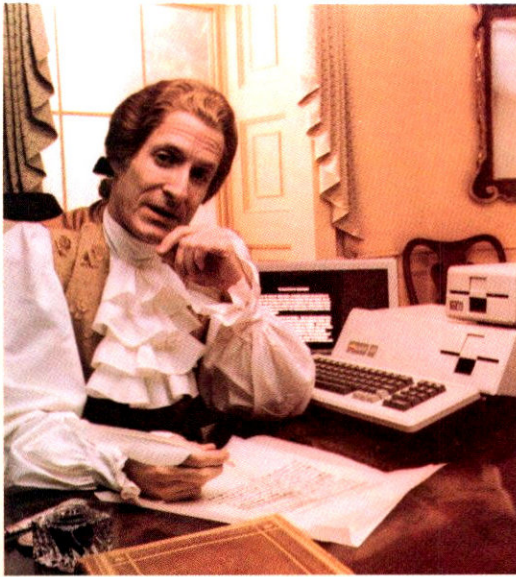


RETAIL STORES OPEN MON-SAT

STORE #1 1310 "B" E. Edinger Santa Ana, CA 92705
Showrooms, Retail, Warehouse
STORE #2 674 El Camino Real Tustin, CA 92660
Specializing in Systems

P.O. Box 17329 Irvine, Calif. 92713
Direct Order Lines: (714) 558-8813
(800) 854-8230 or (800) 854-8241

FOR INTERNATIONAL ORDERS
1310 E. Edinger (714) 953-0604
Santa Ana, CA 92705 TWX: 910-595-1565



Jefferson had one of the best minds of 1776,

but today you can make better decisions with an Apple.

Jefferson and decision makers of every century have faced the same challenge: how to explore every option before you implement the final plan. Today you have a solution. The Apple personal computer.

A tool to test contingencies.

How would higher interest rates affect your forecasts? What if vendors raised prices 2%? What if you could determine capital equipment depreciation at the touch of a key? Now, you can productively test your assumptions with ease by using the Apple system that sits on your desk.

Declare your independence with Apple.

The same Apple system that helps you forecast saves you more time by tackling word-related tasks, too. Apple helps you write reports. Print letters. With Apple's text-

editing capabilities, it might not have taken Jefferson 17 days to draft the Declaration of Independence.

Personal advice on personal computers.

Today, the power and versatility of a \$15,000 computer are at your fingertips with a *complete* Apple system solution for \$2,300 to \$7,800. It's computing made easy and economical. A few minutes with your Apple dealer and you'll understand how powerful the Apple computer can be.

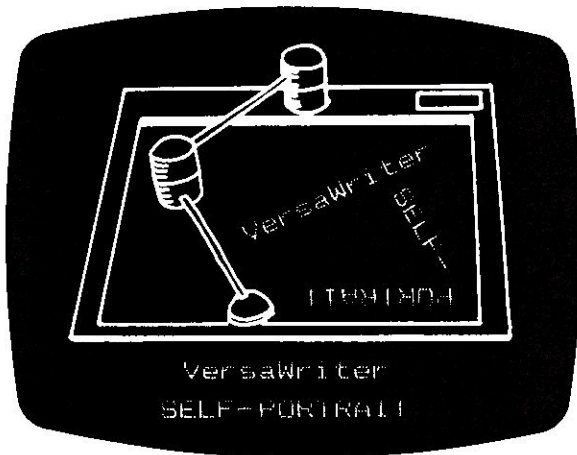
Your dealer will guide you through Apple's extensive line of hardware and software solutions, and he'll prove that all personal computers are not created equal. Don't let history pass you by. Visit your nearest Apple dealer, or call **800-538-9696**. In California, **800-662-9238**. Or write: Apple Computer, 10260 Bandley Drive, Cupertino, CA 95014.



apple® computer inc.



PRICE BREAKTHROUGH



We have used the VersaWriter to draw a picture of itself. Text may be added in any size or direction.

VersaWriter

High-Resolution Color Graphics for Apple II or Apple II Plus

The VersaWriter graphics tablet lets you create multicolor graphics and drawings with your Apple computer. It compares in quality to graphic bit pads and digitizers costing three times more money.

VersaWriter is a digitizer and software package which presents a new approach to hi-res graphics. It consists of a mylar plotting board with a clear plastic overlay. Attached to this board is the drawing arm, which has a magnifying lens with a crosshairs at its end. You simply place any graph, picture or drawing (up to 8½" x 11") under the plastic overlay and "trace" it with the drawing arm. As you trace the drawing appears on the video screen.

The superior software of the VersaWriter enables you to do much more than just trace. Immediate commands include: color choice, brush size (the width of the drawing line), fill figure with color, draw a straight line between two points, use a different scale for drawing (.25 to 4), edit, erase, smoothing factor (rounds off the rough edges as you draw), store picture on disk, and more.

One exceptional feature of the VersaWriter is the Shape Table function. You can take any picture,

or portion of a picture, and store it as a shape table. Then the table can be recalled from memory and placed on any part of the screen. You can change the size of the image, rotate it, add to it, etc. By incorporating a series of images into a single shape table, commonly used symbols can be easily inserted into a variety of different programs. VersaWriter software includes an Electronic Drawing program which is a shape table of common schematic symbols-this program will give you a good idea of what the shape table can do, as well as let you easily produce electronic or logic diagrams.

Other programs included in the software are: the Textwriter, with which text can be added to graphics (UPPER & lower case, choice of color, text size, direction of text, starting point of text). Area/Distance-this program allows you to calculate distances (or perimeters) by establishing a measuring unit (of your choice) and tracing the shape or map route with the drawing arm. Areas of figures are calculated in the same way-this includes irregular and open figures. A very simple calibration program is also on this software disk.

A second software disk contains

VersaWriter demonstration programs. For more advanced use of high-res graphics, there is a skeleton program which contains the guts of the VersaWriter. The VersaWriter is a sturdy peripheral device which plugs into the game paddles I/O port-the VersaWriter does not use up a card slot in the Apple computer. Also, the VersaWriter is not subject to the grounding problems and strong magnetic field problems of other, more expensive, hi-res graphic devices.

VersaWriter requires an Apple II with Applesoft in ROM (or an Apple II Plus), Disk, and a least 32K of memory.

VersaWriter comes complete with 8½" x 11" drawing surface, plastic overlay and two disks of software. Price \$252.00 postpaid in continental USA. VersaWriter has a 90-day warranty on parts and labor.

Credit card customers include card number and expiration date of your Visa, Mastercard or American Express card. No C.O.D.'s. Bankcard customers may order toll-free to:

800-631-8112
(In NJ call 201-540-0445)

Dealer Inquiries Invited.

Peripherals Plus

119 Maple Ave., Morristown, NJ 07960

USING USR

by
Frank Evans
PAN/TECHNOLOGY

*Reprinted from:
Stems from A.P.P.L.E.
August-September, 1980*

Apple Portland Program Library Exchange

Applesoft, like most BASIC implementations now in use, provides two methods of linkage to machine language routines. The simplest is a straight procedure call with no arguments passed or value returned, the CALL. If arguments are to be passed, pre POKeIng them to fixed RAM locations before a CALL, and PEEKing to get the results works for simple arguments, but it involves three function invocations and the attendant overhead. A simpler and more direct method is implemented in the USR(ARG) function call. USR is a function (not a procedure) in that it requires that an argument be provided and USR must be evaluated in an expression, since it returns a value.

The argument for USR is handled the same as any other function argument. A single value, a literal constant or variable may be used, and an expression which may contain other function calls is also allowed. Thus to pass an argument, no special preparation to turn it into POKeable bytes is needed; any "number" will do. The argument is evaluated by the interpreter in floating point format and is left in the Floating Point Accumulator (hereafter FPA) which is located at \$90 thru \$A3. The seven bytes left there are in floating point format for use by the machine language routine. Mixed expressions of

integer and real arguments are properly handled in the conversion as long as the value does not go out of range.

The machine language routine returns to BASIC with a RTS. In returning to BASIC, the function USR(ARG) assumes the value which is found in the FPA just before the RTS. Thus the machine language routine simply puts its results (in real form) in the FPA. The assignment of USR(ARG) to a value requires that it appears in an expression in the BASIC program which invokes the function. Thus:

```
10 X=USR(Y)
20 PRINT USR(Y),Y
30 S=SIN(USR(A+7)*60)
```

are acceptable forms but,

```
40 USR(Y) = X**2
50 INPUT USR(X)
60 FOR USR(I) = 0 TO 10
70 CALL USR(A)
```

are not acceptable calls. One form, is acceptable but USR is expected to calculate an address for subsequent procedure invocation, a function best done entirely in machine code.

The linkage to machine language must be set up by the programmer. When invoked, USR does its housekeeping as described above, then jumps to \$0A. At \$0A there are three bytes of space for the

machine language program. The intent of the APPLE programmers was to have the user place a JMP USER instruction there to get to the real code, but an RTS placed at \$0A would produce a NUL USER function. Once the processor reaches the start of the machine code (at USER) the real work can begin. The machine status and the A register need not be maintained but there is no guarantee of machine state (ie. DEC mode may be set). The code proceeds and need only terminate in an RTS to link back to BASIC.

There are two monitor routines which are very useful in handling integer arguments. If the machine code desires an integer argument, the FPA can be converted to a 16 bit integer by a call, JSR \$E10C, which "fixes" the FPA to two bytes at \$A0 (high byte) and \$A1 (low byte). If an integer result (or just a byte) is to be passed to BASIC, the two bytes may be placed in the A register (high byte) and the Y register (low byte) and, JSR \$E2F2 is executed. This "floats" the 16 bit integer to real format in the FPA. If this is the last event in the machine code (just prior to the RTS exit), the JSR may be replaced by a JMP instruction which then uses the RTS at the end of the float routine to return to BASIC.

The programmer is responsible for making space for his routines and protecting them from system RAM uses. The reader is referred to the manuals for HIMEM: and page 30 for descriptions for possible methods of protection. The USR function can be a very powerful extension of BASIC. Peripheral drivers can be implemented and special calculation routines developed for high speed processing. Multiple USR functions can be installed and one function activated at a time by POKeIng different target addresses in \$0B and \$0C. In addition to illustrating the concepts of this article, the following example shows a method of access to text files from machine language.

(program on page 76)

LIST

10 REM
THIS ROUTINE SIMULATES THE OPERATION OF PRINT C\$; WITH A USR(C) ROUTINE

20 REM
THE MACHINE CODE IS POKED FROM DATA STATEMENTS TO EFFECT THE FOLLOWING:

30 REM

```
*=$0A
JMP USER ;AT $2001
*=$2001
USER JSR $E10C ;FIX FPA
LDA $A1 ;GET VAL OF C
JSR $FDED ;COUT ROUTINE
RTS ;BACK TO BASIC
```

```
100 D$ = CHR$(13) + CHR$(4)
101 DATA 76,10,32,12,225
102 DATA 165,161,32,237,253,96
110 GOSUB 1000
150 REM
SETUP DISK OUTPUT CHARACTR STREAM
```

```
160 PRINT D$"OPEN TEXT"D$"WRITE TEXT"
199 REM
FOR LOOP PRINTS THE CHARACTER SET TO THE TEXT FILE "TEXT"
200 FOR I = 32 TO 192
210 X = USR (I)
220 NEXT I
230 PRINT D$"CLOSE"
240 STOP
```

ATTENTION: Apple Users!
WANT BETTER COLOR...MORE STABILITY?

Ever question the quality of your computer display. If so, you've probably been told "That's the best you can expect from an RF modulator...buy a color monitor".

WE CHALLENGE THAT STATEMENT!

DON'T BE SATISFIED WITH EXISTING QUALITY. See for yourself what our "new concepts" modulator can do for your picture...**MICRO-VERTER Model MVX-500, \$35 PP.** Phone orders welcomed.

HOTLINE DIAL: 402-987-3771

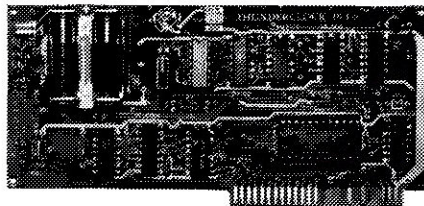
13A BROADWAY **ATV Research** DAKOTA CITY, NE. 68731

DEALERSHIPS AVAILABLE
CATALOG UPON REQUEST

THUNDERCLOCK PLUS™

A REAL-TIME CLOCK/CALENDAR

- Provides month, date, day-of-week, hour, minute, and second
- Software selectable time formats: 24 hour or AM/PM ASCII string, or as numeric values
- Provides interrupts which can be enabled or disabled under software control
- On board battery allows accurate time-keeping for up to 2 years even with your APPLE turned off



AVAILABLE NOW!

TO ORDER TOLL FREE (VISA/MC) CALL:
800-227-6204 Ext. 307 (Outside California)
800-632-2131 Ext. 307 (California Only)

AN INTERFACE FOR THE BSR X-10

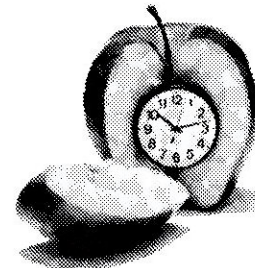
- Control AC outlets with your BSR Command Console and APPLE II
- All 22 BSR commands at your program's fingertips
- Up to 128 separate dim/bright levels
- THUNDERWARE'S Scheduler software will control AC outlets in the background, while you run another program in the foreground

BOTH FEATURES ON ONE CARD

- On-board firmware makes the THUNDERCLOCK PLUS™ exceptionally easy to use
- Read or set time, control interrupts, and send BSR commands with simple BASIC 'INPUT' and 'PRINT' statements
- Completely APPLE II compatible: INT and FP BASIC or PASCAL with Unit support
- Special introductory prices end Jan 15, 1981

- THUNDERCLOCK PLUS™.....\$119.00
Clock, BSR interface, and User's manual
- THUNDERWARE SCHEDULER.....\$24.95
Diskette with Scheduler, examples, demos, and Scheduler manual
- PASCAL SOFTWARE.....\$19.95
Diskette with PASCAL Unit for clock and BSR interface
- MANUALS each\$5.00

PUT TIME AND BSR CONTROL IN YOUR APPLE II

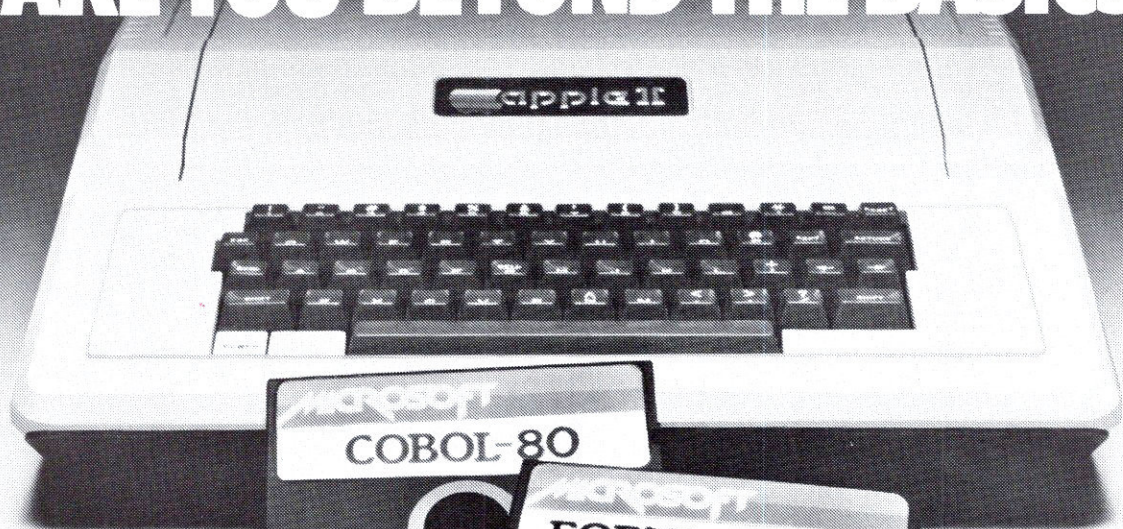


SEE YOUR DEALER

OR

WRITE FOR INFORMATION:
THUNDERWARE INCORPORATED
P.O. Box 13322, Oakland, CA, 94661

NOW THE SOFTCARD™ CAN TAKE YOU BEYOND THE BASICS.



You probably know about the SoftCard — our ingenious circuit card that converts an Apple II® into a Z-80® machine running CP/M®.

You may even know that with the SoftCard, you get Microsoft's powerful BASIC — extended to support Apple graphics and many other features.

Now, whenever you're ready to get beyond the BASICs, the SoftCard can take you into whole new realms. Starting with two advanced language packages from Microsoft.

FORTRAN AND COBOL TO GO.

Now you can run the world's most popular engineering/scientific language and the most popular business language on your Apple. Think what that means: you can choose from literally thousands of "off-the-shelf" applications programs, and have them working with little conversion. Or design your own programs, taking advantage of all the problem-solving power these specialized languages give you.

FORTRAN-80

A complete ANSI-standard FORTRAN (except COMPLEX type), with important enhancements. The extremely fast compiler performs extensive code

optimization, and, since it doesn't require a "P-code" interpreter at run time, your programs will typically execute 2-3 times faster than with Apple FORTRAN.

FORTRAN is easy to learn if you know BASIC, and the package includes a huge library of floating point, math, and I/O routines you can use in all your programs.

COBOL-80

Virtually the only choice for serious business data processing.

It's ANSI 1974 standard COBOL, with many user-oriented features added: formatted screen support for CRT terminals, simple segmenting of very large programs, powerful file handling capability, trace debugging, and much more. A separate Sort package is coming soon.

FORTRAN-80 and COBOL-80 are just two more reasons why the Apple with SoftCard is the world's most versatile personal computer. Get all the exciting details from your Microsoft dealer today. And start getting beyond the BASICs.

MICROSOFT Consumer Products, 400 108th Ave. N.E., Suite 200, Bellevue, WA 98004. (206) 454-1315.

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

MICROSOFT

CRAE 2.0

A fast co-resident Applesoft editor for Applesoft programmers. Now perform global CHANGES & FINDS to anything in your program (no restrictions on the global CHANGE & FIND).

QUOTE (copy a range of lines from one part of your program to another.

A powerful RENUMBER that is 5 times faster than other renumpers. A single line MODIFY insert/delete mode. AUTO line numbering. Formatted memory DUMP to aid in debugging. APPEND ability.

A total of 15 commands in all
Crae need be loaded only once and changes your program in memory. 48K RAM, APPLE II or PLUS, APPLESOFT ROM, and disk.

MCAT 2.0

MCAT 2.0 IS A FAST BINARY UTILITY WHICH CREATES A SORTED MASTER CATALOG WHICH IS SAVED ON DISK AS A BINARY FILE (FAST). THE MASTER CATALOG CAN BE EASILY UPDATED A WHOLE DISKETTE AT A TIME (ADD, DELETE, REPLACE). LIST/PRINT HAVE GLOBAL SEARCH CAPABILITY AND ONE OR TWO COLUMNS. PROVISIONS FOR DUPLICATE VOLUME NUMBERS. APPROXIMATELY 1200 FILE NAMES, 48K OR 32K, 13 OR 16 SECTORS DOS SUPPORTED.

CRAE on disk with 20 page manual MCAT on disk with 10 page manual CRAE and MCAT on one disk

\$24.95

\$19.95

\$39.95 with manuals

The TARTURIAN

THE TARTURIAN requires 48K RAM, APPLESOFT ROM, and disk. As you explore the 160 rooms (each done in HI-RES) gathering weapons and treasure that will prepare you for the final battle against the TARTURIAN, you will encounter deadly KROLLS, battle the MINOTAUR, try and get by COUNT SNOOTT-WEEKER, decipher the YUMMY YAKKY'S secret, make friends with the TULIE-SWEEP, avoid GHOULS, explore the PILLAR tombs, discover secret passages and more. 5 interlocking programs.

TARTURIAN on disk \$24.95



SEE YOUR LOCAL DEALER OR SEND CHECKS TO
HIGHLANDS COMPUTER SERVICES



14422 S. E. 132nd
Renton, Washington 98055
(206) 228-6691



Washington residents add 5.3% sales tax. Applesoft and Apple are registered trademarks of Apple Computers, Inc.



SPACE WAR SOFTWARE REVIEW

by David B. Garson

Program: Galaxy Space War 1
 Author: Frank Tarkany
 Distributor: Galaxy
 Purpose: Entertainment
 Language: Applesoft (ROM), 48K, Disk II
 Price: \$39.95, Disk and Manual

RATINGS

Speed: 75
 Ease of Use: 85
 Documentation: 80
 Error Comments: 85
 Screen Display: 90
 Reliability: 90
 Average: 84

Games have always been one of the strong points of the Apple, and with the Apple's graphic and sound capabilities, it is no wonder. Most of the games that have been written are of the action, or arcade style. Other than the chess games that are around, little in the way of strategy games have been available. Recently, however, strategy type games have begun to appear. One of these is Galaxy Space War I.

The Apple is actually very well suited for the task of strategy game playing. Not only does it have the graphics to display the playing board, but also the power to calculate and display the results of conflicts. In addition, the Apple can be one's opponent when no human is near and when the urge to play arises.

Space War I is set in a space type atmosphere, with your objective to destroy your opponents 'Galaxy' and thus win the game. The beauty of Space War I is that it is easy, yet challenging to play. After reading the manual, one can begin to play in about twenty minutes although it may take several games until one feels completely comfortable with all of the twenty-odd commands.

Throughout the course of the game, one is presented with a hi-res map of the galaxy. This map consists of a 33 x 17 grid, with each player's base in the middle of each end of the grid. Each player is able to build, move, and attack in a day [turn]. The key to the game lies in the shields, or screens, as they are known in the game. Depending on the setting of these screens, it will determine how much attack energy one will have, how far one can move, as well as the distance at which one can detect the enemy.



A couple of problems arise while playing the game. First off, since the game uses the same symbols, but different colors, for the playing pieces, it makes it very difficult to determine who is who with a black and white monitor or TV. Another thing that is annoying is that when playing against the computer, it may take a few minutes for the computer to complete its turn. This is not so bad when comparing this with some of the other strategy games, but does alter the flow of the game (Hasn't anyone ever heard of machine language?).

In summary, Galaxy Space War I offers a nice package. One can construct their own scenarios, either against the computer or another player. Additionally, it takes little time to learn how to play, and offers the variety to keep one's attention at the Apple's screen. If you are looking for a good strategy game that will keep you thinking, then Galaxy Space War I is for you.


PET and APPLE II Users
P A S C A L

ABACUS Software makes available its version of TINY PASCAL for the users of two of the most popular personal computers. TINY PASCAL is a subset of the standard PASCAL as defined by Jensen and Wirth. It includes the structured programming features: IF-THEN-ELSE, REPEAT-UNTIL, FOR TO/DOWN TO-DO, WHILE-DO, CASE-OF-ELSE, FUNC and PROC. Now you can learn the language that is slated to become the successor to BASIC. TINY PASCAL is a complete package that allows you to create, compile and execute programs written in the PASCAL language. You can save source and object code on diskette or cassette (PET version only). Comprehensive user's manual included. The manual can be examined for \$10 (refundable with software order).

REQUIREMENTS	
PET 16K/32K New ROMS cassette	\$40
PET 16K/32K New ROMS diskette	\$35
Apple II 32K Applesoft ROM w/DOS	\$35
Apple II 48K Applesoft RAM w/DOS	\$35
TINY PASCAL User's Manual	\$10
6502 Interpreter Listing	\$20

FREE postage in U.S. and CANADA
 All orders prepaid or COD



ABACUS SOFTWARE
 P. O. Box 7211
 Grand Rapids, Michigan 49510

GALAXY SPACE WAR I

Galaxy Space War I (WAR1) is a game of strategy in which the player has complete control of his space fleet's tactical maneuvers. Each fleet battles its way toward the opponents galaxy in an attempt to destroy it and win the war. WAR1 simulates the actual environment encountered in a space war between two galaxies. Optimum use is made of Apple's high resolution graphics (HIRES) and colors in displaying the twinkling stars universe, the colored ships of each fleet, long range sensors colored illuminations, and the alternating blinking colors used in battles between ships. Complementing HIRES are the sounds of war produced by Apple's speaker.

WAR1 is played between Apple and a player or between two players. You may play with total knowledge of each others fleet or only ships sensor knowledge of the opponents fleet. Each player builds his starting fleet and adds to it during the game. This building process consists of creating the size and shape of each ship, positioning it, and then allocating the total amount of energy for each ship.

During a player's turn he may dynamically allocate his ships total energy between his screen/detection and attack/move partitions. The percentage of the total energy allocated to each partition determines its characteristics. The screen detection partition determines how much energy is in a ship's screens and the detection sector range of its short range sensors. The attack/move determines the amount of energy the ship can attack with, its attack sector range, and the number of sectors it can move in normal or hyperspace.

When an enemy ship is detected by short range sensors, it is displayed on the universe and a text enemy report appears. The report identifies the ship, its position, amount of energy in its screens, probable attack and total energy, a calculated detection/attack/move range, and size of the ship. Also shown is the number of days since you last knew these parameters about the ship. When a ship's long range sensor probes indicate the existence of an enemy presence at a sector in space, this sector is illuminated on the universe.

An enemy ship is attacked and destroyed with attack energy. If your attack energy breaks through his screens, then his attack energy is reduced by two units of energy for every unit you attack with. A text battle report is output after each attack. The program maintains your ship's data and the latest known data about each enemy ship. You may show either data in text reports or display the last known enemy positions on the universe. You can also get battle predictions between opposing ships. The text output calculates the amount of energy required to destroy each ship for different energy allocations.

**APPLE® II, 48K, APPLESOFT
 ROM CARD, DISK II DOS 3.2**

**WAR1 DISK & MANUAL ...\$39.95
 (CA residents add 6% sales tax)**

Write or call for more information



**GALAXY
 DEPT. A03**

P.O. BOX 22072

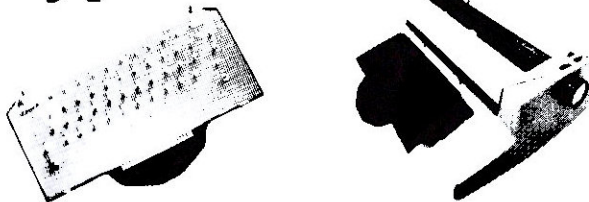
SAN DIEGO, CA 92122

(714) 452-1072

IAC MEMBER CLUB ADDITIONS (from page 49)

- | | | |
|---|--|---|
| <p>COMPUSERS
P.O. Box 2064
Hastings, NB 68901</p> | <p>APPLE PI OF THE PERMIAN BASIN
415 - E. 43rd St.
Odessa, TX 79762
Phone - 915-333-3430</p> | <p>CLUB DE MICRO-ORDINATEUR ST-JEAN
P.O. Box 21
St-Jean, Que., Canada J3B 6Z1</p> |
| <p>PRINCETON APPLE USERS GROUP
c/o Computer Encounter - 2 Nassau St.
Princeton, NJ 08540</p> | <p>TIDEWATER APPLE ORGANIZATION
1021 Tivoli Crescent — Apt. 102
Virginia Beach, VA 23456</p> | <p>BOLO/UB APPLE CLUB
1208 Patenaude #3
Laval — Quebec, Canada H76 3H2
Phone - 514-663-2771</p> |
| <p>TOLEDO APPLE USERS
1417 Bernath Parkway
Toledo, OH 43615
Phone - 419-476-8463</p> | <p>NORTHERN VA APPLE USERS GROUP
PO Box 10411
Alexandria, VA 22310
Phone - 703-971-2070</p> | <p>GRUPO USARIOS APPLE DE COLUMBIA
A.A. 91226
Bogota, Columbia
Phone - 249-71-85</p> |
| <p>MIDWEST CITY HOSPITAL APPLE USERS
2825 Parklawn Dr.
Midwest City, OK 73110
Phone - 405-732-6682</p> | <p>WISCONSIN APPLE USERS
PO Box 11463
Milwaukee, WI 53211
Phone - 414-964-6645</p> | <p>APPLE ORCHARD
Taman Maluri; Batu 3;
Jalan Cheras, Kuala Lumpur, Malaysia</p> |
| <p>AUGUSTA APPLE USERS GROUP
819 Jackson Ave. N.
Augusta, SC 29841
Phone - 803-279-4974</p> | <p>APPLE USERS CLUB WESTERN AUSTRALIA
269 Marmion St.
Cottlesloe — W.A., Australia 6011
Phone - 09-457-1555</p> | <p>APPLE GEBRUIKERS GROEP NEDERLAND
Bergselaan 145A
Rotterdam, Netherlands</p> |
| <p>RAPID CITY APPLE USERS
3016 Glenwood
Rapid City, SD 57701
Phone - 605-343-2949</p> | <p>MICOM
P.O. Box 60
Canterbury—Vict., Australia 3126
Phone - 03-509-9729</p> | <p>APPLE P.I.
3rd Flr/Liberty Bldg/Pasay Road
Makati/Manila, Phillipines 3116
Phone - 88-70-36</p> |
| <p>FRANKLIN MOUNTAIN APPLE ORCHARD
Drawer G
El Paso, TX 79951
Phone - 915-877-2383</p> | <p>OTTOWA 6502 USERS GROUP
POB 6283 Station J
Ottawa-Ontario, Canada K2A 173
Phone - 613-728-6728</p> | <p>TAC2 APPLE USERS GROUP
P.O. Box 87421
Houghton, South Africa 2041</p> |

**At last...the
Typewriter Interface!**



Turn your electric typewriter into a low cost, high quality hard copy printer. 1 Year Warranty

Dynatyper—the patented* RDI—I/O Pak is fast becoming the industry standard for typewriter output. Why? Because:

1. It takes 2 minutes to initially install and 5 seconds to remove or replace.
2. You *do not* have to modify your typewriter. All factory warranties and maintenance agreements on your typewriter will be honored.
3. You can use it with *all* powered carriage return typewriters that have U.S. keyboard. Our Model I works with all *non* Selectrics and our Model II works with Selectrics. Conversion between models takes 2 minutes and the kit (26 plungers) is available for a nominal charge.
4. You don't have to lug around a bulky printer when you travel. If there is a typewriter at your destination, you can install the light (3 lbs.) I/O Pak in just 2 minutes.
5. Same interface for TRS-80, Apple and GPIB. Centronics and Pet compatible interfaces are available in third quarter 1980. Electric pencil available.
6. Delivery: Stock to two weeks. Price: \$499. for the complete system, FOB Rochester, Domestic.

Over 1000 in operation today. VISA and MasterCard accepted. Call Ken Yanicky at 716-442-7804.

*Patent Pending

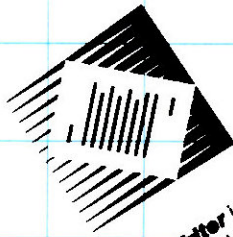
ROCHESTER DATA

3000 Winton Road, South, Rochester, NY 14623 incorporated

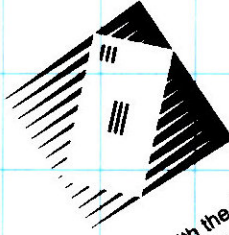
ADVERTISER INDEX

PAGE

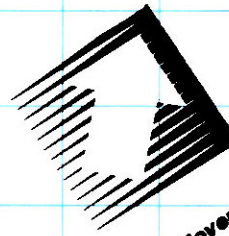
ATV Research	76
Abacus Software	79
Advanced Business Technology, Inc.	19
Advanced Computer Products	72
Andromeda Incorporated	42
Apple Computer, Inc.	47-73
Automated Simulations	7
Bit 3 Computer Corporation	34
Business and Professional Software	67
Cavri Systems, Inc.	19
Computer Station	71
Connecticut Information Systems Co.	64
Dakin5 Corporation	36
Eastern House Software	28
Escon Products, Inc.	19
Galaxy	79
Highlands Computer Services	78
Information Unlimited Software, Inc.	IBC
Instant Software, Inc.	40-41
M & R Enterprises	IFC
Microsoft Consumer Products	77
Mountain Computer, Inc.	1
Nibble Magazine	10
On-Line Systems	21-23-25-27
Peripherals Plus (Creative Computing, Inc.)	74
Programma International, Inc.	8 BC
Rainbow Computing, Inc.	6
Rochester Data	80
SSM Microcomputer Products	5
Sirius Software	54
Southeastern Software	2
Southwestern Data Systems	70
Stoneware Microcomputer Products	64-69
Strategic Simulations, Inc.	30
Synergistic Software	20
Thunderware Incorporated	76
Videx	24



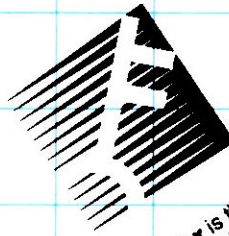
EasyWriter is a powerful word processor designed for the people who want the best.



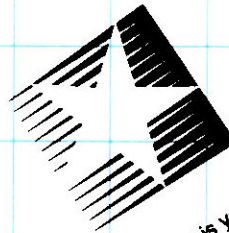
Now, with the **EasyMailer** system, your Apple can be used to greatly reduce time, paperwork, and money spent on form letters, mailing labels, and other documents.



EasyMover is the first Electronic Mail System which combines the versatility of a word processor with the ability to move or transmit text files to another computer.



Datadex is the key to interactive data management for your Apple computer and the heart of an automated office's operation.
(Available first quarter 1981.)



TellStar is your computer window to the celestial objects as they appear at your home, or any location on the Earth you desire.



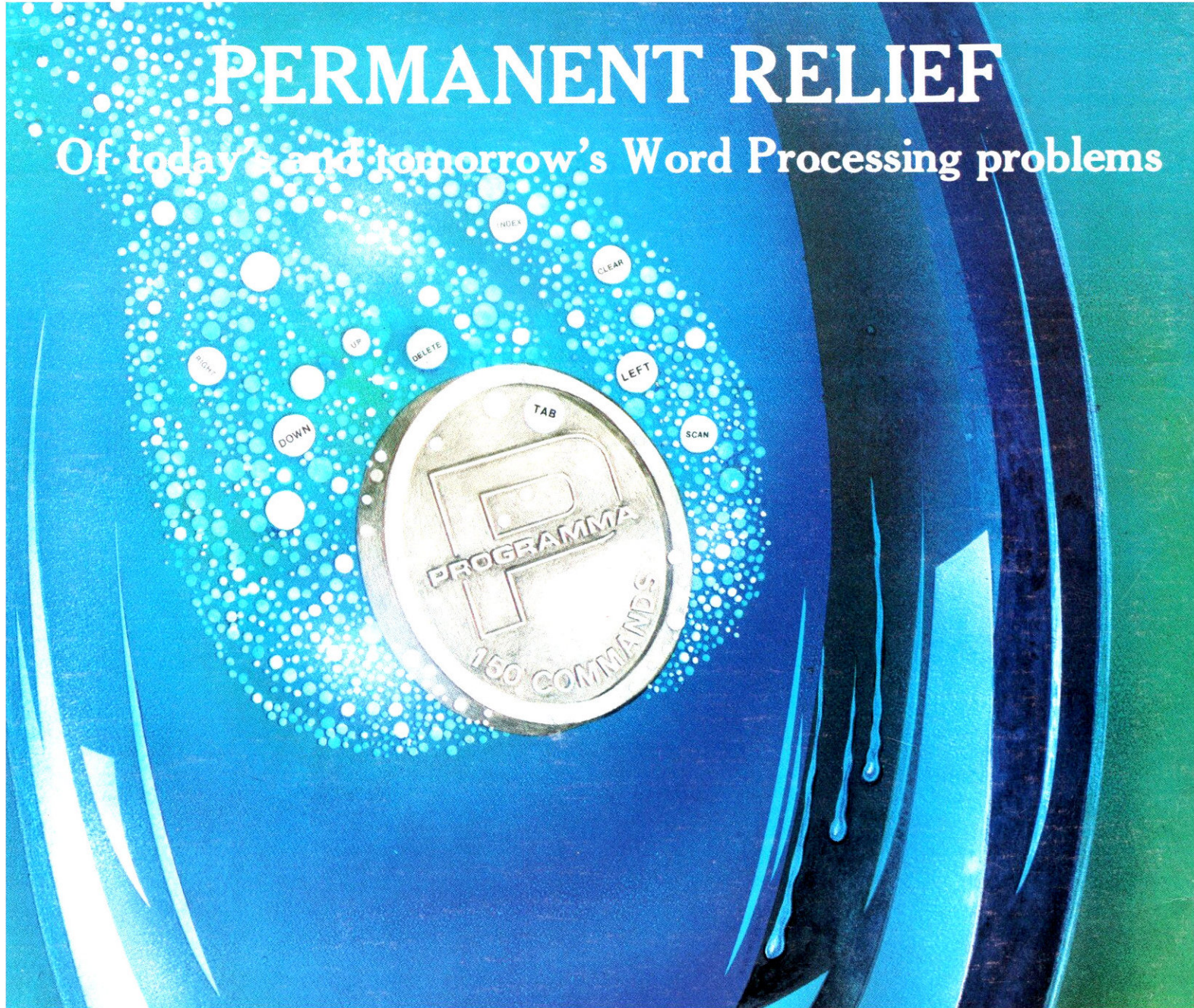
IUS is word processing, data management, data communications, and educational software for the Apple computer. Write or call your local dealer to find out more about the automated office made easy!

IUS (Information Unlimited Software, Inc.)
281 Arlington Avenue, Berkeley, CA 94707
415-525-9452

Apple is a (tm) of Apple Computers, Inc.
EasyWriter copyright 1980, Cap'n Software
Datadex copyright 1980, Sonoma Softworks
TellStar copyright 1980, Scharf Software Services

PERMANENT RELIEF

Of today's and tomorrow's Word Processing problems



Apple PIE

Apple PIE (Programma International Editor) and FORMAT (text formatter) offer full strength solutions to today's word processing problems. These versatile, powerful programs provide document preparation and word processing capabilities previously found only on much larger computer systems.

PIE is a general purpose, full screen editor that uses control keys and function buttons to provide a full range of editing capabilities such as search and replace, delete, copy, insert, move. Changes may be made directly anywhere on the screen and are shown as they are performed.

FORMAT uses simple instructions embedded in the input text to describe the desired appearance of the final document. It handles centering, underlining, indenting, page numbering,

+

Formatter

margins, headers, footers, even form letters, and includes a proofing capability.

These high-quality, cost-effective programs come with comprehensive documentation and run on a 32K Apple II. They are available through your local computer store or direct from Programma International, Inc. at the introductory price of \$79.95*.

VIDEX VERSION T.M.

DOUBLE VISION T.M.

SUPR TERM VERSION T.M.

STANDARD VERSION

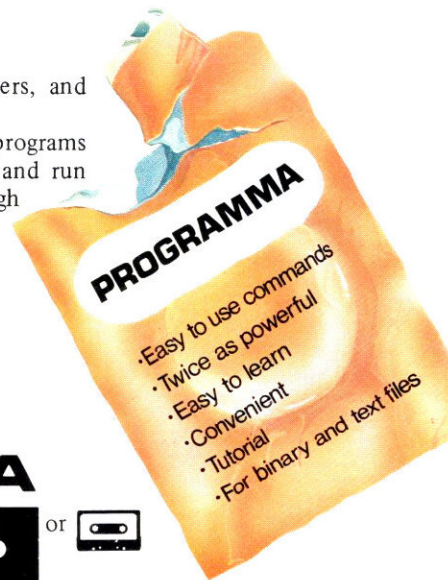
*December 1, \$129.95.

PROGRAMMA

2908 N. Naomi Street
Burbank, California 91504



OR



Simple enough for the beginner. Versatile enough for the professional.