July 1983 USA $2.95

Spreadsheeting:
VisiTips Make It Easy

inCider™

Green's Apple* Magazine

Your Apple Can Pick a Winner

At the track—Harness Race Handicapper p. 68

At the tables—Viva Las Vegas p. 105

Discover the Power of Pascal

AFTER YOUR FIRST DRIVE GO WITH AMDISK.

The AMDISK-I Micro-floppy disk system is an engineering breakthrough in disk size, storage capacity, media protection and user convenience. It's fully compatible with your Apple II*® and Apple IIe and is furnished with an interconnect cable. Enjoy a full 286K storage† capability and the extra convenience of the new 3" hard plastic encased diskette. Packed 10 per box, the disk cartridge fits into a shirt pocket and is easy to mail.

Suggested Retail Price .... $299 (3" cartridge $6.99 ea.)

The Amdisk Single Drive system is ruggedly constructed for years of trouble-free operation and is backed with our 90 day warranty on parts and labor.

Just circle the reader service number for complete specifications.

*Apple II is a registered trademark of Apple Computer, Inc.
† Requires recording on both sides

2201 Lively Blvd. • Elk Grove Village, IL 60007
(312) 364-1180 TLX: 25-4786

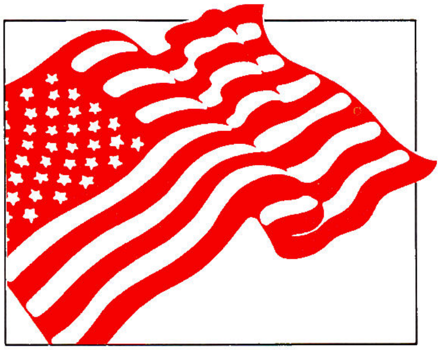
AMDEK CORP.

Amdek . . . your guide to innovative computing!

Circle 95 on Reader Service card.

inCider
July 1983
Vol. 1 No. 7

Your Apple can pick a winner, p. 68 and p. 105.



All-American Program, p.100.

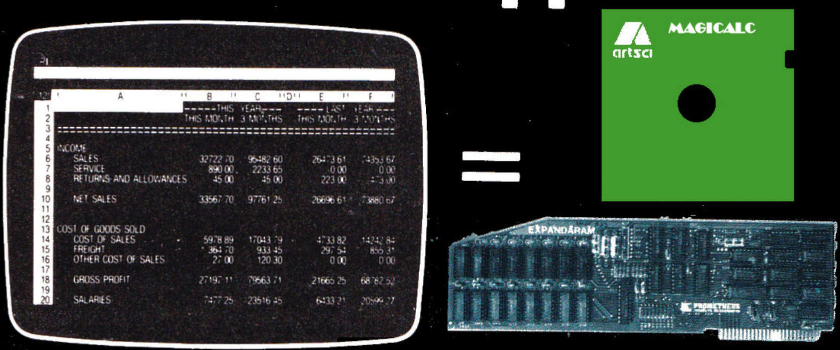
DEPARTMENTS

ARTICLES

Cover art by Erick Ingraham.

inCider (ISSN pending) is published 12 times a year by Wayne Green Inc., 80 Pine St., Peterborough, NH 03458. Phone: 603-924-9471. Second class postage pending at Peterborough, NH, and additional mailing offices. Subscription rates in U.S. are $25 for one year and $53 for three years. In Canada and Mexico, $27.97—one year only, U.S. funds drawn on a U.S. bank. Canadian distributor: Micro Distributing, 409 Queen St. West, Toronto, Ontario, Canada M5V 2A5. BC Canadian distributor: Graymar Data Services, Ltd., #4 258 E. 1st Ave., Vancouver, BC V5T 1A6. Foreign subscriptions (surface mail), $44.97—one year only, U.S. funds drawn on a U.S. bank. Foreign subscriptions (air mail), please inquire. In South Africa contact *inCider*, P.O. Box 782815, Sandton, South Africa 2146. All U.S. and Canadian subscription correspondence should be addressed to *inCider*, Subscription Department, P.O. Box 911, Farmingdale, NY 11737. Please include your address label with any correspondence. Postmaster: Send address changes to *inCider*, Subscription Services, P.O. Box 911, Farmingdale, NY 11737.

the 2nd Generation Spreadsheet for Apple II™ and //e™



MAGICALC™ + EXPAND-A-RAM™

Everything VisiCalc can do and much more. Plus 64K or 128K RAM. Plus 80-column display for Apple //e at no extra cost.

MAGICALC is a completely new second generation spreadsheet program for Apple II. This state-of-the-art system includes 70-column upper and lower case video, full 80-column board display, hard disk compatibility, individual column widths, invisible columns for confidential data, and full compatibility with VisiCalc that lets you utilize existing VisiCalc models. Refer to the box below for a comparison of MAGICALC and VISICALC.

EXPAND-A-RAM 64K or 128K of additional RAM memory for expanded MAGICALC worksheets. Includes MEMORY MANAGEMENT SYSTEM – relocates DOS for more free memory in BASIC. You also get RAMDISK emulators for APPLE DOS, APPLE PASCAL,™ and CP/M™ – a software package that emulates a standard Apple floppy disk drive but is 2 to 10 times faster. Plus more.

Registered trade marks: Apple... belongs to Apple Computer, Inc., Magicalc to Artsci, Inc., Expand-A-Ram to Prometheus Products, Inc., VisiCalc to Visicorp, CP/M to Digital Research, Inc.

EXPAND-A-RAM for Apple II plugs into any slot and works with the many software programs that use 16K of RAM in slot 0 – including CP/M, APPLE PASCAL,™ LISA, MERLIN... and many others. EXPAND-A-RAM/80 for Apple IIe, which includes the 80-column display, goes into the auxiliary slot. No modification of your Apple is required.

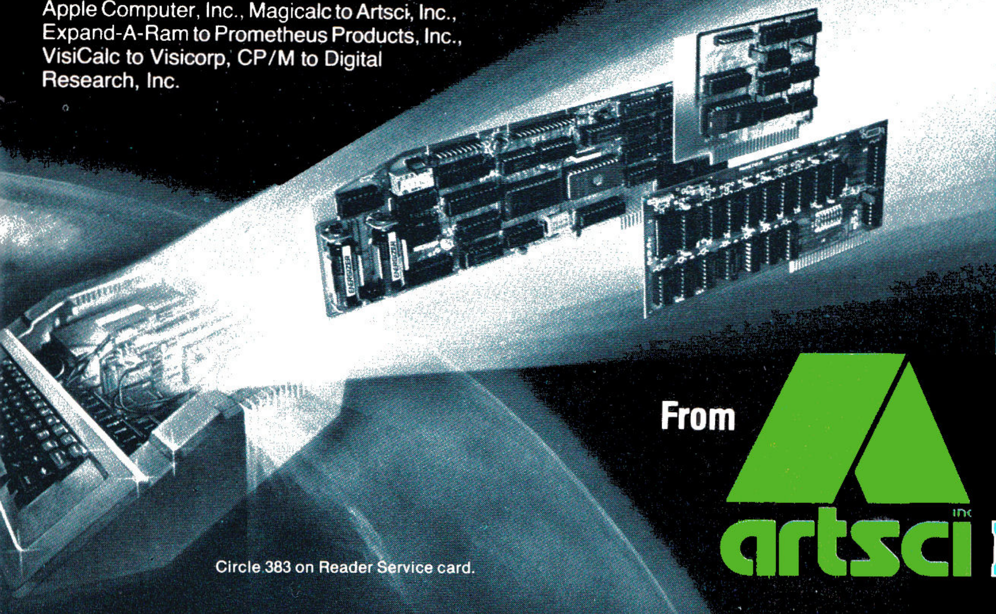
SPECIAL LOW INTRODUCTORY PRICES for the combination MAGICALC plus EXPAND-A-RAM are $399 with 64K RAM and $499 with 128K RAM. Apple IIe users (only), please specify your choice of the standard EXPAND-A-RAM or EXPAND-A-RAM/80 with 80-column Display at no additional charge.

See your local dealer for details:

ARTSCI, INC.
5547 Satsuma Ave., North Hollywood, CA 91601, (213) 985-2922

PROMETHEUS PRODUCTS, INC.
45277 Fremont Blvd., Fremont, CA 94538, (415) 490-2370

COMPARE!	MAGI-CALC	VISI-CALC
Preboot required	NO	YES
70 column upper and lower case video display	YES	NO
Full 80 column board display	YES	NO
Individual column widths	YES	NO
Invisible columns for confidential data	YES	NO
Hard disk compatibility	YES	NO
Full compatibility with VisiCalc	YES	YES
Program plus 128K RAM	$499	$745



From artsci inc ... and PROMETHEUS

Circle 383 on Reader Service card.

Remarks from the Publisher... Wayne Green

When the Apple folk announced the Lisa, despite the calm, assured outer mien of the firm, I'm sure there were all of the normal fears that accompany a new product. And this certainly had to be amplified by virtue of the newness of the concept of the product. . . the battles within the firm over it. . . and the intense scrutiny given this high-flying firm.

After having read a wide variety of reviews of the Lisa, I think the industry judgment is that the system may be easy to use but is just too expensive for the relatively small group at which it is aimed. Most of the reviews were mixed in that respect, perhaps opening the way for the promised Macintosh, which is supposed to do much that Lisa can do.

There is still silence on what may be coming for the low end market. That's a tough place to play ball these days, with the VIC-20 down to around $89 in some stores and the Timex around $50. Whew! Panasonic is about to jump in, joining T.I., Atari, Mattel and others in the blood bath. This is sure a good time to buy cheap computers.

And what about the Model 100 Radio Shack unit? Just about everyone at Wayne Green Inc. rushed out to pick the New Hampshire Radio Shack stores clean of these the day they arrived. It looks like a fair percentage of our people have a Model 100 under their arm when they come to work now. To digress, as is my wont, the 100 is almost exactly the computer that I described two years ago as the computer needed for the portable market. Unfortunately, we're not in the hardware business or else we might have followed through. But if we could think of it here, how about the Apple people? I wonder if they've got one coming like that too? It was a most logical product development.

* * *

When the president of DEC announced the entry of his firm into the desktop computer market last year, he gave a statistic that half of the people who have bought microcomputers have just put them away in the closet and forgotten them. I enjoy statistics like that and am not unknown to come up with them on demand myself.

"Remember that the better Apple is able to do, the more support you are going to have for your particular Apple."

Well, with all due lack of respect, there isn't any way to know what the magnitude of the disaster is, or if it even is a disaster. But there is no question that this is, at least, a miserable problem. . . perhaps edging dangerously on a disaster. Let's look at it.

Now you and I know that no Apple owner is going to put his Apple in the closet and forget it. But this obviously could be the fate of something as inexpensive as the Timex, and. . . worse. . . this situation could come to haunt all of us. Remember that the better Apple is able to do, the more support you are going to have for your particular Apple. You'll have more information, more accessories, more programs, more services and so on. You (and I) have a vested interest in Apple's success.

Okay, if we poison the minds of a growing number of the type of people who rush out to try new things, we could bring the small computer revolution to a grinding halt much sooner than we like to expect. We've been getting used to outlandish growth in the industry. . . firms sprouting up one day, going public a few months later, and getting into the hundreds of millions in sales. All this depends on a fast growing market. . . more people buying computers for the home, for the office, the store and for school.

So, if we start souring people on computers, we're just likely to screw up the whole blooming works. Okay, assuming that you agree with me, the obvious next question is: So what can *I* do about all this? Hell's bells, most of us are just a lonely voice in the wilderness. Aha. . . not as lonely as you might think. No, you have an ally in this misery. . . me. But I can't do much by myself. . . all I can do is put on the pressure, but I have to know from *you* where the pressure should be applied.

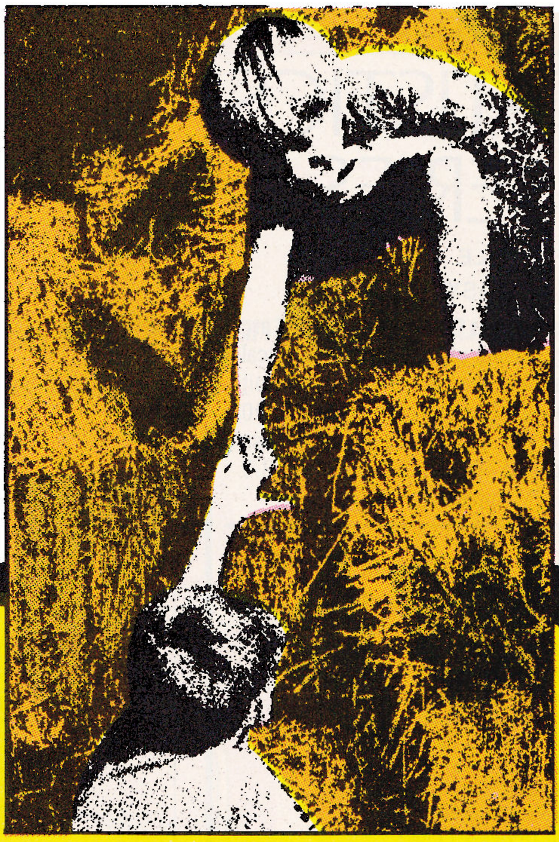
Let's all accept some responsibility

THANK YOU...

...FOR HELPING US TO THE TOP!

Because of the loyal patronage of smart shoppers, Micro Mountain has become one of the industry leaders in the sales of **Apple * hardware, software and accessories.** We began our business on the idea that most computer owner/operators are sophisticated buyers and that they appreciate such features as:

- **EFFICIENT SERVICE:** The prompt delivery, courteous treatment, and attention to detail that comes form a company which has a huge inventory and years of experience.
- **COMPETETIVE PRICES:** The kind you get when you deal with substantial companies who buy in large quantity and pass the savings on to their customers.
- **DEPENDABILITY:** So you know that the people who gave you the GEAT DEAL "Yesterday" are around "Tomorrow;; to be accountable for the goods and services they sell.



micro mountain

MICROSOFT
Multi Plan $199.50

CONTINENTAL SOFTWARE
CPA Accounting Pkgs.
General Ledger, Payroll, Accounts Receivable, Accounts Payable (Ea. Module) **$179.50**

SOFTWARE TECHNOLOGY
The Creator (for Apple) **$109.50**
ALSO AVAILABLE FOR IBM at $199.50

PLATO SOFTWARE
Physics $58.50
(COMPLETE PLATO LINE AVAILABLE-CALL FOR PRICES

EDU-WARE
Algebra V-VI (One disk) **$39.50**

16K Ram Card . . $49.50

SILICON VALLEY SYSTEMS
Word Handler . . $119.50

SOFTWARE MASTERS
The Visible Computer: 6502 . . $39.50

KENSINGTON MICROWARE
System Saver . . . $65.50

CENTRAL POINT SOFTWARE
COPY II PLUS . . $28.50
COPY II PC AVAILABLE FOR IBM

BRODERBUND
Bank Street Writer $58.50

EASTSIDE SOFTWARE
Wild Card $119.50
Includes Utility Disk for Apple II+ and Apple II e

DOW JONES
Market Analyzer . $275.50

MICRO SCI
Disk Drive $269.50
CALL FOR QUANTITY DISCOUNTS

-ASK FOR FREE CATALOG-

EXTRA SAVINGS BONUS!
Any order of 3 or more programs entitles you to your choice of Original Adventure Game or Applesoft® Tutorial on disk FREE!

PHONE IN YOUR ORDER —TOLL FREE— (800) 854-5649
Washington State residents see phone numbers in Order Blank at Right.

ORDER PHONE HOURS:
Mon.-Fri. 9 to 6 (PST)
Sat. & Sun. 10 to 2 (PST)

BE SURE TO ADD $2.50 SHIPPING AND HANDLING FOR ALL SOFTWARE ORDERS, ADD ADDITIONAL $3.00 FOR BLUE LABEL (Air). SHIPPING ON HARDWARE ITEMS—EXTRA. Washington residents add 6½% sales tax. We accept MASTERCARD, VISA and AMERICAN EXPRESS. C.O.D.'s add $5.00.

NAME ______

STREET ______

CITY ______ STATE & ZIP ______

CARD # ______ EXP. DATE ______

SIGNATURE ______

14617 N.E. 169th ST., WOODINVILLE, WA 98072
ORDER PHONE — Outside Wash. — (800) 854-5649
Wash. Residents & Cust. Service (206) 483-2000

PRICES SUBJECT TO CHANGE WITHOUT NOTICE

Circle 55 on Reader Service card.

for keeping our strange industry moving along. If you see someone...perhaps Timex...putting out a computer with lousy instructions, don't just sigh and forget it; write me a letter and bitch about it...offer constructive suggestions. If people struggle with the Timex...and I'm using that as an example, even if the shoe doesn't fit...if they struggle and lose, then all of us lose. For you know as well as I that every one of those people who figure that they got burned...got taken for a sucker...are going to bad mouth not Timex, but the whole computer field.

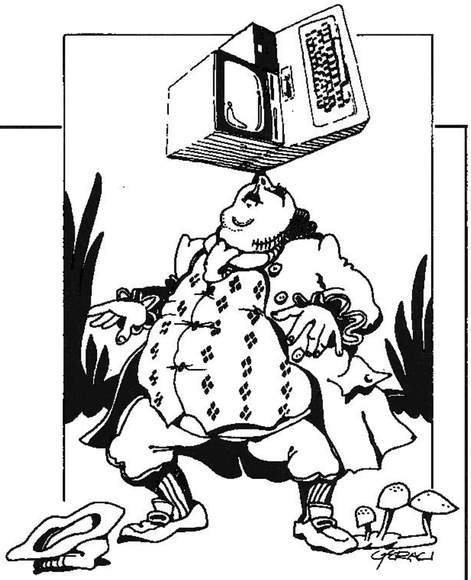
So find out what computers your friends are getting...and how they

"Heat brings light, generally...if you have enough of it."

are doing with them. If they are having any serious problems, get the facts on the situation and send me a letter I can publish. As obscure as some of my magazines may be, they *are* read by the industry. Messages for Timex...and any other firm...will reach the right people. Oh, they may get furious about them, but I'm ready for the heat. Heat brings light, generally...if you have enough of it.

What are we looking for? We're looking for people who have bought a computer and had a hard time of it...for reasons of hardware, service, documentation, instructions or stupidity. The more we can force the desktop computer industry to adjust to the customer, instead of demanding the reverse, the faster the industry is going to move ahead. So you see what is going on and pass the word ...okay? And the end result will be a better industry, more Apple sales and more stuff for your Apple. ■

by Bill O'Brien



Counterpoint: In Defense of the III

April's *inCider* was, as usual, stuffed full of insight and enlightenment about Apple products. Within its pages appeared the usual complement of articles, reviews and commentary, including the editorial by Wayne Green. He's the publisher of *inCider* as well as a bevy of other magazines about computers and what is probably *the* ham radio magazine, *73*. (It was, in fact, the computer articles in *73* that got me interested in these beasts.)

Over the last five years I have alternately been impressed by Mr. Green's insight and angered by what appears to be his never-ending battle to stir the masses by issuing controversial verbiage.

The role of devil's advocate is, of course, not foreign to me. But this time my erstwhile benefactor has invaded a realm near and dear to my heart. One paragraph of his April editorial rubbed me so much the wrong way that I had to find someone to scratch his nails on a blackboard to calm me down.

"It's a good thing the Apple III bombed," quotes the sage.

Mr. Green, sir, in many things I respect your wisdom; after all, you are more deeply involved in the computer industry than I am. But please, there are some perspectives that cannot be gained even by talking to all the retailers in the world.

I was once a computer retailer. I have been interviewed by the *Times* (both *New York* and *London*) concerning matters computer, as well as a few other periodicals. I have been included in an opinion poll conducted by the prestigious Rosen newsletter concerning IBM computers. I was even chosen as spokesperson for the Commodore computer line at a symposium of retailers held in Canada in 1981. With all that, and I will grant you it still costs me 75 cents to use the New York subways, I would never have made as naive a statement about a computer as you made about the Apple III.

> **"Do you blame Chrysler when your Reliant gets a bad tank of gas?"**

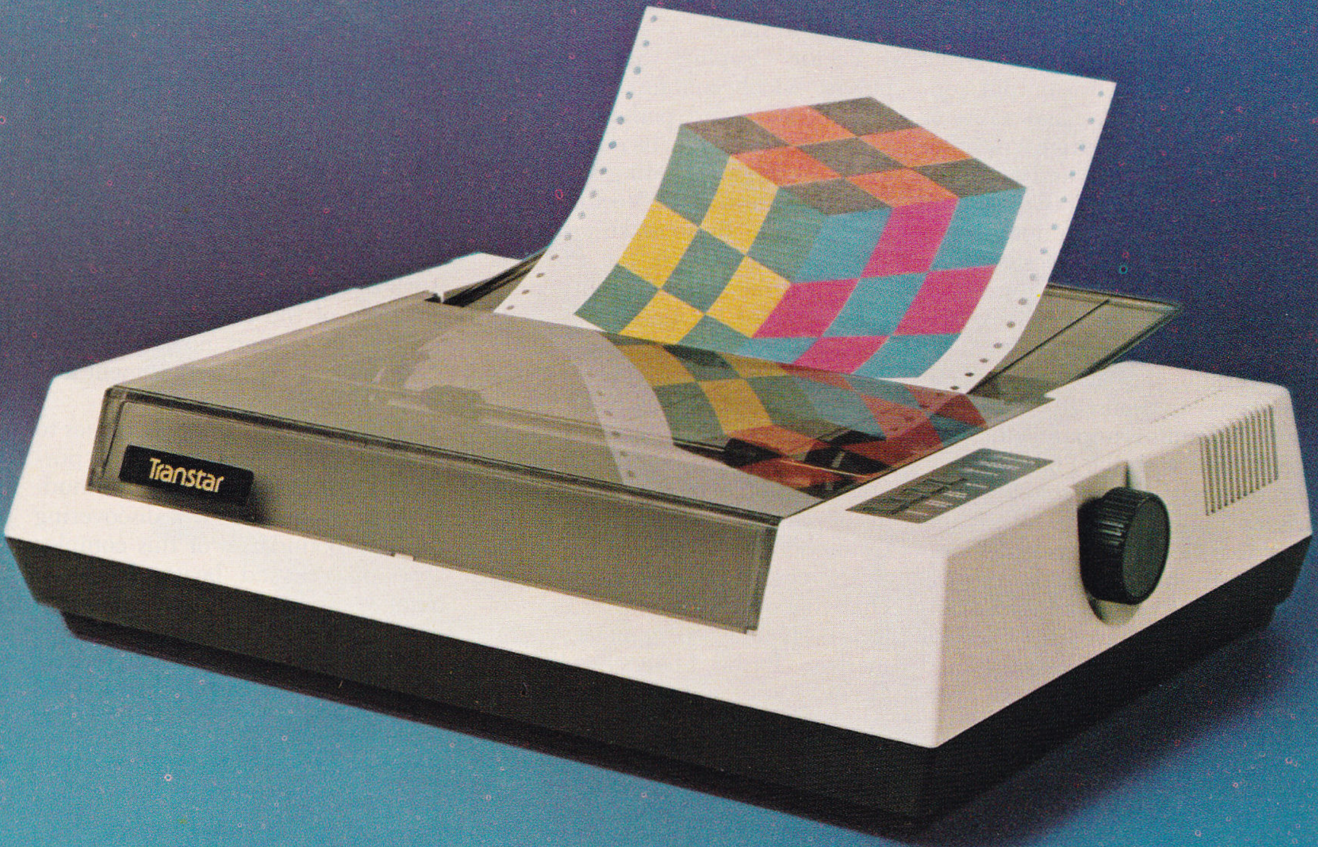
Without dredging up the entire history of the III, a painful tale I am sure most of us know, let me say that the III is quite alive, despite many attempts by retail computer outlets to kill it. I will grant you that sales have in no way approached those of its kin, the Apple II, but there is no reason they should. It's a completely different type of computer. Let's look at the facts.

The Apple III is standard with 128K of memory, and has an 80-column screen with upper and lower case. It has a numeric keypad and an integral disk drive. Why the drive is so small in capacity is a question sometimes raised. The simple answer—to maintain compatibility with Apple II software, something that a host of computer manufacturers have been trying to do but have yet to succeed at. Do we blame the III because software manufacturers have incorporated physical devices or Integer/Applesoft token-passing protection schemes into some of their programs to bolster their paranoia? Do you blame Chrysler when your Reliant gets a tank of bad gas?

The III has excellent video-handling features, including screen partitioning; physical position formatting as well as the typical variable formatting; an inverse mode; four graphics levels; multiple and redefinable character sets (which, thanks to the efforts of Interactive Structures and their PKASO board, are printable on many dot matrix printers, not just the MX-80); non-destructive graphics-to-text or text-to-graphics switching with four distinct graphics display elements; and the inclusion of text into the graphics environment which, at maximum black and white levels, produces a character set identical to the screen character set and, at worst, is simply an over-emphasized low-resolution display, equal to the Apple II. The audio capabilities are equally impressive, but you can read about those in the January issue of *inCider*.

Basic on the III is unrivaled, with its ability to employ more than 64K of memory and to reroute output to any physical device without redefining the program steps that control the output flow. This allows the printing of reports to screen, printer or disk (for inclusion in word processor originated documents), and, in a program requiring all of these options, can cut programming steps by as much as two thirds.

Color for the Price of Black and White



Transtar 315

Color Printer for only $599 suggested retail price

- New technology four-hammer print head
- Unique diagonal ribbon provides simpler and more reliable operation
- Prints 7 colors plus more than 30 shades
- All colors and shades are printed in a single pass of the print head
- Built by Seikosha, the oldest manufacturing company of the famous Seiko group—providing precision products since 1892.

Transtar · Box C-96975 Bellevue, Washington 98009

Circle 265 on Reader Service card.

Included at no extra charge on the III is a software programmable serial port; an RGB color output port; a standard video monitor output connector for either black and white or color; a built-in disk drive controller for up to four drives; and an audio output connector and two joystick ports, one of which can double as a printer port for the inexpensive Apple Silentype printer.

I feel like Homer cataloging the ships that set forth against Troy.

Concerning the III's initial problems, Apple's response in exchanging all affected units is unparalleled. Did Ford do that with the Pinto?

I must also point out that the comparisons implied in the sales campaign for the IBM Personal Computer were based strictly on the Apple II. The only tenuous advantage the PC may have over the III is the use of an *almost* 16-bit 8088, and that was implemented with an operating system rumored to be below average in speed.

Vector Graphics' Series 4 computers, which contain both a Z-80 and an 8088 and perhaps state-of-the-art equipment in its range, do not as yet have an operating system for the 8088. Nor will the Vector Graphics version of Basic make use of the 128K supplied with the machine. The Osborne has reliability problems, or at least so I assume from the rows of keyboards I've seen lined up in some repair facilities here in New York. Altos, although excellent in design and implementation in both 8-bit and true 16-bit versions, lacks anything that can be remotely called a support arm for its equipment. Commodore has proven to be the fickle company of computerdom, changing operating systems more often than most companies change models. Radio Shack—well, Mr. Green, you've said enough about Radio Shack for both of us. And I won't even bother to comment on the clones.

So then, why, if it's such a great machine, isn't the Apple III selling like 30-cent-a-gallon gasoline? The answer—lack of support.

One part of your editorial I totally agree with. Apple will shortly be forced to re-evaluate its dealer network in light of the complexity and direction of the machines it is now marketing. Dealers no longer have the leisure to sell the equipment and forget it, like they did the II.

Until such time as Apple comes to terms with the people who have purchased the III, we owners will muddle through on our own, discovering the true boundaries of this complex and sophisticated beast. It hasn't bombed, and it isn't dead. As was the understated Jacques Brell, the Apple III is alive and well. And, I believe we owners intend to keep it that way. ■

Word processing.

It's simple.

The typewriter was a wonder in its time. **Letter Perfect**™ is a wonder for today. Letter Perfect from LJK lets you use your computer to write letters and create reports and edit your work. Change a word. Move a line. Combine two paragraphs. You can do it all with Letter Perfect, and more. Much more.

When it comes to practical software for Ataris and Apples, Letter Perfect is simple to learn. And hard to beat. Ask your dealer for a demonstration, or write LJK for more information.

LJK, INC. • 7852 Big Bend Blvd., St. Louis, MO 63119 • (314) 962-1855

Circle 106 on Reader Service card.

APPLE JUICE

If your Apple or Franklin isn't giving you all the power you need, it's not rotten. It just needs to be juiced up. With a Starfire™ Winchester hard disk drive from Corona.

It leaves Apple DOS in the dust.

For starters, the Starfire gives you all the power inherent in a hard disk drive. Lightning speed (up to 10 times faster than a floppy), enormous storage capacity (5 and 10 megabytes), and superb data reliability.

With none of the bottlenecks, headaches and hassles of floppy drives.

But even better, it makes Apple DOS infinitely friendlier, faster and more powerful.

A lot more slices from the pie.

Our version of Apple DOS is so powerful and friendly, it's like getting a whole new computer. You can work with files up to 400K. Boot into any program from the hard disk. And run programs with a single keystroke.

We've added other features from the Corona PC, our own 16-bit desktop and personal computers. You can automatically, selectively backup your volumes. Partition the disk and run up to 16 different operating systems, including Apple DOS, Pascal and CP/M.[1]

And depend on our DataGuard™ and Fail-Safe™ data protection systems for the disk reliability and data integrity you need.

Your saving grace.

And if you're running programs that use a lot of K's, you're still okay because we don't use any of your main memory.

The Corona Starfire is software-compatible with industry-standard packages like VisiCalc,[2] DBMaster,[3] dBASE II[4] and many others. So if your Apple is being squeezed, juice it up with the Corona Starfire hard disk.

For the name of your nearest Corona dealer, just call us toll-free at 800-621-6746. Or write Corona Data Systems, 31324 Via Colinas, Westlake Village, CA 91361. (213) 991-1144.

Circle 175 on Reader Service card.

corona

FEEL THE POWER.

Letters

Graphics Dump Enhancement

Thank-you for the article "From Screen to Printer" in the March edition with the Epson graphics dump program. That article alone has justified the price of subscription. I found that some of the hi-res screens did not look very good and suspected that I needed to invert the image. A Basic program will do just that.

```
10 FORJ = 8192 TO 16383
20 POKE J,255 - PEEK(J)
30 NEXT J
```

Unfortunately, it seems to take forever. Not being knowledgeable about machine code I was reluctant to take the plunge. However, the speed of the Basic program forced me. It was a matter of pointing to each location in memory and subtracting from $FF to invert the byte. The code is entered by the monitor at 03C0 and called from Basic by CALL 960.

Glen Roger
Box 785
McBride, B.C.
Canada

Readers' Response To "It Works"

It worked! Did you see how that program of mine in "It Works" in the April *inCider* brought out the computer pros? Now, I could use an old teacher's ploy and excuse my mistakes by saying, "I wrote that program on purpose, just to see if you were paying attention." (But I didn't.) My program was a stab of desperation when the programming got rough.

As a result of publishing my article, I received two nice letters from other hackers who were willing to share their know-how in programming skills. Both solutions were similar, using IF X/12 = INT(X/12) THEN INPUT "RETURN...". With that one, neat line, I am now able to replace ten lines in my own program. You can bet that I'll never forget that solution! Thanks to Robert Tucker of New Jersey and Philip Sturmfels of Missouri.

Mr. Sturmfels also expressed an idea for a "Problem of the Month" column for *inCider*. Just such a column would be a real aid. Right now in my programming, I'm trying to break out of a loop and there must be an easier way. How about it, readers?

Bob Blaske
8867 S.E. Colony
Hobe Sound, FL 33455

Editor's note: We have forwarded an overwhelming number of letters to Bob from our readers giving a simpler solution to his problem.

Cassette Viewpoints

You claim in the Hot Cider editorial (April '83) that "Apple... cut the cassette umbilical cord." And you ask what would it have cost to keep the cassette interface in the system. But both the Apple II Plus and the IIe *still* have the cassette interfaces. The only thing Apple dropped was the free cassette cable—still available for $3.

However did you estimate a $5 to $10 saving on cassette software delivery over diskette? Software companies get their diskettes for less than $2 each and packaging does not add much. A difference of $1.50 is more likely the case.

If you have ever tried to help customers use cassettes to load and save programs as I used to do, then you would understand that the dealer would gladly absorb that difference in cost tenfold rather than deal with cassette recorders. The tapes are not with us because the users found that they were not pleasant to deal with. When I first got involved with Apples, tapes were common—in fact I still have a tape player hooked up to my third Apple (a IIe, which succeeded my II and II Plus). But I soon realized that I would not make good use of tapes. Later, in 1980 when I started to sell Apples, tapes were on the decline. Our users group continued with a tape library into 1981 but the last users converted to disk drives at that time. No one had regrets. Diskettes are wonderfully reliable, fast, and easy to work with. They are not very expensive, even though the disk drives are expensive compared to recorders.

Michael H. Carlson
1201 Lakeview Ave.
Minneapolis, MN 55416

Your comments about preserving cassette tape simply reflect a failed technology. The Apple Corps of Dallas has a thousand members and we no longer support tape distribution, because the person who was handling it wasn't distributing any and bought disk drives. The Apple has never had a very good cassette management system and data storage is just plain difficult.

In my use of cassettes on the KIM-1, Apple and TRS-80 Color Computer, I have found cassette tapes to be less reliable than diskettes; or rather, the recorders are less predictable. In the time I have used diskettes, carrying my machine and the diskettes all round the city and the country, using 200–300 diskettes as a free-lance custom programmer and heavy business user of my own machine, less than five diskettes have failed, one from heavy constant use (anticipated with a backup), all the others failing initialization within the first few minutes of use and unique to a box.

Mike Firth
104 N. St. Mary
Dallas, TX 75214

After reading your editorial Hot Cider in the April '83 issue I just had to sit down and write to you. You stated Apple didn't do users any favors when they cut the cassette umbilical cord. I was most upset. I worked hard to earn the money to buy the Apple II. Within 16 months I could not buy a ready-made cassette program for my Apple. I am not proficient enough to write anything complicated so I am left with about 10 professional tapes and what I can do on my own.

It hurts when you pay $1,400 for an Apple and there is no way to increase its use. In these times I have no way to get additional funds to pur-

PKASO™

Printer Interface Family

Welcomes the New Apple //e... We work together.

PKASO Interfaces come complete with Cable, Instructional Diskette and Comprehensive Manual.

The PKASO family makes you and your Apple Computer a master of text and graphics.

PKASO makes it easy to use the features of your printer—select character sizes, vary line spacing, even print in colors. Simple PKASO commands make these features usable from the keyboard or a program.

PKASO also adds features to your system. Press a few keys and get a snapshot "dump" of the image you see on the screen—text or graphics. Add new characters and symbols that you couldn't print before, using our SuperFont™ system. Add our new PipeLine™ printing buffer and your printer can take its time while you and your Apple move on to the next task. The PipeLine is a modular add-on to the standard PKASO board.

The PKASO interface is designed for Apple II and Apple /// in all the popular configurations. It prints in full color on the IDS Prism Printer, and in striking black on C. Itoh, Centronics, Epson, IDS, NEC, and Okidata matrix printers.

NEW!

The IS Pipeline™ Printing Buffer with Random Access Printing stores paragraphs or pictures for printing in any order—any number of times!

- Universal—works with any parallel (Centronics style) computer/printer combinations.
- 8K to 128K Bytes of memory with data compression for efficient use of memory space.

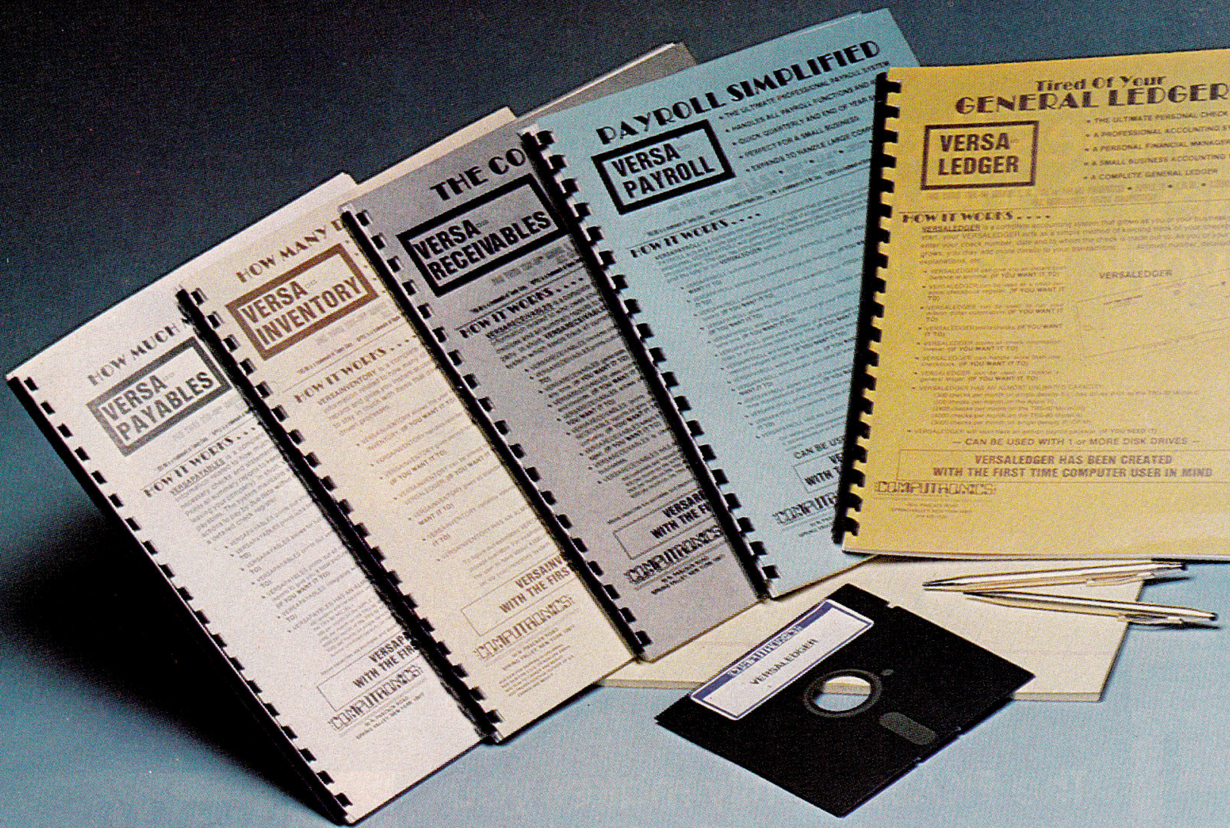
Circle 28 on Reader Service card.

is

Interactive Structures Inc.
146 Montgomery Avenue
Bala Cynwyd, PA 19004
Telephone: (215) 667-1713

Introducing the Most Powerful Business Software Ever!

TRS-80™ (Model I, II, III, or 16) • APPLE™ • IBM™ • OSBORNE™ • CP/M™ • XEROX™



The VERSABUSINESS™ Series

Each **VERSABUSINESS** module can be purchased and used independently,
or can be linked in any combination to form a complete, coordinated business system.

VERSARECEIVABLES™ $99.95

VERSARECEIVABLES™ is a complete menu-driven accounts receivable, invoicing, and monthly statement-generating system. It keeps track of all information related to who owes you or your company money, and can provide automatic billing for past due accounts. VERSARECEIVABLES™ prints all necessary statements, invoices, and summary reports and can be linked with VERSALEDGER II™ and VERSAINVENTORY™.

VERSAPAYABLES™ $99.95

VERSAPAYABLES™ is designed to keep track of current and aged payables, keeping you in touch with all information regarding how much money your company owes, and to whom. VERSAPAYABLES™ maintains a complete record on each vendor, prints checks, check registers, vouchers, transaction reports, aged payables reports, vendor reports, and more. With VERSAPAYABLES™, you can even let your computer automatically select which vouchers are to be paid.

VERSAPAYROLL™ $99.95

VERSAPAYROLL™ is a powerful and sophisticated, but easy to use payroll system that keeps track of all government-required payroll information. Complete employee records are maintained, and all necessary payroll calculations are performed automatically, with totals displayed on screen for operator approval. A payroll can be run totally, automatically, or the operator can intervene to prevent a check from being printed, or to alter information on it. If desired, totals may be posted to the VERSALEDGER II™ system.

VERSAINVENTORY™ $99.95

VERSAINVENTORY™ is a complete inventory control system that gives you instant access to data on any item. VERSAINVENTORY™ keeps track of all information related to what items are in stock, out of stock, on backorder, etc., stores sales and pricing data, alerts you when an item falls below a preset reorder point, and allows you to enter and print invoices directly or to link with the VERSARECEIVABLES™ system. VERSAINVENTORY™ prints all needed inventory listings, reports of items below reorder point, inventory value reports, period and year-to-date sales reports, price lists, inventory checklists, etc.

VERSALEDGER II™ $149.95

VERSALEDGER II™ is a complete accounting system that grows as your business grows. VERSALEDGER II™ can be used as a simple personal checkbook register, expanded to a small business bookkeeping system or developed into a large corporate general ledger system **without any additional software.**

- VERSALEDGER II™ gives you almost unlimited storage capacity (300 to 10,000 entries per month, depending on the system),
- stores all check and general ledger information forever,
- prints tractor-feed checks,
- handles multiple checkbooks and general ledgers,
- prints 17 customized accounting reports including check registers, balance sheets, income statements, transaction reports, account listings, etc.

VERSALEDGER II™ comes with a professionally-written 160 page manual designed for first-time users. The VERSALEDGER II™ manual will help you become quickly familiar with VERSALEDGER II™, using complete sample data files supplied on diskette and more than 50 pages of sample printouts.

SATISFACTION GUARANTEED!

Every **VERSABUSINESS™** module is guaranteed to outperform all other competitive systems, and at a fraction of their cost. If you are not satisfied with any **VERSABUSINESS™** module, you may return it within 30 days for a refund. Manuals for any **VERSABUSINESS™** module may be purchased for $25 each, credited toward a later purchase of that module.

To Order:

Write or call Toll-free (800) 431-2818
(N.Y.S. residents call 914-425-1535)

- * add $3 for shipping in UPS areas
- * add $4 for C.O.D. or non-UPS areas
- * add $5 to CANADA or MEXICO
- * add proper postage elsewhere

VISA AMERICAN EXPRESS

DEALER INQUIRIES WELCOME

All prices and specifications subject to change / Delivery subject to availability.

COMPUTRONICS INC.

50 N. PASCACK ROAD, SPRING VALLEY, N.Y. 10977

* TRS-80 is a trademark of the Radio Shack Division of Tandy Corp. - *APPLE is a trademark of Apple Corp. - *IBM is a trademark of IBM Corp. - *OSBORNE is a trademark of Osborne Corp. *CP/M is a trademark of Digital Research - *XEROX is a trademark of Xerox Corp.

Circle 66 on Reader Service card.

chase a disk drive.

I think the Apple computer is the best, but I am sorry I invested my hard-earned money in one. I would not recommend it to anyone of limited means today. Apple made a great mistake; they cut off a lucrative market when they hurt me and others like me. True, the tapes were slow, but money was also slow coming in. I've never had any problems with my cassettes.

I hope cassette users will write Apple before other companies will fill this need for low-cost cassette use.

Mrs. L. A. Duff
411 Garfield Ave.
Palmyra, NJ 08065

Not Accountable?

There is more to the problem of supporting local Apple dealers than the difficulties dealt Apple owners who have no local dealer in the area. The deeper aspect of this situation is that the local dealer system makes no one accountable for the products Apple sells.

This system might work if Apple supported its products. But when they reach the point of selling a dot matrix printer for several hundred dollars, and do not bother to provide any user's manual for it, how can the local dealer help you?

Naturally, I am not very sympathetic towards a system that leaves the purchaser of Apple products out in the cold, with nowhere to turn. To me it just seems like a way to avoid anyone being accountable. In fact, there ought to be a law against it.

Lola Lahtinen
PO Box 10066
Glendale, CA 91209

Book Review Questioned

As the author of a review of Marvin L. DeJong's book *Apple II Assembly Language* that has been accepted but not yet published by *Microcomputing*, I feel compelled to respond to the review of the same book in your March issue. While the reviewer does make some valid points, she also levels a number of charges that cause me to wonder whether she has a copy of the same book I have.

Your reviewer is entitled to her opinions, but her facts are usually wrong. The book is much better than the reader of your review would be led to believe. It has its weaknesses, but it easily beats most of the competition.

F. Kuechmann
1921 S. 13th Ave.
Maywood, IL 60153

Real Estate Software Needed

We believe the use of computers in the real estate industry is a sleeping giant about to awake.

We plan to publish listings of all available real estate software programs we are made aware of, showing what the programs will do, the language they will operate on, the type of hardware they will run on, their price, and the name and address of the source.

We would appreciate receiving information from all sources outlining the information they would like to have included.

J. Don Shields
Shields Realtors
Drawer 758
Shelby, NC 28150

Portuguese Version

I have already purchased two Apple computers—a Plus and a IIe. But now I am deeply interested in the IIe International Version.

I am a Brazilian citizen who has been using Apple computers a lot, basically with the excellent Apple-Writer program. But when it comes to writing in my native language, Portuguese, there the problems start.

Not that Portuguese has that many characters English does not, but it has a few. And there are a few characters found in English and not in Portuguese.

I very much need a Portuguese version for the Apple IIe. And I believe that in Brazil, a country of 130 million people—not to mention Portugal and its African ex-colonies—I'm not the only one.

I would be immensely grateful if you could help me—and others like me.

Flavio de Campos
5160 Claremont Ave. #405
Oakland, CA 94618

Editor's note: Apple advises us that a Portuguese keyboard for the IIe will be available by the time this issue of inCider *appears.*

Up with Games

Hooray for George Engel! His Fermentations, "In Defense of Gaming," hits the alien squarely with the phaser. Those of us who enjoy a good game are usually looked upon as individuals who suffered through an unhappy childhood. Society has conditioned us to believe that games are almost universally for children. The only acceptable games for adults are cards, chess, and checkers.

"Society has conditioned us to believe that games are...for children."

I, for one, will readily admit to being an avid gamer, both of board games and computer games. I am proud of my game collection and can honestly say that I probably have at least one game in every category for which a game is available. Gaming is my hobby.

One problem with board games is the necessity of a human opponent for the game to be fully enjoyed. This problem vanishes with computer games. A good computer game combines skill with a little luck, in order for the player to be victorious. The game forces the player to use skill, but also allows for the unexpected. By putting out little fires skillfully, the player achieves the ultimate goal and gets the chance to put out the big fire.

Therefore, can't we say that a

game mirrors life itself? Don't we have to put out a number of little fires before we get a crack at the big one? Don't we have to plan and reason before we can successfully attack the problem?

As human beings, we revel in a challenge. We enjoy tackling a tough problem, and relish our successful solution to it. Gamers are similar to early explorers. The early explorer probed to find the limits of what he was searching. Gamers probe to find the limits of their minds. Those limits haven't even been sighted yet.

Martin J. Hrovat
Rt. 1 Box 355-J Lot #109
Slidell, LA 70458

Microline 92 Information?

I am in desperate need of information indicating how to access the special functions of the Okidata Microline 92 from the Apple II Plus using the AppleWriter II word processing program.

Donald L. Amoroso
125 Prescott Court
Athens, GA 30605

Bibliography Program Used With Word Processing

As a reader, I must congratulate you on the excellent quality of the first few issues of *inCider*. As an author, I should point out that my own articles published here have prompted more reader response than I had previously received anywhere, so you must be doing something quite right, indeed!

One point I would like to pass on to the readers. The REFCITED Pascal program of mine that was in the March 1983 issue can be used with manuscripts prepared with any word processing system that puts out files in the DOS 3.3 format—and that appears to include most of the popular word processors. To use it, the DOS textfile must be converted to a Pascal textfile; the VISITERM diskette includes a Basic program that does this, and the Nov/Dec 1981 issue of *Call-A.P.P.L.E.* included a Pascal program that does the same thing.

This should make my Pascal bibliography system more attractive to people who are already heavily committed to DOS-based word processing systems.

James R. Florini, Ph. D.
Biology Dept.
Syracuse University
Syracuse, NY 13210

by Dan Bishop

Basic Apple Graphics

This month I'll present some of the fundamental concepts involved with doing both lo-res and hi-res graphics, with the emphasis on lo-res graphics. Applications of these concepts can go beyond simple video games. The old adage that a picture is worth a thousand words can be applied to business programs and educational programs to make these more effective in transmitting information. Graphic displays of histograms or pie charts depicting a company's profit progress or expense distributions provide a more lasting impression than mere tables of numbers.

Educational programs that add graphic animation to illustrate concepts hold the student's attention longer than displays containing only written text, providing a greater opportunity for the concepts to seep in and be remembered. The real impact of the personal computer in our society will be felt only as more and more applications become available that make extensive and appropriate use of computer graphics.

The emphasis in the previous statement should be on the word appropriate, however. Graphics can impair the usefulness of a program as easily as enhance it. Graphics should not be used simply because they are available. The basic objectives of a given program should be kept uppermost in the programmer's mind at all times, and if graphics can enhance the achievement of those objectives, the added time and effort needed to incorporate the graphics display into the program will be worthwhile.

Apple Graphics and Memory

The Apple supports two different types of graphics displays, lo-res (low resolution) and hi-res. In order to use the hi-res capability, the computer must have at least 16K of RAM memory. This is because the RAM reserved for hi-res displays in the Apple begins at address 8192 and extends up to 16383.

In order to explain just how RAM memory correlates with the video display, you need to consider the available memory in your computer in terms of a memory map, in which each byte of memory has a unique location that is specified by a memory address. The first byte of RAM has an address of 0, the second byte is addressed as 1, etc. You might imagine a street with the houses built along only one side (with a park on the other side of the street) and with the houses numbered sequentially from zero.

Now imagine the street broken up by cross streets, so that there are 256 houses in each block. The houses along the street are still numbered in sequence, so the first house in the second block has the number 256 (remember that we started numbering at 0). One byte of information is allowed to reside at each address. This illustration correlates with how RAM memory is organized. Each block is referred to as a *page* of memory, starting at page 0.

The video display screen is not a part of RAM memory in the Apple. However, each display position on the screen displays a byte of information that corresponds to a byte of information that is stored in RAM, sort of like a mirror. When the computer is displaying a screen full of text, the video monitor is actually reflecting pages 4, 5, 6 and 7 of RAM memory. (This corresponds to actual RAM addresses of 1024 through 2047.) The RAM memory that is reserved for display storage is referred to as a "buffer." Thus pages 4–7 are the memory buffer for the text display.

Your Basic program instructions are stored in memory, starting at location 2048 (page 8), right after this memory buffer. So when your program uses an instruction such as

```
10 HOME: PRINT"HELLO"
```

this instruction is placed in RAM somewhere just above location 2048 in page 8. When the program is run, pages 4–7 are cleared out and the ASCII codes for the word HELLO are placed in memory locations 1024–1028. In addition, those same codes are sent to the video driver that directs the five characters to be reproduced on the screen display.

Lo-res graphics use the same RAM buffer area that the Apple uses for text displays. As far as the Basic user is concerned, only two single word instructions, GR and TEXT, are needed to switch the computer's interpretation of how to send information stored in the buffer to the screen. The GR instruction tells the computer to send that data to the screen as graphics characters, while TEXT tells the computer to send the data as text characters. Of course, if the RAM is filled with graphics codes and the word TEXT is received, the display will fill with garbage as the text character generator produces characters to correspond to the graphics codes. The HOME command must then be used to clear the screen of the garbage.

There are actually two screen buffers for lo-res graphics display. The manual refers to these as "pages," but so as not to confuse this use of the word with a "page" of RAM (containing 256 bytes), I will refer to them as screen buffers. The secondary lo-res screen buffer coincides

Address correspondence to Dan Bishop, PO Box 429, Buena Vista, CO 81211.

"Lo-res and hi-res graphics differ in the number of display positions that are available on the screen."

with the same four pages of RAM that are used for program storage, pages 8–11. Because of this, rather unusual programming techniques are necessary to use this secondary screen buffer. Otherwise, your program would get clobbered. Consequently, the secondary screen buffer for lo-res graphics is not often used.

By the same token, there are two screen buffers for hi-res graphics. The first of these occupies pages 32–63 of RAM memory (addresses 8192–16383), and the second uses pages 64–95 of RAM (addresses 16384–24575). Of course, in order to use the secondary hi-res screen buffer, you must have over 24K of RAM. The instruction that tells the computer to switch from text mode or lo-res mode into hi-res mode is simply HGR to access the primary hi-res screen buffer, and HGR2 to access the secondary hi-res screen buffer.

Since the RAM used for these two buffers is simply an extension of the RAM used for storing your Basic program, a long Basic program that extends above page 32 will restrict your use to the secondary screen buffer. You *can* use both buffers with long programs but, as with the secondary lo-res screen buffer, you'd have to take extraordinary measures to clear the buffer areas while still maintaining the integrity of your program. Figure 1 shows how these areas of RAM relate to each other.

Apple Graphics Screen Displays

At this point you may be asking how lo-res and hi-res graphics differ. The main difference is in the number of display positions that are available on the screen. This in turn affects the size of each point that can be plotted on the screen, and the overall appearance of the display.

Lo-res graphics provide a full screen display that consists of 40 columns by 48 rows. Thus each point that is plotted in lo-res will have the same width as a text character and half the height. In other words, it will appear as a small square (half the

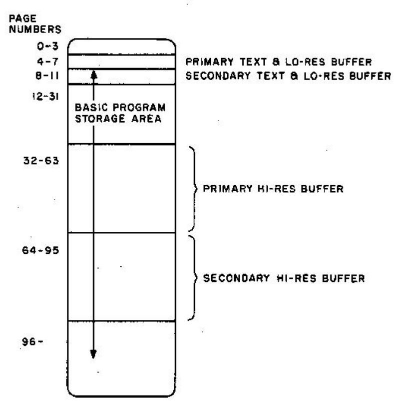


Figure 1. Memory Map of Apple II RAM, showing the locations of the four screen buffer areas used by the Apple for text, lo-res, and hi-res displays.

height of the cursor block). In RAM, each byte contains the color code for two of these blocks. The two blocks are stacked vertically, so memory address 1024 contains the codes for the blocks located in column 0, rows 0 and 1.

Hi-res graphics provide displays that consist of 280 columns and 192 rows. To get that many dots (53,760) on the screen, you can expect the individual dots to be pretty small. This is super for any diagonal lines (they won't have a stair-step appearance) and curves that look like curves. Of course, much more RAM is required to buffer a hi-res screen display (32 pages vs. 4 for lo-res).

The lo-res graphics allows selection of up to 16 different colors (0–15), using the COLOR=## command. In order to support that many colors in hi-res graphics, considerably more memory would be required. Thus a compromise of sorts is reached, and the number of hi-res colors is restricted to 8 (0–7), using the command HCOLOR=##. Figure 2 lists the colors available for both lo-res and hi-res graphics along with the corresponding number appropriate to that color. The number code for hi-res colors is misleading, however, in that the dot's position on the screen has as much to do with determining that dot's color as the number selected for HCOLOR. But more about this next month.

With the Apple it is no simple matter to mix text with graphics on the same area of the screen. In hi-res, this can be done by plotting the letters in or by defining the individual letters as shapes and drawing the shapes on the screen in the desired location. In lo-res, only large block letters may be drawn on the screen. However, it is possible to reduce the size of the screen in either hi-res or lo-res mode so that the bottom four text rows (corresponding to 8 lo-res graphics rows or to 32 hi-res graphics rows) are left open for text. In fact this is the normal result when either GR or HGR commands are used to access these modes. On the other hand, HGR2 results in full screen graphics rather than mixed text and graphics.

To switch between full screen graphics mode and mixed text and graphics, in which the bottom four lines are opened as a "text window," the following instructions are used:

POKE −16301,0 changes graphics display from full screen to mixed text plus graphics.
POKE −16302,0 changes graphics display from mixed text plus graphics to full screen graphics.

In either case, to be able to easily use the four rows of text (using PRINT statements or whatever), you must be using the primary screen buffer area, whether in lo-res or in hi-res graphics. This is because the text buffer area that is displayed on the bottom four rows corresponds to the graphics buffer area then in use, whether it is the primary or secondary area. And recall that the secondary buffer area for lo-res (and thus also for text) coincides with the RAM area used to store your program.

What this means, then, is that if

Lo-Res	Lo-Res	Hi-Res
0—black	8—brown	0—black
1—magenta	9—orange	1—black or green
2—dark blue	10—grey	2—violet or black
3—purple	11—pink	3—violet, green or white
4—dark green	12—green	4—black
5—grey	13—yellow	5—black or red
6—medium blue	14—aqua	6—blue or black
7—light blue	15—white	7—blue, red or black

Note: In Hi-Res mode, the first color listed applies to dots plotted in even columns, the second to dots plotted in odd columns. Two dots side by side horizontally will appear white if the color is 3 or 7.

Figure 2. Color codes that are used in lo-res and hi-res graphics modes to specify the color of the pixel or block to be plotted next.

HLIN aa,bb AT cc	aa = starting column for horizontal line bb = ending column for horizontal line cc = row (0–47) in which the horizontal line will be drawn
VLIN aa,bb AT cc	aa = starting row for vertical line bb = ending row for vertical line cc = column (0–39) in which the vertical line will be drawn

Table 1. Graphic short cuts.

```
POKE -16297,0       SELECT HI-RES GRAPHICS MODE
POKE -16298,0       SELECT LO-RES GRAPHICS MODE

POKE -16299,0       SELECT SECONDARY DISPLAY BUFFER
POKE -16300,0       SELECT PRIMARY DISPLAY BUFFER

POKE -16301,0       SELECT MIXED GRAPHICS + TEXT MODE
POKE -16302,0       SELECT FULL SCREEN GRAPHICS

POKE -16303,0       SELECT TEXT MODE
POKE -16304,0       SELECT GRAPHICS MODE

CALL -1994          CLEAR UPPER 40 LINES OF LO-RES PRIMARY BUFFER
                          AREA

CALL -1998          CLEAR FULL SCREEN (48 LINES) OF LO-RES
                          PRIMARY BUFFER AREA

CALL 62450          CLEAR HI-RES BUFFER AREA (WHICHEVER ONE THAT
                          IS CURRENTLY IN USE) TO BLACK.

CALL 62454          CLEAR HI-RES BUFFER AREA (WHICHEVER ONE THAT
                          IS CURRENTLY IN USE) TO THE LAST COLOR
                          SPECIFIED BY AN HCOLOR COMMAND AND USED
                          BY A PLOT COMMAND.
```

Table 2. Switch controls that can be used to specify the screen display attributes at any given time, and some useful CALL instructions.

you're using full screen graphics in hi-res mode, in the secondary buffer area, and then POKE −16301,0 to switch from full screen graphics to mixed text and graphics, the bottom four lines of display will reveal part of your program code!

To place text material into the bottom four lines of the screen display when using the primary screen buffer in mixed text and graphics mode, use the VTAB instruction to locate the printed material on the line desired (21–24). To return to text mode when the screen is displaying graphics, use the command TEXT:HOME. The reset key will return the screen to text mode also.

Figure 3 lists several commands that, when used as a group, define the specific screen mode at any time. Four "switches" are controlled by the POKE commands. These switches determine the following four attributes of the screen display:

1. Text display or graphics display.
2. Primary buffer or secondary buffer.
3. Hi-res or lo-res (if in graphics mode).
4. Full screen or mixed text and graphics (if in graphics mode).

In addition to these commands, four other useful assembly-language subroutines can be called from Basic using the CALL commands listed in Figure 3. They provide functions that are similar to the HOME instruction for the text mode.

Lo-Res Graphics

When working with graphics on any system, you will find that a tablet of graph paper is your most valuable tool. This is true whether you are using lo-res graphics or hi-res graphics. Outline the screen on the paper, encompassing the same number of squares that your particular graphics mode supports on the video display (e.g., 40 by 48 for lo-res full screen mode). Then sketch in your figure(s) just as you would have them appear on the display, filling in the squares completely. If you wish, you might use colored pencils to correspond to the colors you will use in the display. You should then number the columns (0–39) and rows (0–47).

Your Basic program will now begin the graphics display by using the following procedure: first select lo-res graphics mode and, if desired, switch the display to full screen graphics. You may also wish to blacken the full screen (GR just cleared the top 40 lines) with a CALL −1998. Then specify the color you plan to use for the first block to be drawn.

```
10 REM DEMONSTRATION PROGRAM THAT FRAMES A BOX IN ONE COLOR
11 REM           AND FILLS IT WITH ANOTHER COLOR.  TO CHANGE
12 REM           BOX POSITION AND DIMENSIONS, CHANGE THE
13 REM           VALUES IN LINE 30.
20 HOME
30 C1=15: C2=10: R1=8: R2=20
40 GR
50 COLOR=2: GOSUB 100
60 COLOR=1: GOSUB 200
70 END
100 REM SUBROUTINE TO DRAW BOXES; C1 & R1 ARE UPPER LEFT CORNER
101 REM            COLUMN AND ROW; C2 & R2 ARE WIDTH AND HEIGHT
110 IF C1+C2 > 40 THEN C2 = 40-C1
120 IF R1+R2 > 40 THEN R2 = 40-R1
130 HLIN C1, C1+C2-1 AT R1
140 HLIN C1, C1+C2-1 AT R1+R2-1
150 VLIN R1, R1+R2-1 AT C1
160 VLIN R1, R1+R2-1 AT C1+C2-1
170 RETURN
200 REM SUBROUTINE TO PAINT BOXES IN; C1 & R1 ARE UPPER LEFT
201 REM            CORNER OF BOX FRAME.  C2 & R2 ARE WIDTH
202 REM            AND HEIGHT.
210 FOR I=R1+1 TO R1+R2-2
220     HLIN C1+1, C1+C2-2 AT I
230 NEXT I
240 RETURN
```

Listing 1. Two subroutines that can be used to frame a box in any color specified and to fill the box in with a second color.

```
10 GR: POKE – 16302, 0: CALL – 1998
```

Now if you have an orange block to fill in at column 18, row 15, use the instruction

```
20 COLOR = 9: PLOT 18,15
```

The color may be redefined at any time with another COLOR = ## command. In this way you may fill the entire screen, one block at a time.

Fortunately, there are several short cuts to this technique. If you wish to draw a horizontal line with blocks having the same color, the HLIN...,... AT ... instruction may be used. Similarly, for vertical lines there is the VLIN...,... AT ... command. Both of these commands require that three numbers be included to complete the command. Table 1 indicates what these numbers represent for each of the two commands. So if we wanted to draw a vertical line through the center of the screen (column 19) in blue, we would add

```
30 COLOR = 2: VLIN 0,47 AT 19
```

and to add a horizontal line through the center of the screen that is green on the left side of the vertical line and yellow on the right side you would need:

```
40 COLOR = 12: HLIN 0,18 AT 23
50 COLOR = 13: HLIN 20,39 AT 23
```

So much for individual blocks, horizontal and vertical lines. With that we can draw boxes on the screen. But what if we want a diagonal line? For example, we may wish to draw a bright red X across the screen. To make the X symmetrical, start at row 4 for the top of the letter and end at row 44 for the bottom. In that way you will traverse 40 rows as you traverse the 40 columns of the screen.

The FOR/NEXT loop can be used to good advantage for diagonal lines. In this case, you want the diagonal line to be drawn by incrementing both the column and row values by 1 after each block is drawn. That is, you want to plot point 0,4 first, then point 1,5; then point 2,6; etc. To accomplish this with a simple loop, the following lines should be added to your program:

```
60 COLOR = 1
70 FOR I = 0 TO 39
80    PLOT I, 4 + I
90 NEXT I
```

As you can see, the loop counter I is used for determining both the column position and the row position. Also note that the PLOT instruction (as well as the HLIN and VLIN instructions) can use variables or mathematical expressions to specify the column and row positions.

Lines 60 through 90 plot half of the X. The loop counter steps the program through 40 PLOT instructions. In order to draw the other half of the X, it will be necessary to step through another 40 PLOT instructions. A similar loop could be written to accomplish this task. The program can be made more efficient, however, if you plot the second diagonal line within the same loop that has already been set up. Add the following line to your program:

```
85 PLOT I, 44 – I
```

This takes care of the second diagonal line. Notice the difference in the two PLOT commands of line 80 and 85 as far as the row specification is concerned. In line 80, each time through the loop the row position is incremented by 1 (just as the loop counter is incremented by 1). In line 85, each time through the loop, the row position is decreased by 1.

Paint

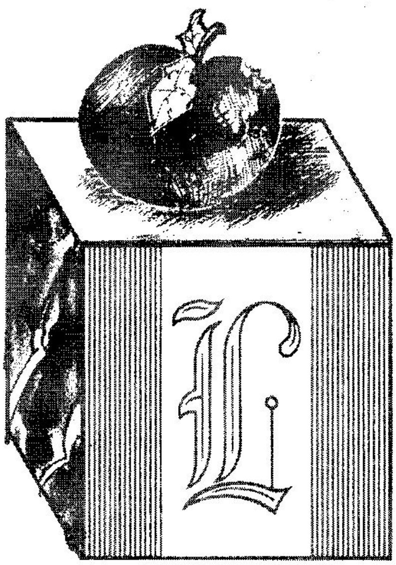
Some Basics have a PAINT command that allows the programmer to fill in geometric figures on the screen. However, proper use of FOR/NEXT loop instructions, using the loop counter as part of the PLOT instruction, enables any Basic to accomplish this task. Usually the filling in of a figure on the screen will require two nested loops. In Applesoft Basic, however, the HLIN and VLIN instructions act as built-in loops and their use can allow a figure to be "painted in" with just one FOR/NEXT loop.

Listing 1 contains a short program that uses two subroutines. The first will outline a rectangular box on the screen in a given color, while the second will fill the box in with a second color. Before calling the subroutines, the column and row position (C1 and R1) of the top left corner of the box must be defined. Similarly, the width of the box (C2) and its height (R2) must be given. By changing the values in line 30, you can construct boxes of any size anywhere on the screen. You can even make the entire screen into a box, with lots of smaller boxes inside it.

Lo-Res Animation

Now the fun begins. It is just a short step from creating static displays to drawing objects that appear to move across the screen. The secret to motion is that if an object is drawn in the same color as the background color, the effect is the same as if the object were erased. Thus by drawing an object on the screen in some color

APPLES
APPLES
APPLES
. . . So What



I'll Tell You What . . .

Now you can put a bigger byte in your Apple. **NOT JUST**

16K RAM on one Board
or 64K RAM on one Board
or even 128K RAM on one Board

NEW from the legendary leader, you get the extraordinary **S-CARD** combination, only from Legend, to increase your Apple's byte from 64K RAM to one Megabyte on one RAM Board.

The fantastic **S-CARD** combination!
From 64K RAM to one Megabyte
user upgradable
with just one RAM Board
only from Legend Industries, Ltd.
the legendary leaders in Apple
compatible products.

Legend Industries Ltd.
2220 Scott Lake Rd.
Pontiac, MI 48054
(313) 674-0953

Ask about our complete line of products at your local dealer.

AUTHORIZED DISTRIBUTORS:

EUROPE
B.I.P.
220 Joseph Disjon
75018 Paris, France
(1) 255-4463

WEST COAST
Group 3
2613 Manhattan Beach Blvd.
Redondo Beach, CA 90278
(213) 973-7844

NEW ENGLAND
CDS
226 Route #122
Amherst, NH 03031
(603) 673-7375

FLORIDA
World Wide Products
Electronics, Inc.
10818 N.W. 6th Court
P.O. Box 680279
Miami, FL 33168
(305) 754-5475

EAST COAST and MIDWEST
Legend Industries, Ltd.
2220 Scott Lake Rd.
Pontiac, MI 48054
(313) 674-0953
Ask for Joe Krajenke
or Norm Dybowski

Circle 104 on Reader Service card.

```
10 REM JUMPING JACK - SIMPLE LO-RES ANIMATION EXAMPLE
20 GR
30 COLOR=2: GOSUB100
40 COLOR=13: GOSUB 400: GOSUB 200: REM DRAW FIGURE A
50 COLOR=0:  GOSUB 400:            REM ERASE FIGURE A
60 COLOR=13: GOSUB 500: GOSUB 210: REM DRAW FIGURE B
70 COLOR=0:  GOSUB 500:            REM ERASE FIGURE B
80 GOTO 40
100 HLIN 15,25 AT 39:RETURN:       REM DRAW PLATFORM
200 FOR I=1 TO 200: NEXT I: RETURN: REM LONG PAUSE
210 FOR I=1 TO 50: NEXT I: RETURN:  REM SHORT PAUSE
400 REM SUBROUTINE TO DRAW FIGURE STANDING STRAIGHT
410 VLIN 34,38 AT 18
420 VLIN 33,38 AT 19
430 VLIN 33,38 AT 20
440 VLIN 34,38 AT 21
450 RETURN
500 REM SUBROUTINE TO DRAW FIGURE B STRADDLED
510 PLOT 17,32: PLOT 22,32
520 HLIN 17,22 AT 34
530 PLOT 17,33: PLOT 22,33
540 VLIN 33,36 AT 19
550 VLIN 33,36 AT 20
560 PLOT 18,37: PLOT 21,37
570 PLOT 17,38: PLOT 22,38
580 RETURN
```

Listing 2. Jumping-Jack program to illustrate fundamental principles involved with computer animation.

(with black background), then redrawing that object, using exactly the same coordinates but using COLOR = 0 (black), then drawing the object again in the original color but using slightly different coordinates, you will have made the object appear to have moved.

The simplest example of this technique is a short program that causes a block to appear to move across the screen from left to right. A pause loop is inserted into the program at line 50 so the progress of the block can be more easily observed.

```
10 GR
20 FOR I = 0 TO 39
30     COLOR = 12
40     PLOT I,20
50     FOR J = 1 TO 50: NEXT J
60     COLOR = 0
70     PLOT I,20
80 NEXT I
```

To get the block to return from right to left, add line 120 (below) and repeat lines 30 through 80 using line numbers 130 through 180.

```
120 FOR I = 39 TO 0 STEP -1
```

Then add line 200:

```
200 GOTO 20
```

You now have a block that seems to bounce back and forth across the screen.

To achieve the same effect with a more complicated figure, use a subroutine filled with the necessary PLOT information to define the figure. Then in lines 40, 70, 140 and 170 use a GOSUB instruction instead of the PLOT commands to jump to this subroutine. In this way a static figure (such as a ball, flying saucer or submarine) can be drawn, erased and redrawn, causing the figure to move back and forth across the screen.

If the figure itself is to show movement, such as an animal, human figure or bird, in which arms, legs or wings alternate between different positions, then two or more subroutines must be called in succession, with each subroutine showing the figure in a different position. Each cycle through the loop will take the figure through all of its various positions. The basic principle of drawing the figure, erasing it and then redrawing it in a different position and location remains the same as before.

Listing 2 contains a very simple example of this technique. Although not very sophisticated with the block graphics of lo-res, and using only slow Basic commands, this program does illustrate the concepts described above. The figures must be kept small so that the Basic PLOT commands can complete the figure as quickly as possible. The same techniques work with hi-res graphics as well, and can produce some really astonishing effects, particularly when more sophisticated imaging techniques are used.

Detecting a Hit

Only one additional command needs to be introduced in order to complete the repertoire of lo-res graphics commands (with the exception of a a few POKEs to be mentioned later). Many games are designed around the idea of shooting at or bombing a target or intercepting a ball. There must be some way of determining when an attempt to plot a point on the screen results in placing the point over a screen position that has already been plotted. The command that does this uses the SCRN(aa,bb) function. This function "looks at" the screen position that has column aa and row bb and returns a number that corresponds to the color found at that location. If the screen display is using black for a background, then the statement

```
X = SCRN(15,32)
```

will return a value of 0 if position 15,32 has not been lit up with a PLOT instruction, or some value between 1 and 15 if it has been lit. So before plotting the new position for the tennis ball, for instance, the proposed position is checked to see if the SCRN() value for that position is still 0. If it is, the ball is redrawn to the new position. If the SCRN() value for that position is not 0, then a hit must have been made with either the racket or with the border at the edge of the screen. By using different colors for the racket and the border, the value of X will also tell you which object the ball bounced against.

Lo-Res Histogram Construction

Listing 3 illustrates a useful application of lo-res graphics to business applications. Again, this program has been designed to illustrate the concepts presented in this article and thus lacks frills. Even so, it stands as a useful program without further additions. The program allows the user to construct a histogram (bar graph) for up to five different sets of data. Each set will be represented by a different color, with bars drawn side by side on the screen. A total of

```
  10 REM LO-RES HISTOGRAM (BAR GRAPH) PLOTTER
  20 REM     SD$ & SD  NUMBER OF DATA SETS
  30 REM     NE$ & NE  NUMBER OF ELEMENTS IN EACH SET
  40 REM     D$(I)     NAME FOR EACH DATA SET
  50 REM     C(I)      COLOR FOR EACH DATA SET
  60 REM     D(I,J)    J'TH DATA ELEMENT FOR SET I
  70 REM     MX        MAXIMUM DATA VALUE ENTERED
  80 REM     I & J     LOOP COUNTERS USED INTERNALLY
  90 REM
 100 HOME
 110 DIM C(5), D$(5)
 120 GOSUB 500
 130 GOSUB 1000
 140 GOSUB 800
 150 VTAB 24:INPUT"PRESS (RETURN) TO END...";X$
 160 TEXT: HOME: END
 490 REM DATA ENTRY SUBROUTINE
 500 INPUT"HOW MANY SETS OF DATA (5 MAX)? ";SD$
 510 SD=VAL(SD$): IF SD<1 OR SD>5 THEN 500
 520 PRINT"HOW MANY ELEMENTS IN EACH SET (";INT(40/SD);" MAX)?";
 530 INPUT" ";NE$ : NE=VAL(NE$)
 540 IF NE<1 OR NE>INT(40/SD) THEN PRINT"TOO MUCH DATA":GOTO 500
 550 FOR I=1 TO SD
 560     PRINT"KEY NAME FOR DATA SET #";I;
 570     INPUT": ";D$(I)
 580 NEXT I
 590 DIM D(SD,NE)
 600 C(1)=1: C(2)=7: C(3)=9: C(4)=12: C(5)=2
 610 MX = 0
 620 FOR I=1 TO SD
 630     HOME: PRINT"DATA FOR ";D$(I)
 640     FOR J=1 TO NE
 650         PRINT"SET #";I;" - ELEMENT ";J;":  ";
 660         INPUT"";D$
 670         D(I,J)=VAL(D$)
 680         IF D(I,J)> MX THEN MX=D(I,J)
 690     NEXT J
 700 NEXT I
 710 RETURN
 800 GR: HOME: REM BAR GRAPH PLOTTING SUBROUTINE
 810 FOR J=1 TO NE
 820     FOR I=1 TO SD
 830         COLOR=C(I)
 840         VLIN 40-D(I,J),39 AT I+SD*(J-1)-1
 850         VTAB 21: IF J/2=INT(J/2+.1) THEN VTAB 22
 860         PRINT LEFT$(D$(I),1);
 870     NEXT I
 880 NEXT J
 890 RETURN
 990 REM SUBROUTINE THAT NORMALIZES DATA TO FIT SCREEN DISPLAY
1000 FOR I=1 TO SD
1010     FOR J=1 TO NE
1020         D(I,J) = INT(D(I,J)*40/MX)
1030     NEXT J
1040 NEXT I
1050 RETURN
```

Listing 3. Business Application for lo-res graphics. This program plots a histogram (bar graph) for up to 40 elements which may be divided into up to five sets or categories.

40 different elements may be displayed. So if a single set of data is being used, 40 elements from that set may be entered. If two sets are being compared, then each set may have up to 20 elements. If five sets are used, then each set is limited to eight elements.

For example, suppose you wished to illustrate the relative GNP for the United States, Germany, Great Britain and France over the past ten years. You will have four sets of data, and each set will contain ten data elements. The program will ask for the number of sets you wish to have displayed (five max) and will ask you to provide a name for each set. The display will use only the first letter of the name, so choose names for each set that start with a different letter. For example, the four countries listed above could be named United States, Germany, Britain and France.

You are then asked for the number of elements to be included in each set. The computer will reject any entry here that will result in more than 40 total elements to be plotted. The program uses this information to dimension the data array, and you are then prompted to enter the data elements for each set, one complete set at a time.

In order to make the most use of the screen height, the program keeps track of the data elements as they are entered and saves the single element with the largest value as MX. This element is then used to adjust the relative heights of all of the other elements so that MX will rise from the base line to the top of the screen. This process might be referred to as a normalization, with each element being multiplied by 40 and divided by the value of MX. Thus their relative values will remain the same, but all of the elements will now have values between 0 and 40 and will thus fit onto the video screen.

The plotting routine displays the first elements from each set, side by side; then the second elements of each set; then the third; etc. Each set is represented by a different color, as well as a one-letter prompt on the text display line in rows 21 and 22, so the different sets can be differentiated and compared with ease.

Another example using this program might be to compare the number of older homes sold vs. new homes sold for the last 20 months within the city or county. Or how about comparing interest rates vs. number of real estate sales over the last 20 months. A note of caution about the values you choose when comparing different items, however. This program assumes that all values will be in the same ball park. So if your interest rates run from 9 percent to 18 percent and the property sales run from 120 to 350, then multiply the interest rates by 15 before entering them. The range of interest rates then becomes 135 to 270 (relative values), which can then be plotted side by side with the property sales.

Conclusion

With the procedures described in this column mastered, you will find little difficulty in jumping into hi-res graphics, which will be the subject for next month's column. I will demonstrate a method for using the shape table capability of the Apple that is simpler to use than the method described in the *Applesoft Manual*, and suggest a method for allowing the use of both hi-res buffer areas, even when the Basic program extends into one or both of these areas. ■

The Assembly Advantage

by Randy Hyde

Control Structures

In the previous three installments of this column I've described how to define and use SPEED/ASM variables and how to perform the standard integer arithmetic operations, and I've discussed some 6502 code necessary to write complete SPEED/ASM programs. With most of the basics behind us, its' time to begin a consideration of SPEED/ASM's control structures.

The FOR. . .NEXT Loop

SPEED/ASM supports two variations on Basic's FOR...NEXT loop: FOR and FOR0. The FOR subroutine is a generalized FOR loop that allows variable initial/ending values and a stepsize. The FOR0 routine is a specialized version of FOR that assumes the initial and ending values are constants and that the stepsize is one. Since the FOR0 loop is used better than 90 percent of the time, I will describe it first.

The FOR0 loop uses the structure:

```
        JSR    FOR0
        ADR    IVAR,STARTVAL,ENDVAL

; Body of loop

        JSR    NEXT
```

IVAR must be the name of a SPEED/ASM integer variable; STARTVAL and ENDVAL *must* be integer constants. If you specify a SPEED/ASM variable name for STARTVAL or ENDVAL, SPEED/ASM will use the *address* of the variable, not the contents of the variable, as the initial or final value.

Since most FOR loops take the form:

FOR I = 1 TO 10

the FOR0 routine turns out to be quite adequate for most applications. See Listing 1 for a sample Basic FOR... NEXT routine and its SPEED/ASM equivalent.

As in Basic, it is illegal to jump into the middle of a FOR...NEXT loop in SPEED/ASM. Unlike Basic, SPEED/ASM isn't nice enough to tell you that you've executed a NEXT without a

Address correspondence to Randy Hyde, 925 Lorna St., Corona, CA 91720.

```
10 FOR I = 1 TO 20
20 PRINT "I = ";I
30 NEXT I
40 END
```

Listing 1a. Sample Basic FOR...NEXT routine.

```
EXIT    EQU    $FF69
;
        JSR    INIT       ;Always before running a SPEED/ASM PGM.
        JSR    FOR0
        ADR    I,1,20
;
;Body of loop
;
        JSR    PRINT
        BYT    "I = ",0
        JSR    PRINT
        ADR    I
        LDA    #CR        ;Print the return at the end
        JSR    PUTC       ;of the line.
;
        JSR    NEXT
        JMP    EXIT
;
;Variable declaration(s)
;
I       ADR    0
        END
```

Listing 1b. SPEED/ASM equivalent.

```
        JSR    FOR0
        ADR    I,1,10
;
;Body of loop
;
        JMP    EXITLP
;
        JSR    NEXT
        JMP    DIDNTXIT
;
;Exit condition at this point.
;
EXITLP  PLA
        PLA
        PLA
        PLA
        PLA
        PLA
        PLA
        PLA
;
DIDNTXIT:
```

Listing 2. Popping data off a stack.

Is 42 seconds worth $1000?

The new Transtar 130 daisy wheel printer generates a full-page letter in 78 seconds. The least expensive 40 cps printer does it in 36. Only 42 seconds difference...for <u>twice</u> the price.

At only $895, the Transtar 130 letter-quality printer makes speed its only compromise. Shannon-text rated at 16 cps, the Transtar 130 gives you better printing quality than any 40 cps printer. It allows you the full range of word-processing functions such as proportional spacing, superscript, subscript, underscoring and a true boldface. The 130 is "plug and go" compatible with the best-selling word processing packages. It's quiet: only 65dB. It's durable. It boasts a unique new autoload feature that automatically loads paper to one of four preselected positions with the touch of a button. And, as if that weren't enough, its end-user warranty runs a full six months—twice that of most of its competitors.

Affordable and loaded with all the features of printers costing twice as much, the Transtar 130 letter-quality printer retails for less than $900. But it's up to you: Is 42 seconds really worth $1000?

Transtar

Box C-96975, Bellevue, Washington 98009

Circle 280 on Reader Service card.

matching FOR. Instead, the system simply hangs up (or begins doing bizarre things to the screen or accessing your peripheral devices). Therefore, you should always make sure that there are no jumps into the range of a FOR loop. For example, the following is definitely forbidden:

```
JMP    ENTERFOR ;Can't do!
       JSR    FOR0
       ADR    I,1,10
;
ENTERFOR:
;
       JSR    NEXT
;
```

It is in equally bad taste to jump outside the range of a FOR...NEXT loop from *within* a FOR loop. Basic also disallows this, but is much more forgiving. The latter won't catch the error until you overflow the FOR...NEXT stack (by doing it too many times). SPEED/ASM, on the other hand, explodes spectacularly the next time you execute an RTS or otherwise attempt to access the top of the stack.

The reason a jump into or out of a FOR...NEXT loop causes problems is that the FOR subroutine pushes data onto the top of the stack and leaves it there. This data is popped off and processed by the NEXT subroutine later on. Obviously, if you call the FOR routine from within a subroutine and then execute an RTS without first completing the loop by calling the NEXT routine, the 6502 will attempt to use the data pushed onto the stack by the FOR routine as the return address. Typically this will not return you to the spot you're interested in.

For those of you who regularly use the POP command in Applesoft, yes, you can pop this data off the stack and prematurely exit a FOR...NEXT loop. The SPEED/ASM FOR...NEXT loop pushes 8 bytes of data onto the stack so you can repair the stack by popping 8 bytes off. This is accomplished by executing eight PLA instructions in a row. See Listing 2.

The FOR subroutine is similar to the FOR0 subroutine. The major difference is that the FOR subroutine allows variable starting and ending values and a variable stepsize. The exact syntax for the FOR subroutine is:

```
JSR    FOR
ADR    <index var>,<start var>, <end var>,
<step var>
```

where index var is the name of the variable used as the loop index, start var is the name of the variable containing the starting value, end var is the name of the variable containing the ending value, and step var is the name of the variable containing the stepsize. To simulate the Basic statements:

```
10 FOR I = J TO K STEP STP
20 NEXT I
```

you would use the SPEED/ASM code:

```
JSR    FOR
ADR    I,J,K,STP

JSR    NEXT
```

Note that the FOR loop accepts only variable names; constants are not allowed. If either the starting or ending value must be a variable, or your loop requires a stepsize, then you must use a FOR loop and all the values must be specified as variables. For example, consider the Basic loop:

```
10 FOR I = J TO 100
20 NEXT I
```

Because the starting value (J) is specified as a variable, the FOR0 routine cannot be used. The FOR routine, however, requires that the starting, ending and stepsize values all be specified as variables. Therefore, to convert the statement above into SPEED/ASM code you will need to create two "dummy" variables: one to hold the ending value constant (100) and one to hold the stepsize constant (1). Use the code in Listing 3.

Note that the constants/variables C100 and C1 are sandwiched between a JMP instruction and the beginning of the loop. Since these values are static (they do not change) they should be incorporated into the code instead of standing at the end of the program with the variables. Although they are constant values they still must not be executed as 6502 instructions. Hence, a JMP instruction was executed to skip past the constants.

The FOR subroutine is very powerful, even though it may be somewhat cumbersome to use if the starting, ending and/or stepsize values are constants instead of variables. The inconvenience is actually minimal since most of the time the FOR0 subroutine is used instead of the FOR subroutine.

```
         JMP    STRTLP
C100     ADR    100
C1       ADR    1
;
STRTLP   JSR    FOR
         ADR    I,J,C100,C1
;
;
;Body of loop
;
         JSR    NEXT
```

Listing 3. A FOR routine using "dummy" variables.

```
1 J = 2
5 I = 1
10 IF I> = 10 THEN GOTO 20
11 I = I*J
12 GOTO 10
20 END
```

Listing 4a. Sample Basic IF routine incorporating a GOTO statement.

```
EXIT     EQU    $FF69
;
         JSR    INIT
         JSR    LOAD
         ADR    2,J
         JSR    LOAD
         ADR    1,I
L10      JSR    IFI0
         ADR    I,GE,10
         BTR    L20
         JSR    MUL
         ADR    I,J,I
         JMP    L10
;
L20      JMP    EXIT
         END
```

Listing 4b. SPEED/ASM equivalent using BTR.

The SPEED/ASM IFI And IFI0 Routines

SPEED/ASM uses two routines to compare integer values: IFI and IFI0. IFI compares two integer variables; IFI0 compares an integer variable to an integer constant. Six types of comparison are possible. The IFI and IFI0 routines return "true" if a com-

Listing 6. IF and FOR Demo.

```
0800            TTL "Listing  6 : IF and FOR Demo"
0800     ;
0800     ;
0800     ;
0800     **********************************
0800     *                                *
0800     *      SPEED/ASM Equates         *
0800     *                                *
0800     **********************************
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     *************
0800     * CONSTANTS *
0800     *************
0800     ;
0800     ;
0800     ;
0800     ; The following symbols are constants
0800     ; for the values "FALSE", "TRUE", and
0800     ; Carriage Return (respectively).
0800     ;
0800     ; These symbols should only appear
0800     ; as immediate operands to a 6502
0800     ; instruction or in the operand field
0800     ; of a pseudo-opcode like BYT.
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0000     FALSE      EQU 0
0001     TRUE       EQU 1
008D     CR         EQU $8D
0087     BELL       EQU $87
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ; "IF" STATEMENT EQUATES
0800     ;
0800     ; The following symbols should only
0800     ; be used in the ADR pseudo-opcode
0800     ; following a call to the SPEED/ASM
0800     ; IFx routines.
0800     ;
0800     ;
0800     ;
00BD     EQ         EQU "="
00A3     NE         EQU "#"
00BE     GT         EQU ">"
00BC     LT         EQU "<"
BDBE     GE         EQU ">"|"="*256
BDBC     LE         EQU "<"|"="*256
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     **************************
0800     * SPEED/ASM ENTRY POINTS *
0800     **************************
0800     ;
0800     ;
0800     ;
0800     ;
0800     ; NOTE: THE EQUATE OF PUTC MUST
0800     ; BE CHANGED IF YOU RELOCATE
0800     ; SPEED/ASM TO SOME LOCATION
0800     ; OTHER THAN $7800
0800     ;
0800     ;
0800     ;
7800     PUTC       EQU $7800
7803     GETC       EQU PUTC+3
7806     SAGL       EQU GETC+3         ;FOR USE BY S/A ONLY- SEE DOC.
7809     SAPC       EQU SAGL+3         ; "   "   "  "   "    "   "
780C     HOME       EQU SAPC+3         ;HOME AND CLEAR
780F     READLN     EQU HOME+3
7812     INIT       EQU READLN+3
7815     FOR        EQU INIT+3
7818     FOR0       EQU FOR+3
781B     NEXT       EQU FOR0+3
781E     IFI        EQU NEXT+3
7821     IFI0       EQU IFI+3
```

Listing continued.

parison holds, "false" if it does not.

You can call the IFI0 routine with the format:

```
JSR   IFI0
ADR   <var1>,<op>,<value>
```

where var1 is a SPEED/ASM integer variable, value is any integer constant, and op is the operator specifying which comparison to perform. The latter may be: EQ—test for equality; NE—test for inequality; LT—test for less than; LE—test for less than or equal to; GT—test for greater than; or GE—test for greater than or equal to. For example, to see if IVAR is less than 4376 use the statement:

```
JSR   IFI0
ADR   IVAR,LT,4376
```

The EQ, NE, GT, GE, LT and LE symbols are defined for you in SPEED/ASM Equates, Listing 7. Refer there for the values corresponding to these symbols.

The IFI0 and IFI routines return "true" or "false" in the 6502 accumulator (where true = 1 and false = 0), and set the 6502 Z bit in the P register so that the BTR and BFL (branch-if-true and branch-if-false) instructions can be used immediately after the call to IFI0 or IFI. To emulate the Basic IF statement you need only add a branch statement to the IFI0 or IFI statement to make it fully functional. Refer to Listing 4 for an example.

This relatively complete SPEED/ASM program (it still needs the

```
IF I < >10 THEN J = -I
```

Listing 5a. Sample Basic IF routine.

```
     JSR   IFI0
     ADR   I,NE,10
     BFL   INE10
     JSR   MOVE
     ADR   I,J
     JSR   NEG
     ADR   J
;
INE10:
```

Listing 5b. SPEED/ASM equivalent using BFL.

Circle 381 on Reader Service card.

H & W COMPUTER SYSTEMS

MAIL ORDER

SOFTWARE

WORD HANDLER	130.00
RENDEZVOUS	29.50
ZORK I, II, III	28.25
DEADLINE	39.00
STARCROSS	39.00
SOFTCARD	250.00
GOLD RUSH	26.00
LISA 2.5	60.00
ULTIMA II	42.00

MASTERCARD & VISA

CALL or SEND for CAT.

(702)727-4274

PO BOX 1207
PAHRUMP,NV 89041

Circle 366 on Reader Service card.

CHECK-POINT

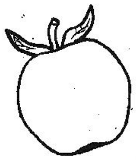
-LOAD OR SAVE 64K VISICALC MODELS IN 13 SECONDS-

MAKE BACKUP COPIES OF MEMORY RESIDENT 48K AND 64K SOFTWARE

-FREEZE GAMES AT CRITICAL PLAY POINTS-

-DEBUG TROUBLESOME SOFTWARE-

II or IIe



CHECKPOINT is a sophisticated hardware/software system that captures/restores the memory image and operation of your Apple II or IIe including display and graphic modes. CHECKPOINT creates an image of your system on disc with a single push of a button. CHECKPOINT is undetectable until called into action. CHECKPOINT disengages after backup is created and Apple operation resumes automatically at point of interruption. Reloading is just as easy. No Technical knowledge is required.

COMPARE OUR FEATURES

NEW!

* No complicated codes or instructions like nibble copiers, Just press one button
* No complicated or time consuming conversion utilities.
* Extra fast operation, copy/resume or load 48K software in 10 seconds, 64K in 13 seconds
* Full two disc operation if desired, load from one disk, copy to the other
* Full function hardware, 16K card NOT required. No chips to pull.
* 16K card, if it exists, is not altered nor does it need to be removed
* Friendly, easy to use - menu driven and self prompting
* ALL CHECKPOINT software and files created are DOS compatible and copyable
* Autoload feature for automatic boot, load, and run
* Easy installation, just plug into vacant slot
* 16K card can be used for reload if desired, to free slot for other uses
* No need to lift cover to operate
* Debug/analysis software provided free
* Comprehensive, non-technical users manual
* Every system fully assembled and tested with 1 year guarantee
* Same product works on II or new IIe with no changes.
* Software loads without any additional hardware on IIe.

II or IIe

-DON'T SETTLE FOR LESS, OURS IS THE BEST-

INTRODUCTORY PRICE: $134.95

Free shipping with check or money order COD $5 extra (Conn. residents add 7.5% sales tax)

APPLITEK INC. BOX 5114 BRIDGEPORT, CT 06610 203-378-0623

* Systems Requirements: 48K or 64K Apple II plus. or IIe, 1 or 2 disc drives with DOS 3.3

* Apple is a registered trademark of Apple Computer, Inc. * VisiCalc is a registered trademark of VisiCorp, Inc.

CHECKPOINT IS INTENDED FOR LEGITIMATE BACKUP AND SYSTEM USES AND SHOULD NOT BE USED FOR ILLEGAL PURPOSES

FREE SHIPPING

IBM® Personal Computer Products

Davong 5 MB Hard Disk System – **$1495.00** 12 MB – **$1995.00**

IBM PC-2 Drive System **$ CALL**

3" DUAL DRIVE SUBSYSTEM **$725.00**

Quadram – Quadboard with Parallel Port, Serial Port, Clock/Calendar, Expandable to 256K

64K on brd. – **$340.00** 128K on brd. – **$395.00**

192K on brd. – **$439.00** 256K on brd. – **$499.00**

Quadram Memory Expansion

192K Maximum

64K on brd. – **$230.00** 128K on brd. – **$290.00**

192K on brd. – **$350.00**

AST & PERSYST MEMORY EXPANSION PRODUCTS **$ CALL**

Amdek Monitors

Mod 300 Phosphor – **$150.00** Composite Color III – **$345.00**

IBM RGB Compatible Color II – **$599.00** Color I – **$300.00**

IBM/TRS 80 Disk Drives/Cabinets

TM 100 Single 40 Track Drive – **$189.00**

TM 100-2 Double 40 Track Drive – **$280.00**

TEAC 5¼" SLIM SINGLE & DOUBLE DRIVE SUBSYSTEMS **$ CALL**

Apple II® Computer Products

SYSCOM APPLE COMPATIBLE SYSTEM **$625.00**

Apple Compatible Controller Card .... **$ 79.95**

Apple Compatible Disk Drive w/Cabinet & Cable **249.00**

w/Controller **315.00**

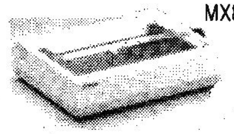
TEAC SLIM LINE 5¼" DRIVE $265.00 **DUAL SLIM LINE $525.00**

Printer/Graphics Interface **99.95**

Davong 5 MB Hard Disk System – **$1495.00** — 12 MB – **$1995.00**

Apple Compatible Joysticks **29.95**

Epson/Smith-Corona Printers



MX80 **$425.00** FX80 **$550.00** MX100 **$675.00**

TRS 80 / Parallel Printer Cable **$20.00**

IBM Parallel Printer Cable **$35.00**

STAR MICRONICS GEMINI 10 $ CALL

GEMINI 15 $ CALL

Smith Corona TP-1 Letter Quality Daisy Wheel **$575.00**

JMR ELECTRONICS INC.

TRS-80 MOD III Disk Controller Incl: Disk Controller, Power-Supply, Mounting Hardware, Cables & Instruction Manuals **$239.00**

POWER SUPPLIES AND CABINETS Dual 8" Slim Line – **$180.00** Dual 5¼" – **$ 99.00** Single 5¼" – **$ 69.00**

3" DUAL DRIVE SUBSYSTEM FOR IBM **$725.00**

VISA, MASTERCARD ($100 Min., Add 2%)
Or Certified Check
90 Day Warranty (Parts & Labor)
TRS 80 is a Registered Trademark, Tandy Corp.
Prices Subject to Change Without Notice

DATA—MAIL
P.O. Box 818. Reseda. CA 91335
1-800-635-5555

FREE SHIPPING IN CONTINENTAL U.S.

(213) 993-4804 (IN CALIF.)

Circle 351 on Reader Service card.

Listing continued.

```
7824          IFS      EQU IFI0+3
7827          IFS0     EQU IFS+3
782A          MOVE     EQU IFS0+3
782D          LOAD     EQU MOVE+3
7830          MOVS     EQU LOAD+3
7833          LDSTR    EQU MOVS+3
7836          PRINT    EQU LDSTR+3
7839          PRTSTR   EQU PRINT+3
783C          PRTINT   EQU PRTSTR+3
783F          RDSTR    EQU PRTINT+3
7842          RDINT    EQU RDSTR+3
7845          ONXGOTO  EQU RDINT+3
7848          CASE     EQU ONXGOTO+3
784B          CASEI    EQU CASE+3
784E          INSET    EQU CASEI+3
7851          NOTINSET EQU INSET+3
7854          ABS      EQU NOTINSET+3
7857          NEG      EQU ABS+3
785A          MUL      EQU NEG+3
785D          DIV      EQU MUL+3
7860          MOD      EQU DIV+3
0800          ;
0800          ;
0800          ;
0800          ; Apple DOS equates:
0800          ;
03EA          INITDOS  EQU $3EA            ;DOS init routine.
0084          CTLD     EQU $84             ;Control-D character
0800          ;
0800          ;
0800          ; Apple Monitor equates
0800          ;
FF59          EXIT     EQU $FF59
0800          ;
0800          ;
0800          ;
0800          ;
0800          ;
0800          ;
0800          ***********************************
0800          *                                 *
0800          *  IF and FOR demonstration pgm.  *
0800          *                                 *
0800          *    This program creates a text  *
0800          *  file on the disk (it's named   *
0800          *  "NUMBERS") and writes out a    *
0800          *  series of numbers to the disk.*
0800          *  The second half of this pro-   *
0800          *  gram reads the data back in    *
0800          *  (in various formats) and dis-  *
0800          *  plays the input info on the    *
0800          *  screen.                        *
0800          *                                 *
0800          ***********************************
0800          ;
0800          ;
0800          ;
0800          ;
0800          ;
0800          ; Initialization section:
0800          ;
0800          ;
0800          ;
0800 20 12 78 SA.PGM.4 JSR INIT            ;Always do this first!
0803          ;
0803          ;
0803          ; The following call activates Apple DOS by
0803          ; "reconnecting" the input/output
0803          ; hooks at locations $36..$39  (See
0803          ; "Beneath Apple DOS" page 5-5).
0803          ;
0803 20 EA 03          JSR INITDOS
0806          ;
0806          ;
0806          ; Emulate the BASIC statement:
0806          ;
0806          ;   PRINT CHR$(4);"NOMON O,I,C"
0806          ;
0806 20 36 78          JSR PRINT
0809 8D 84 CE          BYT CR,CTLD,"NOMON O,I,C",CR,0
080C CF CD CF
080F CE A0 CF
0812 AC C9 AC
0815 C3 8D 00
0818          ;
0818          ;
0818          ;
0818          ;
0818          ***********************************
0818          ;
0818          ; File creation section:
0818          ;
0818          ;
0818          ;
0818          ; Open the file "NUMBERS" in the typical
0818          ; Apple DOS fashion.  Note that a
```

Listing continued.

SPEED/ASM equates at the top) exactly duplicates the Basic program. Note that the BTR (branch-if-true) instruction branches if the condition I> = 10 is true.

Sometimes you may want to execute an instruction other than a GOTO if an expression is true. For example, both Integer Basic and Applesoft Basic allow IF statements of the form IF <cond> THEN <statement>. This is easily simulated in SPEED/ASM by using the BFL (branch-if-false) instruction to branch around the statement you wish to execute. Listing 5 illustrates SPEED/ASM code of this type and a Basic equivalent.

In addition to setting the 6502 Z flag so that the BTR and BFL instructions can be used after an IFI0 instruction, the IFI0 routine returns "false" or "true" (0 or 1) in the 6502 accumulator register. This feature can be applied to several situations. Consider the Basic statement I = J< = 10. This assignment stores 0 into I if J is not less than or equal to ten; it stores 1 into I if J is less than or equal to ten. This action can easily be accomplished using the SPEED/ASM statements that follow:

```
JSR   IFI0
ADR   J,LE,10
STA   I
LDA   /0
STA   I+1
```

Don't forget that the high order byte of I must be set to 0. The LDA /0 and STA I + 1 statements take care of this problem.

You use the IFI routine to compare two integer *variables*. IFI's syntax is almost identical to that of IFI0. The only difference is that you specify a second SPEED/ASM variable instead of a numeric constant. The format for the IFI routine is:

```
JSR   IFI
ADR   <var1>,<op>,<var2>
```

Interfacing to Apple DOS

While I could go on discussing how the FOR and IF subroutines work, the best way to explain their use is through some concrete examples. Listing 6 uses FOR and IF to demonstrate how to create and access text files under Apple DOS.

As in Basic, in order to interface to Apple DOS you must print a control-D followed by a DOS command. Normally, Apple DOS only allows text files to be accessed from a running Basic program. Since SPEED/ASM is definitely not Basic, we must trick DOS into thinking that a Basic program is running. This is easily accomplished by storing the value $80 into locations $75 and $D9 in the Apple's 0 page memory space. This feat is accomplished using the 6502 code:

```
LDA  #$80
STA  $75
STA  $D9
```

Storing $80 into location $D9 tells DOS that an Integer Basic program is running; storing $80 into location $75 informs DOS that an Applesoft program is running. Whenever DOS receives a command to manipulate a text file it looks to see which Basic is currently active and then checks the appropriate 0 page location to determine whether or not the Basic is running. If location $75 contains $FF and Applesoft is active or if location $D9 is positive (less than $80) and Integer Basic is active, then a NOT DIRECT COMMAND error is issued and everything stops. Since many SPEED/ASM programs will be running under the control of Apple DOS, the SPEED/ASM INIT routine automatically stores $80 into locations $D9 and $75 for you. While this is ideal for Apple DOS users, if attempting to run your SPEED/ASM program under a different operating system (like ANIX, OS/A, or APEX) you should be aware of the fact that SPEED/ASM manipulates these two locations on power-up.

This month's demonstration program is quite simple. It writes out a sequence of numbers to a random access file and then reads them back, displaying them on the screen. Next month I will begin discussing string variables and expand this sample program into a mini database/mailing list program. ■

Listing continued.

```
0818            ; carriage return is printed first
0818            ; in order to insure that the DOS
0818            ; command is recognized.
0818            ;
0818            ;
0818 20 36 78           JSR PRINT
081B 8D 84 CF           BYT CR,CTLD,"OPEN NUMBERS,S6,D1,L8",CR,0
081E D0 C5 CE
0821 A0 CE D5
0824 CD C2 C5
0827 D2 D3 AC
082A D3 B6 AC
082D C4 B1 AC
0830 CC B8 8D
0833 00
0834            ;
0834 20 18 78           JSR FOR0
0837 63 0D 01           ADR I,1,10
083A 00 0A 00
083D            ;
083D            ;
083D            ; Emulate the BASIC statement:
083D            ;
083D            ;   PRINT CHR$(4);"WRITE NUMBERS, R";I;",B0"
083D            ;
083D 20 36 78           JSR PRINT
0840 84 D7 D2           BYT CTLD,"WRITE NUMBERS,R",0
0843 C9 D4 C5
0846 A0 CE D5
0849 CD C2 C5
084C D2 D3 AC
084F D2 00
0851 20 3C 78           JSR PRTINT
0854 63 0D              ADR I
0856 20 36 78           JSR PRINT
0859 AC C2 B0           BYT ",B0",CR,0
085C 8D 00
085E            ;
085E            ;
085E            ; Now output the value I to the textfile.
085E            ;
085E            ; This code emulates the BASIC statement:
085E            ;
085E            ;    PRINT I
085E            ;
085E            ;
085E 20 3C 78           JSR PRTINT
0861 63 0D              ADR I
0863 A9 8D              LDA #CR              ;Output a CR to finish
0865 20 00 78           JSR PUTC             ;off the line
0868            ;
0868            ;
0868            ;  Repeat for 10 numbers
0868            ;
0868 20 1B 78           JSR NEXT
086B            ;
086B            ;
086B            ; At this point a random access file
086B            ; has been created with 100 records.
086B            ; Each record contains the record
086B            ; number of that particular record.
086B            ; To insure the integrity of the
086B            ; file, it must be closed.  The following
086B            ; SPEED/ASM code emulates the BASIC
086B            ; code:
086B            ;
086B            ;   PRINT CHR$(4);"CLOSE NUMBERS"
086B            ;
086B 20 36 78           JSR PRINT
086E 84 C3 CC           BYT CTLD,"CLOSE NUMBERS",CR,0
0871 CF D3 C5
0874 A0 CE D5
0877 CD C2 C5
087A D2 D3 8D
087D 00
087E            ;
087E            ;
087E            **** End of file creation section.
087E            ;
087E            ;
087E            ;
087E            ;
087E            ************************************
087E            ;
087E            ;
087E            ; The following section re-opens
087E            ; the text file just created and
087E            ; prints the records onto the screen
087E            ; This verifies proper operation
087E            ; of the file creation process.
087E            ;
087E            ;
087E            ;
087E 20 36 78           JSR PRINT
0881 8D D6 E5           BYT CR,"Verifying Records",CR,CR,0
0884 F2 E9 E6
```

Listing continued.

Listing continued.

```
0887 F9 E9 EE
088A E7 A0 D2
088D E5 E3 EF
0890 F2 E4 F3
0893 8D 8D 00
0896          262 ;
0896          263 ;
0896 20 36 78 264          JSR PRINT
0899 8D 84 CF 265          BYT CR,CTLD,"OPEN NUMBERS,L8",CR,0
089C D0 C5 CE
089F A0 CE D5
08A2 CD C2 C5
08A5 D2 D3 AC
08A8 CC B8 8D
08AB 00
08AC          266 ;
08AC 20 18 78 267          JSR FOR0
08AF 63 0D 01 268          ADR I,1,10
08B2 00 0A 00
08B5          269 ;
08B5 20 36 78 270          JSR PRINT
08B8 84 D2 C5 271          BYT CTLD,"READ NUMBERS,R",0
08BB C1 C4 A0
08BE CE D5 CD
08C1 C2 C5 D2
08C4 D3 AC D2
08C7 00
08C8 20 3C 78 272          JSR PRTINT
08CB 63 0D    273          ADR I
08CD 20 36 78 274          JSR PRINT
08D0 AC C2 B0 275          BYT ",B0",CR,0
08D3 8D 00
08D5          276 ;
08D5 20 0F 78 277          JSR READLN         ;Force a new line to be read.
08D8 20 42 78 278          JSR RDINT
08DB 65 0D    279          ADR J
08DD          280 ;
08DD          281 ; Print the record's value.
08DD          282 ;
08DD 20 36 78 283          JSR PRINT
08E0 D2 E5 E3 284          BYT "Record number ",0
08E3 EF F2 E4
08E6 A0 EE F5
08E9 ED E2 E5
08EC F2 A0 00
08EF 20 3C 78 285          JSR PRTINT
08F2 63 0D    286          ADR I
08F4 20 36 78 287          JSR PRINT
08F7 A0 E3 EF 288          BYT " contained '",0
08FA EE F4 E1
08FD E9 EE E5
0900 E4 A0 A7
0903 00
0904 20 3C 78 289          JSR PRTINT
0907 65 0D    290          ADR J
0909 20 36 78 291          JSR PRINT
090C A7 8D 00 292          BYT "'",CR,0
090F          293 ;
090F          294 ; Move on to the next record.
090F          295 ;
090F 20 1B 78 296          JSR NEXT
0912          297 ;
0912          298 ;
0912          299 ; ** End of record verification.
0912          300 ;
0912          301 ;
0912          302 ********************************
0912          303 ;
0912          304 ; Now print the file out backwards
0912          305 ;
```

```
0912          306 ;
0912          307 ;
0912 20 36 78 308          JSR PRINT
0915 8D D2 E5 309          BYT CR,"Reading records in reverse order",CR,CR,0
0918 E1 E4 E9
091B EE E7 A0
091E F2 E5 E3
0921 EF F2 E4
0924 F3 A0 E9
0927 EE A0 F2
092A E5 F6 E5
092D F2 F3 E5
0930 A0 EF F2
0933 E4 E5 F2
0936 8D 8D 00
0939          310 ;
0939 4C 42 09 311          JMP DOFOR          ;Hide the following constants
093C          312 ;
093C FF FF    313 CM1      ADR !-1
093E 01 00    314 C1       ADR 1
0940 0A 00    315 C10      ADR 10
0942          316 ;
0942          317 ;
0942 20 15 78 318 DOFOR    JSR FOR
0945 63 0D 40 319          ADR I,C10,C1,CM1    ;Stepsize of -1
0948 09 3E 09
094B 3C 09
094D          320 ;
094D 20 36 78 321          JSR PRINT
0950 84 D2 C5 322          BYT CTLD,"READ NUMBERS,R",0
0953 C1 C4 A0
0956 CE D5 CD
0959 C2 C5 D2
095C D3 AC D2
095F 00
0960 20 3C 78 323          JSR PRTINT
0963 63 0D    324          ADR I
0965 20 36 78 325          JSR PRINT
0968 AC C2 B0 326          BYT ",B0",CR,0
096B 8D 00
096D          327 ;
096D 20 0F 78 328          JSR READLN          ;Fetch a new line
0970 20 42 78 329          JSR RDINT
0973 65 0D    330          ADR J
0975          331 ;
0975 20 36 78 332          JSR PRINT
0978 D2 E5 E3 333          BYT "Record number ",0
097B EF F2 E4
097E A0 EE F5
0981 ED E2 E5
0984 F2 A0 00
0987 20 3C 78 334          JSR PRTINT
098A 63 0D    335          ADR I
098C 20 36 78 336          JSR PRINT
098F A0 E3 EF 337          BYT " contained '",0
0992 EE F4 E1
0995 E9 EE E5
0998 E4 A0 A7
099B 00
099C 20 3C 78 338          JSR PRTINT
099F 65 0D    339          ADR J
09A1 20 36 78 340          JSR PRINT
09A4 A7 8D 00 341          BYT "'",CR,0
09A7          342 ;
09A7 20 1B 78 343          JSR NEXT
09AA          344 ;
09AA          345 ;
09AA          346 ********************************
09AA          347 ;
09AA          348 ;
```

Listing continued.

Listing continued.

```
09AA          349 ; Now get the bounds for output
09AA          350 ; from the keyboard.
09AA          351 ;
09AA 20 36 78 352         JSR PRINT        ;This turns off the DOS
09AD 8D 84 8D 353         BYT CR,CTLD,CR,0    ;READ mode.
09B0 00
09B1          354 ;
09B1 20 36 78 355 REREAD0 JSR PRINT
09B4 C9 EE F0 356         BYT "Input a lower bounds:",0 ;Get first record #
09B7 F5 F4 A0
09BA E1 A0 EC
09BD EF F7 E5
09C0 F2 A0 E2
09C3 EF F5 EE
09C6 E4 F3 BA
09C9 00
09CA          357 ;
09CA 20 0F 78 358         JSR READLN
09CD 20 42 78 359         JSR RDINT
09D0 67 0D    360         ADR FIRST
09D2 70 DD    361         BVS REREAD0
09D4          362 ; FIRST must be greater than zero
09D4          363 ; and less than or equal to 10.
09D4          364 ;
09D4 20 21 78 365         JSR IFI0
09D7 67 0D BE 366         ADR FIRST,GT,0
09DA 00 00 00
09DD D0 29    367         BTR FIRSTOK0
09DF          368 ;
09DF 20 36 78 369         JSR PRINT
09E2 8D 87 D6 370         BYT CR,BELL,"Value must be greater than zero",CR,0
09E5 E1 EC F5
09E8 E5 A0 ED
09EB F5 F3 F4
09EE A0 E2 E5
09F1 A0 E7 F2
09F4 E5 E1 F4
09F7 E5 F2 A0
09FA F4 E8 E1
09FD EE A0 FA
0A00 E5 F2 EF
0A03 8D 00
0A05 4C B1 09 371         JMP REREAD0
0A08          372 ;
0A08 20 21 78 373 FIRSTOK0 JSR IFI0
0A0B 67 0D BC 374         ADR FIRST,LE,10
0A0E BD 0A 00
0A11 D0 24    375         BTR FIRSTOK1
0A13          376 ;
0A13 20 36 78 377         JSR PRINT
0A16 8D 87 D6 378         BYT CR,BELL,"Value must be less than 11",CR,0
0A19 E1 EC F5
0A1C E5 A0 ED
0A1F F5 F3 F4
0A22 A0 E2 E5
0A25 A0 EC E5
0A28 F3 F3 A0
0A2B F4 E8 E1
0A2E EE A0 B1
0A31 B1 8D 00
0A34 4C B1 09 379         JMP REREAD0
0A37          380 ;
0A37          381 ;
0A37          382 FIRSTOK1:
0A37 20 36 78 383 REREAD1 JSR PRINT
0A3A 8D C5 EE 384         BYT CR,"Enter the final value:",0
0A3D F4 E5 F2
0A40 A0 F4 E8
0A43 E5 A0 E6
0A46 E9 EE E1
0A49 EC A0 F6
0A4C E1 EC F5
0A4F E5 BA 00
0A52          385 ;
0A52 20 0F 78 386         JSR READLN
0A55 20 42 78 387         JSR RDINT
0A58 69 0D    388         ADR ENDVAL
0A5A 70 DB    389         BVS REREAD1         ;In case of error
0A5C          390 ;
0A5C          391 ; Check to make sure that ENDVAL is
0A5C          392 ; in the range 1..10.
0A5C          393 ;
0A5C 20 21 78 394         JSR IFI0
0A5F 69 0D BE 395         ADR ENDVAL,GT,0
0A62 00 00 00
0A65 D0 30    396         BTR ENDOK0
0A67          397 ;
0A67 20 36 78 398         JSR PRINT
0A6A 8D 87 C5 399         BYT CR,BELL,"Ending value must be greater than zero",CR,0
0A6D EE E4 E9
0A70 EE E7 A0
0A73 F6 E1 EC
0A76 F5 E5 A0
0A79 ED F5 F3
0A7C F4 A0 E2
0A7F E5 A0 E7
0A82 F2 E5 E1
0A85 F4 E5 F2
0A88 A0 F4 E8
0A8B E1 EE A0
0A8E FA E5 F2
0A91 EF 8D 00
0A94 4C 37 0A 400         JMP REREAD1
0A97          401 ;
0A97 20 21 78 402 ENDOK0  JSR IFI0
0A9A 69 0D BC 403         ADR ENDVAL,LE,10
0A9D BD 0A 00
0AA0 D0 2A    404         BTR ENDOK1
0AA2          405 ;
0AA2 20 36 78 406         JSR PRINT
0AA5 8D 87 C5 407         BYT CR,BELL,"Ending value must be less than 11",0
0AA8 EE E4 E9
0AAB EE E7 A0
0AAE F6 E1 EC
0AB1 F5 E5 A0
0AB4 ED F5 F3
0AB7 F4 A0 E2
0ABA E5 A0 EC
0ABD E5 F3 F3
0AC0 A0 F4 E8
0AC3 E1 EE A0
0AC6 B1 B1 00
0AC9 4C 37 0A 408         JMP REREAD1
0ACC          409 ;
0ACC          410 ;
0ACC          411 ; Now get a stepsize value from the user.
0ACC          412 ;
0ACC          413 ENDOK1:
0ACC 20 36 78 414 REREAD2 JSR PRINT
0ACF 8D C5 EE 415         BYT CR,"Enter a stepsize value:",0
0AD2 F4 E5 F2
0AD5 A0 E1 A0
0AD8 F3 F4 E5
0ADB F0 F3 E9
0ADE FA E5 A0
0AE1 F6 E1 EC
0AE4 F5 E5 BA
0AE7 00
0AE8          416 ;
0AE8 20 0F 78 417         JSR READLN
0AEB 20 42 78 418         JSR RDINT
```

Listing continued.

mockingboard™

YA GOTTA

The most exciting things to happen to the Apple since Eve . . . MOCKINGBOARD™ the peripheral card that brings sound effects and synthesized speech to your Apple and MOCKINGBOARD™ SOUND AND SPEECH ENHANCED PROGRAMS by your favorite software companies.

Here's a sampling of some of the great programs you can HEAR with MOCKINGBOARD™.*

ZAXXON
Datasoft

RENDEZVOUS
Edu-Ware

SPACE ARK
Datamost

COVETED MIRROR
Penguin Software

ULTIMA I
California Pacific

DESECRATION
Mind Games

MICROBE
Synergistic Software

ASTEROID BELT
Superior Software

SPELLING BEE
Edu-Ware

NIGHT STRIKE
Sierra On-Line

HUMAN FLY
Datamost

MATING ZONE
Datamost

PIE MAN
Penguin Software

ROUND-ABOUT
Datamost

THUNDERBOMBS
Penguin Software

EXODUS, ULTIMA III
Origin Systems

TACTICAL ARMOR COMMAND
Avalon Hill Software

MAZE CRAZE CONSTRUCTION SET
DTI Data-Trek

BROADSIDE
to be announced

*This partial list was compiled in April to meet publishing deadlines. Many more programs supporting MOCKINGBOARD may be available at press time.

HAVE ONE

It took tremendous cooperation to make this dream come true but, when great companies support a great product, you enjoy great programs. Here are some of the great companies supporting MOCKINGBOARD™:

BRODERBUND • SIERRA ON-LINE • EDUWARE • AMI • PENGUIN • DATAMOST • MIND GAMES • AVANT-GARDE • H.A.L. LABS • SIR-TECH • DATASOFT • GYST • SUPERIOR • AVALON HILL • SAMS SOFTWARE • SYNERGISTIC • GEBELLI • BUDGECO • CALIFORNIA PACIFIC • MICRO LAB • LIGHTNING • HAYDEN • NEXA • MORNINGSIDE • DESIGNWARE • AMERICAN EDUCATIONAL • EARTHWARE • WINNER'S CIRCLE • EASI • FUNTASTIC • BANDINELLI • LEGEND • VAGABONDO • ORIGIN SYSTEMS • SILICON VALLEY SYSTEMS • ARTSCI • OH NO SOFTWARE • and more!

And it's just the beginning. See? YA GOTTA HAVE ONE!

Inspect and experience MOCKINGBOARD™ at fine computer stores everywhere. Or, if presently unavailable in your area, you can still hear what's going on risk free. Call **1-800-341-8001** and order MOCKINGBOARD™ Sound/Speech I for only $179.00. If you are not 100% satisfied, return MOCKINGBOARD™ within 30 days for a no questions asked refund.** By telephone: Amex/Mastercard/Visa. By mail: major credit card (with expiration date), money order or personal check. Please add $3.50 for shipping and handling.

mockingboard™

Sweet Micro Systems, 150 Chestnut Street, Providence, Rhode Island 02903 401-273-5333
MOCKINGBOARD™ is Apple II, Apple II-Plus and Apple IIe compatible.
Apple is a registered trademark of Apple Computer Company.
**Refund policy applies to sales made by Sweet Micro Systems and does not obligate individual retailers.

Circle 222 on Reader Service card.

Listing continued.

```
0AEE 6B 0D    419         ADR STEPSIZE
0AF0 70 DA    420         BVS REREAD2
0AF2          421  ;
0AF2          422  ; The stepsize must be checked to
0AF2          423  ; make sure that it matches the
0AF2          424  ; following conditions:
0AF2          425  ;
0AF2          426  ;  It must be in the range 1..10
0AF2          427  ;     or -1..-10.
0AF2          428  ;
0AF2          429  ;  If ENDVAL is greater than FIRST,
0AF2          430  ;  STEPSIZE must be positive.
0AF2          431  ;
0AF2          432  ;  If ENDVAL is less than FIRST,
0AF2          433  ;  STEPSIZE must be negative.
0AF2          434  ;
0AF2          435  ;
0AF2 20 21 78 436         JSR IFI0
0AF5 6B 0D BD 437         ADR STEPSIZE,EQ,0
0AF8 00 00 00
0AFB F0 26    438         BFL SSOK0
0AFD          439  ;
0AFD 20 36 78 440         JSR PRINT
0B00 8D 87 D3 441         BYT CR,BELL,"Stepsize must not equal zero",CR,0
0B03 F4 E5 F0
0B06 F3 E9 FA
0B09 E5 A0 ED
0B0C F5 F3 F4
0B0F A0 EE EF
0B12 F4 A0 E5
0B15 F1 F5 E1
0B18 EC A0 FA
0B1B E5 F2 EF
0B1E 8D 00
0B20 4C CC 0A 442         JMP REREAD2
0B23          443  ;
0B23          444  ;
0B23 20 21 78 445  SSOK0  JSR IFI0
0B26 6B 0D BE 446         ADR STEPSIZE,GT,0
0B29 00 00 00
0B2C F0 78    447         BFL SSISNEG
0B2E          448  ;
0B2E          449  ; STEPSIZE > 0, make sure that ENDVAL >= FIRST.
0B2E          450  ;
0B2E 20 1E 78 451         JSR IFI
0B31 69 0D BE 452         ADR ENDVAL,GE,FIRST
0B34 BD 67 0D
0B37 D0 23    453         BTR CHKPOSSS
0B39          454  ;
0B39 20 36 78 455         JSR PRINT
0B3C 8D 87 D3 456         BYT CR,BELL,"STEPSIZE must be negative",CR,0
0B3F D4 C5 D0
0B42 D3 C9 DA
0B45 C5 A0 ED
0B48 F5 F3 F4
0B4B A0 E2 E5
0B4E A0 EE E5
0B51 E7 E1 F4
0B54 E9 F6 E5
0B57 8D 00
0B59 4C CC 0A 457         JMP REREAD2
0B5C          458  ;
0B5C          459  ; Make sure that STEPSIZE is in the range 1..10.
0B5C          460  ;
0B5C 20 21 78 461  CHKPOSSS JSR IFI0
0B5F 6B 0D BE 462         ADR STEPSIZE,GE,1
0B62 BD 01 00
0B65 8D 6D 0D 463         STA BOOLEAN        ;Save for compound test.
0B68 20 21 78 464         JSR IFI0
0B6B 6B 0D BC 465         ADR STEPSIZE,LE,10
0B6E BD 0A 00
0B71 2D 6D 0D 466         AND BOOLEAN
0B74 F0 03    467         BFL BADSS
0B76 4C 1E 0C 468         JMP GOODSS
0B79          469  ;
0B79          470  ;
0B79 20 36 78 471  BADSS  JSR PRINT
0B7C 8D 87 D3 472         BYT CR,BELL,"STEPSIZE must be in the range 1..10",CR,0
0B7F D4 C5 D0
0B82 D3 C9 DA
0B85 C5 A0 ED
0B88 F5 F3 F4
0B8B A0 E2 E5
0B8E A0 E9 EE
0B91 A0 F4 E8
0B94 E5 A0 F2
0B97 E1 EE E7
0B9A E5 A0 B1
0B9D AE AE B1
0BA0 B0 8D 00
0BA3 4C CC 0A 473         JMP REREAD2
0BA6          474  ;
0BA6          475  ;
0BA6          476  ;
0BA6          477  ; STEPSIZE < 0, make sure that ENDVAL <= FIRST.
0BA6          478  ;
0BA6 20 1E 78 479  SSISNEG JSR IFI
0BA9 69 0D BC 480         ADR ENDVAL,LE,FIRST
0BAC BD 67 0D
0BAF D0 24    481         BTR CHKNEGSS
0BB1          482  ;
0BB1 20 36 78 483         JSR PRINT
0BB4 8D 87 D3 484         BYT CR,BELL,"STEPSIZE must be positive.",CR,0
0BB7 D4 C5 D0
0BBA D3 C9 DA
0BBD C5 A0 ED
0BC0 F5 F3 F4
0BC3 A0 E2 E5
0BC6 A0 F0 EF
0BC9 F3 E9 F4
0BCC E9 F6 E5
0BCF AE 8D 00
0BD2 4C CC 0A 485         JMP REREAD2
0BD5          486  ;
0BD5          487  ;
0BD5          488  ; Make sure that STEPSIZE is in the range -10..-1.
0BD5          489  ;
0BD5 20 21 78 490  CHKNEGSS JSR IFI0
0BD8 6B 0D BE 491         ADR STEPSIZE,GE,!-10
0BDB BD F6 FF
0BDE 8D 6D 0D 492         STA BOOLEAN        ;Save for compound test
0BE1 20 21 78 493         JSR IFI0
0BE4 6B 0D BC 494         ADR STEPSIZE,LE,!-1
0BE7 BD FF FF
0BEA 2D 6D 0D 495         AND BOOLEAN
0BED D0 2F    496         BTR GOODSS
0BEF          497  ;
0BEF 20 36 78 498         JSR PRINT
0BF2 8D 87 D3 499         BYT CR,BELL,"STEPSIZE must be in the range -1..-10",CR,0
0BF5 D4 C5 D0
0BF8 D3 C9 DA
0BFB C5 A0 ED
0BFE F5 F3 F4
0C01 A0 E2 E5
0C04 A0 E9 EE
0C07 A0 F4 E8
0C0A E5 A0 F2
0C0D E1 EE E7
0C10 E5 A0 AD
0C13 B1 AE AE
```

Listing continued.

Listing continued.

```
0C16 AD B1 B0
0C19 8D 00
0C1B 4C CC 0A    500          JMP REREAD2
0C1E             501  ;
0C1E             502  ;
0C1E             503  ; If STEPSIZE contains an appropriate value
0C1E             504  ; print the records as requested.
0C1E             505  ;
0C1E             506  ;
0C1E A9 8D       507  GOODSS  LDA #CR
0C20 20 00 78    508          JSR PUTC          ;Init for DOS.
0C23             509  ;
0C23 20 15 78    510          JSR FOR
0C26 63 0D 67    511          ADR I,FIRST,ENDVAL,STEPSIZE
0C29 0D 69 0D
0C2C 6B 0D
0C2E             512  ;
0C2E             513  ; Read the specified record
0C2E             514  ;
0C2E 20 36 78    515          JSR PRINT
0C31 84 D2 C5    516          BYT CTLD,"READ NUMBERS,R",0
0C34 C1 C4 A0
0C37 CE D5 CD
0C3A C2 C5 D2
0C3D D3 AC D2
0C40 00
0C41 20 3C 78    517          JSR PRTINT
0C44 63 0D       518          ADR I
0C46 20 36 78    519          JSR PRINT
0C49 AC C2 B0    520          BYT ",B0",CR,0
0C4C 8D 00
0C4E             521  ;
0C4E 20 0F 78    522          JSR READLN
0C51 20 42 78    523          JSR RDINT
0C54 65 0D       524          ADR J
0C56             525  ;
0C56             526  ;
0C56 20 1E 78    527          JSR IFI
0C59 63 0D BD    528          ADR I,EQ,FIRST
0C5C 00 67 0D
0C5F F0 21       529          BFL >0
0C61             530  ;
0C61 20 36 78    531          JSR PRINT
0C64 D4 E8 E5    532          BYT "The first record contains ",0
0C67 A0 E6 E9
0C6A F2 F3 F4
0C6D A0 F2 E5
0C70 E3 EF F2
0C73 E4 A0 E3
0C76 EF EE F4
0C79 E1 E9 EE
0C7C F3 A0 00
0C7F 4C 25 0D    533          JMP PRTREC
0C82             534  ;
0C82 20 1E 78    535  ^0      JSR IFI
0C85 63 0D BD    536          ADR I,EQ,ENDVAL
0C88 00 69 0D
0C8B F0 20       537          BFL >1
0C8D             538  ;
0C8D 20 36 78    539          JSR PRINT
0C90 D4 E8 E5    540          BYT "The last record contains ",0
0C93 A0 EC E1
0C96 F3 F4 A0
0C99 F2 E5 E3
0C9C EF F2 E4
0C9F A0 E3 EF
0CA2 EE F4 E1
0CA5 E9 EE F3
0CA8 A0 00
0CAA 4C 25 0D    541          JMP PRTREC
0CAD             542  ;
0CAD             543  ;
0CAD 20 21 78    544  ^1      JSR IFI0
0CB0 63 0D BD    545          ADR I,EQ,2
0CB3 00 02 00
0CB6 F0 1F       546          BFL >2
0CB8             547  ;
0CB8 20 36 78    548          JSR PRINT
0CBB D4 E8 E5    549          BYT "The 2nd record contains ",0
0CBE A0 B2 EE
0CC1 E4 A0 F2
0CC4 E5 E3 EF
0CC7 F2 E4 A0
0CCA E3 EF EE
0CCD F4 E1 E9
0CD0 EE F3 A0
0CD3 00
0CD4 4C 25 0D    550          JMP PRTREC
0CD7             551  ;
0CD7 20 21 78    552  ^2      JSR IFI0
0CDA 63 0D BD    553          ADR I,EQ,3
0CDD 00 03 00
0CE0 F0 1F       554          BFL >3
0CE2             555  ;
0CE2 20 36 78    556          JSR PRINT
0CE5 D4 E8 E5    557          BYT "The 3rd record contains ",0
0CE8 A0 B3 F2
0CEB E4 A0 F2
0CEE E5 E3 EF
0CF1 F2 E4 A0
0CF4 E3 EF EE
0CF7 F4 E1 E9
0CFA EE F3 A0
0CFD 00
0CFE 4C 25 0D    558          JMP PRTREC
0D01             559  ;
0D01             560  ;
0D01 20 36 78    561  ^3      JSR PRINT
0D04 D4 E8 E5    562          BYT "The ",0
0D07 A0 00
0D09 20 3C 78    563          JSR PRTINT
0D0C 63 0D       564          ADR I
0D0E 20 36 78    565          JSR PRINT
0D11 F4 E8 A0    566          BYT "th record contains ",0
0D14 F2 E5 E3
0D17 EF F2 E4
0D1A A0 E3 EF
0D1D EE F4 E1
0D20 E9 EE F3
0D23 A0 00
0D25             567  ;
0D25             568  ;
0D25 20 36 78    569  PRTREC  JSR PRINT
0D28 BA A0 A0    570          BYT ":  '",0
0D2B A7 00
0D2D 20 3C 78    571          JSR PRTINT
0D30 65 0D       572          ADR J
0D32 20 36 78    573          JSR PRINT
0D35 A7 8D 00    574          BYT "'",CR,0
0D38             575  ;
0D38             576  ;
0D38 20 1B 78    577          JSR NEXT
0D3B             578  ;
0D3B             579  ;
0D3B             580  ************************************
0D3B             581  ;
0D3B             582  ;
0D3B             583  ; All Done...
0D3B             584  ;
0D3B 20 36 78    585          JSR PRINT
0D3E 8D 8D 8D    586          BYT CR,CR,CR,"That's all folks!",CR,0
0D41 D4 E8 E1
0D44 F4 A7 F3
0D47 A0 E1 EC
0D4A EC A0 E6
0D4D EF EC EB
0D50 F3 A1 8D
0D53 00
0D54             587  ;
0D54             588  ;
0D54             589  ;
0D54             590  ; Close all open files.
0D54             591  ;
0D54 20 36 78    592          JSR PRINT
0D57 8D 84 C3    593          BYT CR,CTLD,"CLOSE",CR,0
0D5A CC CF D3
0D5D C5 8D 00
0D60             594  ;
0D60             595  ;
0D60 4C 59 FF    596          JMP EXIT
0D63             597  ;
0D63             598  ;
0D63             599  ; Variable declarations
0D63             600  ;
0D63             601  ;
0D63 00 00       602  I        ADR 0
0D65 00 00       603  J        ADR 0
0D67 00 00       604  FIRST    ADR 0
0D69 00 00       605  ENDVAL   ADR 0
0D6B 00 00       606  STEPSIZE ADR 0
0D6D 00          607  BOOLEAN  BYT 0
0D6E             608  ;
0D6E             609  ;
0D6E             610           END


***** END OF ASSEMBLY


]

Verifying Records


Record number 1 contained '1'
Record number 2 contained '2'
Record number 3 contained '3'
Record number 4 contained '4'
Record number 5 contained '5'
Record number 6 contained '6'
Record number 7 contained '7'
Record number 8 contained '8'
Record number 9 contained '9'
Record number 10 contained '10'

Reading records in reverse order

Record number 10 contained '10'
Record number 9 contained '9'
Record number 8 contained '8'
Record number 7 contained '7'
Record number 6 contained '6'
Record number 5 contained '5'
Record number 4 contained '4'
Record number 3 contained '3'
Record number 2 contained '2'
Record number 1 contained '1'

Input a lower bounds:1

Enter the final value:7
```

Listing continued.

Listing continued.

```
Enter a stepsize value:2

The first record contains :  '1'
The 3rd record contains :  '3'
The 5th record contains :  '5'
The last record contains :  '7'


That's all folks!

Verifying Records


Record number 1 contained '1'
Record number 2 contained '2'
Record number 3 contained '3'
Record number 4 contained '4'
Record number 5 contained '5'
Record number 6 contained '6'
Record number 7 contained '7'
Record number 8 contained '8'
Record number 9 contained '9'
Record number 10 contained '10'

Reading records in reverse order

Record number 10 contained '10'
Record number 9 contained '9'
Record number 8 contained '8'
Record number 7 contained '7'
Record number 6 contained '6'
Record number 5 contained '5'
Record number 4 contained '4'
Record number 3 contained '3'
Record number 2 contained '2'
Record number 1 contained '1'

Input a lower bounds:9

Enter the final value:3

Enter a stepsize value:-2

The first record contains :  '9'
The 7th record contains :  '7'
The 5th record contains :  '5'
The last record contains :  '3'


That's all folks!

Verifying Records


Record number 1 contained '1'
Record number 2 contained '2'
Record number 3 contained '3'
Record number 4 contained '4'
Record number 5 contained '5'
Record number 6 contained '6'
Record number 7 contained '7'
Record number 8 contained '8'
Record number 9 contained '9'
Record number 10 contained '10'

Reading records in reverse order

Record number 10 contained '10'
Record number 9 contained '9'
Record number 8 contained '8'
Record number 7 contained '7'
Record number 6 contained '6'
Record number 5 contained '5'
Record number 4 contained '4'
Record number 3 contained '3'
Record number 2 contained '2'
Record number 1 contained '1'

Input a lower bounds:0

Value must be greater than zero
Input a lower bounds:11

Value must be less than 11
Input a lower bounds:1

Enter the final value:0

Ending value must be greater than zero

Enter the final value:12

Ending value must be less than 11
Enter the final value:4

Enter a stepsize value:-3

STEPSIZE must be positive.

Enter a stepsize value:0

Stepsize must not equal zero

Enter a stepsize value:4

The first record contains :  '1'


That's all folks!
```

Listing 7. SPEED/ASM Equates.

```
0800          TTL "Listing  7 : SPEED/ASM Equates"
0800     ;
0800     ;
0800     ;
0800     ***********************************
0800     *                                 *
0800     *        SPEED/ASM Equates        *
0800     *                                 *
0800     ***********************************
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ; GENERAL PURPOSE EQUATES
0800     ;
0800     ; The following variables are used
0800     ; by the SPEED/ASM package and
0800     ; shouldn't be used by the SPEED/ASM
0800     ; programmer.
0800     ;
0800     ;
0800     ;
0800     ;
0000     FORASAV   EPZ 0
0001     FORXSAV   EPZ FORASAV+1
0002     FORYSAV   EPZ FORXSAV+1
0003     FORZPG    EPZ FORYSAV+1
0005     DESTADR   EPZ FORZPG+2
0007     PTRADR    EPZ DESTADR+2
0009     ISIMMED   EPZ PTRADR+2
000A     OP        EPZ ISIMMED+1
000C     MAXLEN    EPZ OP+2
000D     VALUE     EPZ MAXLEN+1
000F     DIGIT     EPZ VALUE+2
0010     LEAD0     EPZ DIGIT+1
0011     JMPADR    EPZ LEAD0+1
0013     COUNT     EPZ JMPADR+2
0014     GOTLN     EPZ COUNT+1
0015     LINEINDX  EPZ GOTLN+1
0016     SIGN      EPZ LINEINDX+1
0017     ACL       EPZ SIGN+1
0018     ACH       EPZ ACL+1
0019     XTNDL     EPZ ACH+1
001A     XTNDH     EPZ XTNDL+1
001B     AUXL      EPZ XTNDH+1
001C     AUXH      EPZ AUXL+1
0800     ;
0033     PROMPT    EPZ $33
004E     RNDL      EPZ $4E
004F     RNDH      EPZ $4F
0100     STACK     EQU $100
0200     INPUT     EQU $200
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0800     *************
0800     * CONSTANTS *
0800     *************
0800     ;
0800     ;
0800     ;
0800     ; The following symbols are constants
0800     ; for the values "FALSE", "TRUE", and
0800     ; Carriage Return (respectively).
0800     ;
0800     ; These symbols should only appear
0800     ; as immediate operands to a 6502
0800     ; instruction or in the operand field
0800     ; of a pseudo-opcode like BYT.
0800     ;
0800     ;
0800     ;
0800     ;
0800     ;
0000     FALSE     EQU 0
0001     TRUE      EQU 1
008D     CR        EQU $8D
0800     ;
0800     ;
0800     ;
0800     ;
```

Listing continued.

Listing continued.

```
0800         ;
0800         ;
0800         ; "IF" STATEMENT EQUATES
0800         ;
0800         ; The following symbols should only
0800         ; be used in the ADR pseudo-opcode
0800         ; following a call to the SPEED/ASM
0800         ; IFx routines.
0800         ;
0800         ;
0800         ;
00BD         EQ        EQU "="
00A3         NE        EQU "#"
00BE         GT        EQU ">"
00BC         LT        EQU "<"
BDBE         GE        EQU ">"|"="*256
BDBC         LE        EQU "<"|"="*256
0800         ;
0800         ;
0800         ;
0800         ;
0800         ;
0800         ;
0800         ;
0800         **************************
0800         * SPEED/ASM ENTRY POINTS *
0800         **************************
0800         ;
0800         ;
0800         ;
0800         ;
0800         ; NOTE: THE EQUATE OF PUTC MUST
0800         ; BE CHANGED IF YOU RELOCATE
0800         ; SPEED/ASM TO SOME LOCATION
0800         ; OTHER THAN $7800
0800         ;
0800         ;
0800         ;
7800         PUTC      EQU $7800
7803         GETC      EQU PUTC+3
7806         SAGL      EQU GETC+3        ;FOR USE BY S/A ONLY- SEE DOC.
7809         SAPC      EQU SAGL+3        ; "   "   "  "    "     "  "
780C         HOME      EQU SAPC+3        ;HOME AND CLEAR
780F         READLN    EQU HOME+3
7812         INIT      EQU READLN+3
7815         FOR       EQU INIT+3
7818         FOR0      EQU FOR+3
781B         NEXT      EQU FOR0+3
781E         IFI       EQU NEXT+3
7821         IFI0      EQU IFI+3
7824         IFS       EQU IFI0+3
7827         IFS0      EQU IFS+3
782A         MOVE      EQU IFS0+3
782D         LOAD      EQU MOVE+3
7830         MOVS      EQU LOAD+3
7833         LDSTR     EQU MOVS+3
7836         PRINT     EQU LDSTR+3
7839         PRTSTR    EQU PRINT+3
783C         PRTINT    EQU PRTSTR+3
783F         RDSTR     EQU PRTINT+3
7842         RDINT     EQU RDSTR+3
7845         ONXGOTO   EQU RDINT+3
7848         CASE      EQU ONXGOTO+3
784B         CASEI     EQU CASE+3
784E         INSET     EQU CASEI+3
7851         NOTINSET  EQU INSET+3
7854         ABS       EQU NOTINSET+3
7857         NEG       EQU ABS+3
785A         MUL       EQU NEG+3
785D         DIV       EQU MUL+3
7860         MOD       EQU DIV+3
7863         RND       EQU MOD+3
7866         SUBSTR    EQU RND+3
7869         INDEX     EQU SUBSTR+3
786C         LENGTH    EQU INDEX+3
786F         CONCAT    EQU LENGTH+3
7872         GETWZPG   EQU CONCAT+3      ;USED BY SPEED/ASM
7875         RDFP      EQU GETWZPG+3
7878         PRTE      EQU RDFP+3
787B         PRTF      EQU PRTE+3
787E         FADD      EQU PRTF+3
7881         FSUB      EQU FADD+3
7884         FMUL      EQU FSUB+3
7887         FDIV      EQU FMUL+3
788A         FLT       EQU FDIV+3
788D         FIX       EQU FLT+3
7890         FNEG      EQU FIX+3
7893         FADDTN    EQU FNEG+3
7896         FSUBTN    EQU FADDTN+3
7899         FTIMES    EQU FSUBTN+3
789C         FINTO     EQU FTIMES+3
789F         IFF       EQU FINTO+3
78A2         MOVFP     EQU IFF+3
0800         ;
0800         ;
0800         ;
0800                   END

***** END OF ASSEMBLY
```

by Gregory R. Glau

Goof-Proof Your Programs

Oh, you've seen 'em and I have too. They're your secretaries and bookkeepers and the guy who takes care of your inventory.

We've all seen them sitting in front of your Apple, staring at its screen, trying to interpret what it says.

And you can almost hear them thinking: "What in the world do I do now?"

Sure, you explained the programs to them. And you did a good job, too; after all, you wrote them. You went over all the details and menu selections and they told you they understood how to run the things, but now—more often than they should—your people seem to end up staring at your Apple's video screen, frustrated because they can't tell what to do next.

But maybe it really isn't their problem—perhaps there *are* a few areas in the design of the programs that might cause difficulties. Maybe the best way to say it is there are some specific guidelines that will make your programs better, easier to use, more efficient in their operation, and "goof-proof" to anyone who uses them.

Now those commercial systems you use are probably pretty good this way. That's part of what you pay for. But when we write programs ourselves, we understand how they work and what they're supposed to do, but sometimes we just don't consider that others will one day use them. And they'll need more help than we do.

The place to begin is even before you start to code your program. Flowcharting—making a visual map of where you want a program to go and what you want it to do—is out of style right now. Where once it seemed you had to start with a flowchart before you sat down to write a program, too many of us simply plunk ourselves at our Apple and start typing away.

That's probably not a major difficulty for those of you who are outstanding programmers. But I know when I do that—write before I think—it causes all sorts of problems down the road both in terms of program organization and in how well the program works.

And it's doubly true if others will one day use those programs.

So even if you don't want to buy one of those cute plastic flowchart rulers with all its neat holes and scales, at least try to plan a program before you start to code it.

Just a list of what you want a program to do, along with a general outline of where in the system you'll do certain things, will be a big help. You should also have a list of your variables here; it's helpful. You might call this a pseudo-flowchart. Figure 1 is an example.

Once you have a flowchart in mind, its listing will give you some idea of what type of routines you'll want to use.

```
The purpose of this program is to maintain a list of about
1000 inventory items, along with the ability to produce
the reports indicated.

1. DIMension any arrays and set any variables that we
   need ( Lines 50 - 99 )

2. Program menu ( Lines 100 - 200 )

3. Create a file for the data ( Lines 1000 - 1900 )

4. Add items to the inventory ( Lines 2000 - 2900 )

                 / /

10. Change any/all data for an item
    ( Lines 10000 - 10900 )

11. Print a reorder summary ( Lines 11000 - 11900 )

12. Print a price list ( Lines 12000 - 12900)

Also, list your variables here:

N$.........................Item number

R..........................Retail price

C..........................Cost price

N..........................The number in stock right now

And so on...
```

Figure 1. Pseudo-flowchart of a simple inventory program.

Address correspondence to Gregory R. Glau, PO Box 1627, Prescott, AZ 86302.

```
1000 PRINT "ANSWER 'Y' FOR YES, 'N' FOR NO:"
1010 PRINT
1020 PRINT "DO YOU WANT TO ADD MORE ITEMS"
1030 INPUT "TO THE INVENTORY FILES "; A$
1040 IF A$="Y" THEN 2000
1050 IF A$="N" THEN 3000
2000 HOME: REM THIS SECTION PRINTS A PRICE LIST...
```

Figure 2. An example of a problem in entry input.

```
1000 PRINT "ANSWER 'Y' FOR YES, 'N' FOR NO:"
1010 PRINT
1020 PRINT "DO YOU WANT TO ADD MORE ITEMS:"
1030 INPUT" TO THE INVENTORY FILES ";A$
1040 IF A$="Y" THEN 2000
1050 IF A$="N" THEN 3000
1060 GOTO 1000
2000 HOME: REM THIS SECTION PRINTS A PRICE LIST...
```

Figure 3. Corrected data entry sequence.

Generally, the more GOSUBs you use in a program, the better it is. This means you can access about any part of the program with a GOSUB command and use the code over and over.

For instance, in an inventory program you'd want to display the inventory items on your screen, to determine whether you want to change them, correct prices, and so on. But you'll also probably want to print much the same type of information.

If you put the items in your display section into a format that you can also use when you print them, then it's a simple matter to use the same subroutine(s) to display and also to print the information. You just have to indicate to your Apple if you wish to print to the screen or to your printer. You write the code only once.

You'll also notice the pseudo-flowchart in Figure 1 is written in blocks; you map a section of the program for each function. You only need to know the approximate number of program lines to save for a particular task, and with a little experience you'll have a good idea. Better to leave too many than too few.

It's also helpful to have all of your dimensioning in one spot, the menus in one area, the printing off by itself, and so on. You always know what is where.

A good rule to remember in any menu a user might see is to give them an exit. Few things are more frustrating and confusing than to confront five or six possible choices of action, to understand none of them, and yet have no way to escape the situation.

So always have a menu selection that allows an exit from that particular part of the program. You may want to have the user return to a former level in the system, or pull up a HELP screen, or just to stop. But always allow an exit.

On that same line, why not incorporate HELP messages, with access to them from any menu in the program? More and more programs come with them now, so why shouldn't the businessperson/programmer also use this valuable tool?

Just put a menu selection in every menu inside your programs that allows a user to request a HELP screen. Many will be similar, but all that's required is a basic explanation of what's going on in the program at this point, as well as what's expected from the user. And what better place is there to put your program instructions, anyway, than right in the program itself?

It's also helpful to use single-digit menu selections. Why have the user type in five or six (or even two) characters where one will suffice?

Along this same line, always check the entry data for correctness.

Now there's no way for a programmer to determine if a specific entry amount is right or wrong, but you can design in parameters that won't allow a user to enter data that simply *can't* be right.

For instance, in that inventory program your part numbers may run from 1 to 1000. You'd want a line in your program to reject any number less than 0 (<0), and another line to reject any number larger than 1000 (>1000). Nothing complex, and you can just tell the program to return to the line that asks for the entry until the user enters a proper figure.

You also want to check the Yes/No responses inside your programs, as much as possible, for correct input. For instance, you may have a menu that allows five selections. You'd want lines following it to reject any number or selection that is not on the menu.

Along this same line, I like to use 1 or 2 to indicate Yes or No in our programs, rather than Y or N. This is because a line in a program that asks for the input of a number will keep saying REENTER if you just press return, while a line that asks for a string variable (like Y or N) will accept a return and simply move on.

Take a look at Figure 2. What do you think happens to the program if you just press return instead of answering Y or N?

```
****************************************************************
*                  THE COURTNEY COLLECTION                     *
*                     FINE NEEDLEWORK                          *
*                   INVENTORY FILE MENU                        *
*                      JULY 9, 1983                            *
*      *** SECTION 3.B OF THE INVENTORY PROGRAM ***            *
*                                                              *
*    PLEASE ENTER THE MENU SELECTION YOU WISH TO USE:          *
*                                                              *
*** ENT *  ENTER NEW ITEMS AS THEY'RE RECEIVED.                *
*                                                              *
*** REM *  REMOVE ITEMS FROM STOCK AS THEY'RE USED             *
*                                                              *
*** PNT *  PRINT A PRICE LIST                                  *
*                                                              *
*** DIS *  DISPLAY THE VALUE OF ALL ITEMS IN INVENTORY         *
*          AS OF TODAY'S DATE, EXCLUDING RETURNS               *
*                                                              *
*** PSUM   PRINT A SUMMARY OF ALL ITEMS WE'VE RUN OUT OF       *
*          SINCE WE LAST PRINTED THIS SUMMARY                  *
*                                                              *
*** CHA *  CHANGE WITH DIRECT ACCESS THE ITEMS                 *
*                                                              *
****************************************************************
```

Figure 4. A cluttered and unclear menu display.

```
INVENTORY UPDATE PROGRAM

PLEASE SELECT --->
------------------------------------------------------------
1. ADD NEW ITEMS

2. DEDUCT ITEMS

3. CORRECT / CHANGE AN ENTRY

4. DISPLAY THE VALUE OF ALL ITEMS IN-STOCK

5. PRINT A REORDER SUMMARY

6. PRINT A PRICE LIST

7. LOOK AT A HELP SCREEN DISPLAY

8. STOP NOW
------------------------------------------------------------

ENTER YOUR SELECTION HERE --->....
```

Figure 5. The inventory menu as it should be.

The program will move to line 1500, as that's the next logical place for it to go. Someone in a hurry may intend to press Y or N but just hit return instead and find him/herself in a part of the program where she/he didn't really want to be. You can fix it, of course, by adding in a line (1060) that tells the program to ask again if an incorrect response was made, as shown in Figure 3.

It's also a good idea to start your programs at line 50 or higher. Too often I've run into this problem: I have, say, ten menu selections on the main menu. The user is asked to pick one, with #10 as a stop (10. STOP NOW).

Well, when I've dimensioned things or put GOSUBs in low-numbered lines, I often find a user will press 10 to stop for a moment (to change disks, get some coffee or whatever) and then when he starts over, he forgets to type RUN.

Yes, you guessed it. Perhaps he wanted to do the first menu selection (remember the menu will still display on the screen), so he enters the number 1 and presses return. This eliminates line 1 from the program.

So. . . if he realizes what hap-

1. Use a flowchart or a list, a pseudo-flowchart, to outline what you want to do and where in the program you want things to happen. List your variables here; while you may not know them all to start with, you'll at least have a starting point.
2. Use as many subroutines as possible; this gives you a lot of flexibility and makes the program work faster and better.
3. Write each program section in blocks.
4. Give each part of the program (menus, printing, etc.) its own area in your program.
5. Always give the user an exit from any part of the program.
6. Let the user request HELP screens to tell him what's happening inside the program.
7. Make each menu selection a single-digit item.
8. Check data input within value ranges, if possible. If you have no inventory items with a part number higher than 2755, don't let the user *enter* a larger number.
9. Check all menu selections as they're entered. If you have only six selections, don't allow the user to enter #7 or # – 12 and so on.
10. Check Yes/No responses to make sure the program doesn't skip somewhere it should not. Error-trap all responses.
11. Start your program lines with *higher* numbers than your menus use.
12. Remember the K.I.S.S. principle. Keep your menus easy to read and understand. Keep your displays as clean as you can. Put nothing on the screen but what that particular program section requires.

Figure 6. A dozen tips to make your program more efficient.

pened, he must reload the program and start over. Worse, if he doesn't understand what he's done and just starts over with a RUN, since line 1 isn't present any longer in the program structure, all sorts of neat things can happen. If line 1 dimensioned something vital, that function will no longer work. If line 1 sent the user on a GOSUB or GOTO to a HELP screen or an instruction section, he'll no longer see that information.

The answer is really simple—keep the line numbers that start your program *higher* than your menu selection figures.

Finally, keep in mind the old K.I.S.S. rule (Keep It Simple, Stupid) as it applies to what the user sees when he runs your programs. Any instructions you display should be easy to understand and read. Write in short paragraphs. Say exactly what you mean. Show your screen instructions to someone who *won't* use your programs. If they can understand them, you're in good shape.

The same rule applies to your screen displays. Keep them neat and clean. Any information that isn't required for a particular part of a program doesn't belong on the screen when you're using that section of the system.

Again with that inventory program in mind, Figure 4 shows an example of a poor display, while Figure 5 presents all the same information *without* all the hoopla.

Detailed in Figure 6 are a dozen little tips that will make your programs easier to use, easier to understand, and simply make them work more efficiently. They'll also goof-proof your programs. Isn't that the whole idea? ■

A SPEECH SYNTHESIZER ON A DISK!

That's why Apples and Ataris are saying: "Talk Is Cheap"

IT'S CALLED THE SOFTWARE AUTOMATIC MOUTH, S.A.M. FOR SHORT

It's a high quality speech synthesizer created entirely in software. You use it as a software utility, load it into RAM, and then use your machine as usual, except now you can make your programs talk. It generates the speech sounds on demand, so there is no limit to what it can say.

When you hear S.A.M., you'll probably agree that it sounds better than all the hardware speech synthesizers for Apple or Atari computers. And, it has a truly remarkable price.

YOU CONTROL INFLECTION, PITCH AND SPEED

With its user-variable inflection, S.A.M. can accent words on the right syllable and emphasize the important words in a sentence.

You can also make S.A.M.'s speech higher or lower, and faster or slower, over a wide range of settings.

USE EASY PHONETIC INPUT OR PLAIN ENGLISH TEXT

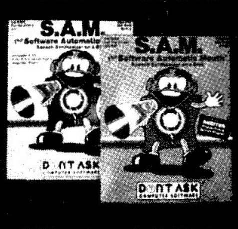
S.A.M. understands a simple phonetic spelling system, not a mysterious alpha-numeric code. S.A.M. helps you learn phonetic spelling by showing you your mistakes, and the owner's manual gets you started with an English-to-phonetics dictionary of 1500 words. So it's easy to make S.A.M. produce exactly the sounds you want.

But suppose you want to type ordinary English, or you want your machine to read a word processor file aloud. The S.A.M. disk comes with RECITER, an English text-to-speech conversion program that lets S.A.M. speak from plain English text.

ADD SPEECH TO YOUR PROGRAMS WITH EASE

In a BASIC program, you add speech with just a couple of commands. In a machine language program, it's just as easy. S.A.M. comes with four demonstration programs to show off its distinctive features and help to write your own talking programs. Write adventure games with talking characters, educational programs that explain aloud, or utilities with spoken prompts — put your imagination to work.

S.A.M.



S.A.M. programmed by Mark Barton

You can order S.A.M. directly from DON'T ASK. Add $2.00 for shipping and handling to your check or money order (or order C.O.D.)

S.A.M. for the Apple II/II+

Includes an 8-bit digital-to-analog converter and audio amplifier on a board. Only **$124.95**

Requires 48K, disk. (S.A.M. takes up to 9K; RECITER 6K.) You will also need a speaker.

NEW: S.A.M. HAS MANY VOICES

We've added two software KNOBS that you adjust (with a POKE) to change the shape of S.A.M.'s "mouth" and "throat". These KNOBS make S.A.M. sound like a different person — an old lady, a little kid, an extraterrestrial. You create S.A.M.'s vocal personalities.

NEW FOR APPLE USERS:

S.A.M. for the Apple II, II+, or //e can load into a 16K RAM card, leaving more memory for talking programs.

DON'T ASK INC
COMPUTER SOFTWARE

2265 Westwood Boulevard, Suite B-150
Los Angeles, California 90064
Telephone (213) 397-8811

Hear S.A.M. at your favorite computer store today!
Dealer inquiries welcome.

Circle 173 on Reader Service card.

Fudge It!

by Don Fudge

Shape Sequence Animation

Last month I began a discussion of animation by describing how to effect scrolling on the screen. That launches us into a look at shape sequence animation.

With vector shapes you can use whatever shape table numbers you want, in whatever order you want, and any number of shapes can occur in a sequence. For example, suppose you wanted to make a stick-figure man "walk." You might have a sequence of 4, 7, 6, 8, 1, 9, 3, 5, 2, 7 for shape numbers of your sequence shapes. That's ten sequence-shape numbers, with repeating allowed. Shape numbers are referring to Applesoft shape numbers, which get numbers because of their shape table index. You needn't use shapes' numbers from assembly, but in Basic it's the only convenient way to DRAW or XDRAW. It might be more convenient to have your shape table numbers be the same as the shape sequence numbers. One thing that makes this not particularly important is the fact that you'll often use specific shapes more than once in a sequence.

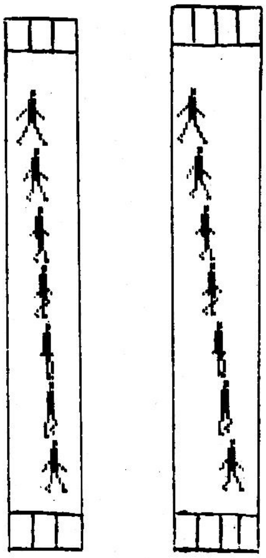


Figure 1. Printout of block shape sequences.

Walking

Think of walking. There are a couple of times in a walking sequence when, viewed from the side, one specific shape could represent more than one specific aspect of the sequence. There's no sin in using the number 4 shape twice, for example. So an algorithm to have a stick figure walk will be constructed like so:

1) Erase, by XDRAW, the shape at old coordinates (OX,OY).
2) Draw, by XDRAW, the shape at new coordinates (X,Y).
3) Dump the new coordinates into the old coordinates (OX=X:OY=Y).
4) Calculate the new coordinates using a step value, X=X+STEP. If the figure is moving vertically as well as horizontally, such as walking upstairs, also do Y=Y+STEP.
5) Go back to 1.

Remember that if you're doing page flipping things will be more complex and you'll be drawing on one screen while displaying the next. The fundamentals of this method were covered in my April column. Page flipping is a way to stop showing the drawing *process* and begin showing the drawing *results* only. The effect of this is to smooth things out and make the animation not look flickery.

Which Screen, Which Shape, Draw and Erase Chart

	screen on which to erase/draw	screen displayed	shape # erased	shape # drawn	HL of shape erased	HL of shape drawn
	1	2	1	3	0	0
	2	1	2	4	0	0
	1	2	3	5	0	0
	2	1	4	6	0	0
	1	2	5	7	0	0
•	2	1	6	1	0	1
••	1	2	7	2	0	1
	2	1	1	3	1	1
	1	2	2	4	1	1
	2	1	3	5	1	1
	1	2	4	6	1	1
	2	1	5	7	1	1
•	1	2	6	1	1	2
••	2	1	7	2	1	2
	1	2	1	3	2	2
	↓	↓	↓	↓	↓	↓
	2	1	5	7	34	34
•	1	2	6	1	34	0
••	2	1	7	2	34	0
	1	2	1	3	0	0

Table. Which Screen, Which Shape, Draw and Erase chart.

Address correspondence to Don Fudge at Avant-Garde Creations, PO Box 30160, Eugene, OR 97403.

INTRA's PSIO

Programmable Serial I/O Board

AND SOFTERM

Turn your APPLE into an intelligent file transfer terminal for accessing applications on hosts and time sharing systems. **Supporting keyboard selectable transfer rates to 9.6 kbit/s,** the ***PROGRAMMABLE SERIAL I/O BOARD,** when used with Softronics Inc.'s **Softerm** lets your APPLE emulate IBM's 3101, DEC's VT100, DG's D-200, LSI's ADM-3A and ADM-5A, Hazeltine 1400 and 1500, ADDS Regent, and TeleVideo 900 Series Terminals.

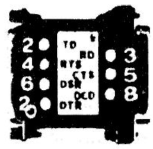
& ASCII EXPRESS

Give your **APPLE** the most versatile and accurate data transfer capabilities available. Combining the **PSIO's** programmable hardware options with the **Editor, Directory,** and **Keyboard Macro** features of **SDS's ASCII EXPRESS: PROFESSIONAL,** there is virtually **NO** Computer System it can't be used with. This is the only hardware/software combination available to offer **5-LEVEL BAUDOT CODE; CURRENT-LOOP** capability; and all desireable **RS-232** and **ASCII UTILITIES for APPLE COMMUNICATIONS.**

PSIO Board	199.95
SOFTERM Software	150.00
ASCII EXPRESS: Professional	129.95
300 Baud 103LP Modem	245.00
1200 Baud 202LP Modem	295.00
1200 Baud 212LP Modem	495.00

Special
10% Off
when you buy 2 or more items

LINE-POWERED
RS-232 LINE MONITOR
$40

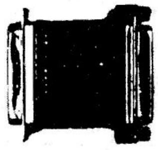


SEX CHANGE $28

COMPACT
MALE-MALE OR FEMALE-FEMALE SEX CHANGER
$28

ASYNCHRONOUS NULL - MODEM
$28
P.c. Design

DB-25 CABLE CONFIGURATOR
$60
1-year warranty

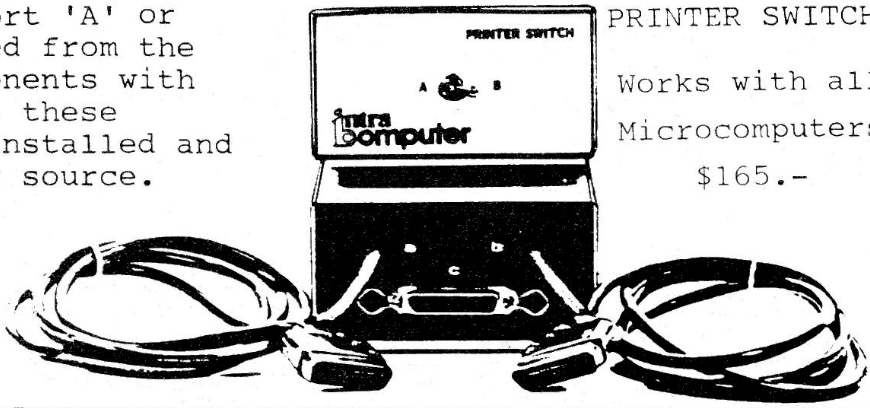


AB SWITCHES   One year warranty.

Intra Computer offers a complete family of manual line switching devices to enable Port 'C' connectorized signals to be directed to either Port 'A' or Port 'B'. Constructed from the highest quality components with gold plated contacts, these products are easily installed and require no A.C. power source.

come to INTRA for DATA·COMM

CENTRONICS INTERFACE COMPATIBLE PRINTER SWITCH



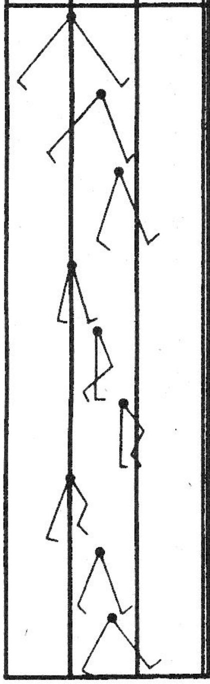
Works with all Microcomputers

$165.-

intra computer
101 W. 31st, N.Y., N.Y. 10001
(212) 947-5533

MANA

3-byte wide block shape

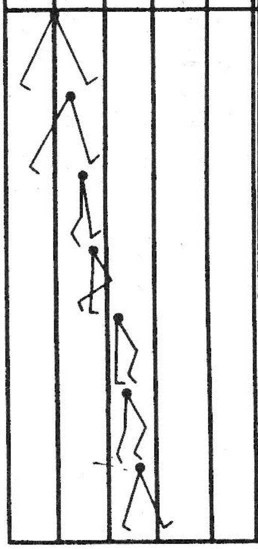


1st byte	2nd byte	3rd byte	sequence #	shape # from MAN	centered on this hor. coord.	VT,VB	move HL and HR up by 1?	hor. coords of block-shape boundaries	MANC seq. #
			1	1	35	19,40	yes	28-49	-
			2	6	38	19,40	no	28-49	1
			3	2	40	19,40	no	28-49	2+7
			4	7	42	19,40	yes	35-56	3
			5	3	45	19,40	no	35-56	4
			6	8	47	19,40	no	35-56	5
			7	4	49	19,40	yes	42-63	6
			8	9	52	19,40	no	42-63	-
			9	10	54	19,40	no	42-63	-

Figure 2a. MANA. Incrementing the horizontal byte column (X) coordinate by the step value three times per sequence.

MANC

5-byte wide block shape



1st byte	2nd byte	3rd byte	4th byte	5th byte	seq. #	shape #	shape centered on hor coord	VT,VB	move HL and HR up by 2?	hor coords of block-shape boundaries
					1	6	35	0,21	yes	28-63
					2	2	37	0,21	no	28-63
					3	7	39	0,21	no	28-63
					4	3	41	0,21	no	28-63
					5	8	43	0,21	no	28-63
					6	4	45	0,21	no	28-63
					7	2	47	0,21	no	28-63

Figure 2b. MANC. Adding the step value to the horizontal byte column (X) coordinate when the sequence is finished.

This is a good place to discuss block shape sequences. With block shapes, it's not just a matter of drawing proper shape sequences in the proper places at the proper times and incrementing by a constant step value for the next coordinate. It's true that you can place vector shapes anywhere on the screen at any time, with illegal positions at X<0,X>279,Y<0 and Y>191 (so use these for parameter checking). But block shapes cannot be handled likewise.

With block shapes you must stay within Y = 0 and Y = 191 and also X byte column (horizontal offset) 0 and 39. And you can't move less than 1 byte horizontally if you have only one shape, unless you want to use relatively slow *shift animation*. See *Hi-Res Secrets* for details on that. So, you'll almost always be using what's known as *pre-shifted shapes*, in sequences of seven.

Pre-shifted shape sequences are block shape sequences that allow less-than-7-dot (1 visible byte) moves horizontally. For similar graphics objects, such as seven identical flying saucers, pre-shifted shapes are a simple matter of running an automatic sequence creator (Listing 1) on the first flying saucer and saving the resultant seven-shape sequence as a table. Take a look at Figures 1, 2a and 2b.

In Figure 1 we see a step 1 (per move), seven-shape block shape sequence that is 3 bytes wide, and a step 2, seven-shape block shape sequence that is 4 bytes wide. Consider the left and right boundaries of these shape blocks to be the actual block shape boundaries. Notice how throughout the running of a shape sequence, neither the X coordinate nor the Y coordinate changes one iota. It is only when the sequence is finished that we add the step value to the horizontal byte-column coordinate. This is illustrated in Figure 2b. In Figure 2a, however, the X coordinate is incremented three times per sequence. In both diagrams, HR means horizontal right coordinate, HL means horizontal left coordinate, VT means vertical top coordinate, and VB means vertical bottom coordinate:

```
        VT
     *******
HL*******HR
     *******
     *******
        VB
```

Again, block shapes have only 40 possible X coordinates per screen, not 280 like vector shapes, because block shapes use byte-column coordinates, not regular X coordinates, in the horizontal direction.

Block Shape Sequences

In Figure 1, shapes 1–10 were extracted from a vector shape table (MAN) to create the nine shapes in MANA's block shape sequence table, which was updated three times per sequence in a very non-standard way. But from MANA was created MANC, a standard seven-shape ever-incrementing sequence of block shapes (Figure 2a). All it took was

Listing 1. Sequence Creator.

```
0  ONERR  GOTO 63990
1  PRINT  CHR$ (4);"BLOADTEST H (CALL2186)": GOSUB 2500: GOTO 600
2  HOME : INPUT "SHAPE TABLE NAME: ";STN$: IF  LEN (STN$) = 0 THEN 600
4 D$ =  CHR$ (4): PRINT D$"BLOAD";STN$
5  HOME : VTAB 21: INPUT "SHAPE #: ";SHN: POKE 7,SHN
15  POKE  - 16304,0: POKE  - 16297,0
18 VS = 1:BS = 0
20  INPUT "VTOP:";VT: INPUT "VBOT:";VB: INPUT "HRIGHT:";HR: INPUT "HLEFT:"
    ;HL
30  POKE 252,VT: POKE 253,VB: POKE 254,HR: POKE 255,HL
42  CALL 2116
43  HOME : VTAB 21: INPUT "DO YOU WANT ANOTHER SHAPE? (Y/N):";QW$: IF  LEN
    (QW$) = 0 THEN 43
44  IF  ASC (QW$) <  > 89 THEN  HOME : VTAB 21: GOSUB 63000: GOTO 600
45  GOTO 5
47  POKE  - 16303,0: POKE  - 16298,0: HOME : VTAB 1: PRINT "USE THE PADDLE
    S TO MOVE THE DOT TO THE  UPPER LEFT RECTANGLE POINT. HIT PDL 0   BUT
    TON. THEN MOVE THE DOT TO THE LOWER  RIGHT RECTANGLE POINT. HIT PDL 1
     BUTTON.": GOSUB 63000
48  POKE 232,248: POKE 233,8: SCALE= 1: ROT= 64
49  POKE  - 16304,0: POKE  - 16297,0
50  HOME :P1 =  PDL (1): IF P1 > 150 THEN 50
55 P0 =  PDL (0): XDRAW 1 AT P0,P1:X% =.P0:Y% = P1
60 P1 =  PDL (1): IF P1 > 150 THEN 60
65  FOR QW = 1 TO 200: NEXT : HOME : VTAB 21: PRINT "X: "P0: PRINT "Y: "P1

70 P0 =  PDL (0): XDRAW 1 AT X%,Y%: XDRAW 1 AT P0,P1:X% = P0:Y% = P1
80 B0 =  PEEK ( - 16287): IF B0 > 127 AND FL = 0 THEN FL = 1: GOTO 100
85 B1 =  PEEK ( - 16286): IF B1 > 127 AND SG = 0 THEN SG = 1: GOTO 110
90  GOTO 60
100 VT = P1:HL =  INT (P0 / 7): PRINT  CHR$ (7): IF SG = 1 THEN 120
105  GOTO 60
110 VB = P1:HR =  INT (P0 / 7): PRINT  CHR$ (7): IF FL = 1 THEN 120
115  GOTO 60
120  HOME : VTAB 21: PRINT "HOR.--FROM:"HL" TO "HR"---WIDTH:"(HR - HL)
125  XDRAW 1 AT P0,P1
130  PRINT "VER.--FROM:"VT" TO "VB"---HEIGHT:"(VB - VT): VTAB 23: PRINT "J
     OT THIS DOWN! (HIT ANY KEY TO CONT.)": GOSUB 63010
150  POKE 252,VT: POKE 253,VBOT: POKE 254,HRIGHT: POKE 255,HLEFT
155  HCOLOR= 3
160  HPLOT 7 * HRIGHT + 7,VT TO 7 * HRIGHT + 7,VB TO 7 * HLEFT,VB TO 7 * H
     LEFT,VT TO 7 * HRIGHT + 7,VT
170  IF ZQ = 1 THEN  RETURN
```

Listing continued.

Listing continued.

```
180  PRINT : INPUT "IS THE RECTANGLE DONE O.K.? (Y/N):";AN$: IF  LEN (AN$)
      = 0 THEN 180
185  IF  ASC (AN$) = 78 THEN SG = 0: HCOLOR= 0:FL = 0:ZQ = 1: GOSUB 160:ZQ
      = 0: HCOLOR= 3: GOTO 50
191  GOTO 600
204  HOME : VTAB 21
205  PRINT "SHAPE # "ST
208  POKE 7,ST: POKE  - 16304,0: POKE  - 16297,0
210  IF QZ = 0 THEN QZ = 1:ZQ = 1: HCOLOR= 0: GOSUB 160: HCOLOR= H
215 NN = NN + 1
220  CALL 2048
225  IF NN >  = NS THEN 300
240  FOR QQ = 1 TO SS: CALL 2186: NEXT
245 ST = ST + 1
250  GOTO 204
300 D$ =  CHR$ (4)
301  VTAB 21
302  INPUT "FILE NAME: ";N$: IF  LEN (N$) = 0 THEN 302
303  INPUT "DID YOU GET IT RIGHT? (Y/N):";Z$: IF  LEN (Z$) = 0 THEN 302
304  IF  ASC (Z$) <  > 89 THEN 302
307  TEXT : VTAB 1: HOME : GOSUB 5040
308 LL = 256 * LS
309 PL = LS
310  PRINT D$"BSAVE";N$;",A2304,L";LL
312  PRINT "LAST SHAPE AND ALL THE SHAPES THAT CAME BEFORE IT TOOK UP "LL"
      BYTES.": PRINT "LAST SHAPE: ";LS: PRINT "(HIT ANY KEY TO CONTINUE):"
     : GOSUB 63010
400  GOTO 600
402  HOME : VTAB 21: INPUT "STEP SIZE: ";SS
403  PRINT : INPUT "# OF SHAPES IN SEQUENCE: ";NS: PRINT : INPUT "# OF 1ST
      BLOCK-SHAPE IN SEQUENCE TO BE  SAVED: ";ST
404  PRINT : INPUT "READY TO BEGIN AUTOMATIC SCAN & SAVE    PROCESS FOR TH
     IS SEQUENCE? (Y/N):";QW$: IF  LEN (QW$) = 0 THEN 600
405  IF  ASC (QW$) <  > 89 THEN 600
410  GOTO 204
600  POKE  - 16303,0: POKE  - 16298,0: HOME : VTAB 1: INVERSE : HTAB 18: PRINT
     "MENU:": NORMAL
601 SG = 0:FL = 0:ZQ = 0:QZ = 0:NN = 0
602  SCALE= S: HCOLOR= H: ROT= R
603  PRINT "(HIT ESC TO QUIT)": PRINT
605  PRINT "(0)ABORT SCREEN---START OVER": PRINT
610  PRINT "(1)LOAD BLOCK SHAPE TABLE": PRINT
640  PRINT "(2)GIVE HOR. STEP SIZE FOR BLOCK-SHAPE    SEQUENCE & SAVE ENT
     IRE SEQUENCE": PRINT
650  PRINT "(3)DEFINE BLOCK SHAPE WITH PADDLES": PRINT
660  PRINT "(4)VIEW SCREEN": PRINT
690  FLASH : PRINT "(CHOOSE 0-4):";: NORMAL : GET A$: PRINT  CHR$ (13)
692  IF  ASC (A$) = 27 THEN  TEXT : HOME : END
700  IF  LEN (A$) = 0 THEN 690
710  IF  VAL (A$) < 0 OR  VAL (A$) > 4 THEN 690
719  IF A$ = "0" THEN 912
720  ON  VAL (A$) GOTO 2,402,47,920,600
721  GOTO 600
912  INPUT "SURE YOU WANT TO ABORT SCREEN? (Y/N):";QW$: IF  LEN (QW$) = 0 THEN
     912
913  IF  ASC (QW$) <  > 89 THEN 600
914  HGR : GOTO 600
920  POKE  - 16304,0: POKE  - 16297,0: VTAB 21: GOSUB 63000: GOTO 600
2500  POKE 2296,1: POKE 2297,0: POKE 2298,4: POKE 2299,0: POKE 2300,4: POKE
     2301,0
2510  POKE  - 16301,0
2511  POKE  - 16303,0: POKE  - 16298,0: INVERSE : PRINT "IF YOU ENTERED TH
     IS PROGRAM WITH SOME-  THING ON THE HI-RES SCREEN YOU WANTED TOSAVE,
     HIT THE SPACE BAR NOW--            OTHERWISE HIT ANY KEY EXCEPT THE SP
     ACE  BAR.": NORMAL
2512 PK =  PEEK ( - 16384): IF PK > 127 THEN  POKE  - 16368,0: GOTO 2514
2513  GOTO 2512
2514  IF PK = 160 THEN 2520
2515  HGR
2520  RETURN
5040  HOME : VTAB 21: INPUT "# OF LAST SHAPE IN BLOCK-SHAPE TABLE: ";LS: IF
     LS < 1 OR LS > 23 THEN 5040
5050  RETURN
63000  PRINT "                                        (HIT ANY KEY TO CONT
     INUE):                 "
63010 PK =  PEEK ( - 16384): IF PK > 127 THEN  POKE  - 16368,0: RETURN
63020  GOTO 63010
63990  PRINT  CHR$ (7): POKE 216,0
63991 PP =  PEEK (222): IF PP = 254 THEN  RESUME
63994  POKE  - 16303,0: POKE  - 16298,0
63995  PRINT "YOUR ERROR IS CODE #:"PP: GOSUB 63000: CALL 54915: GOTO 600
```

loading various vector and block shapes into SCANA (see the April column) and saving them at pre-calculated coordinates (saving them as various shape table numbers).

Looking at Figure 2a again, notice that the block shapes are 5 bytes wide, but could just as easily have been 4 bytes wide. (The extra "blank" byte was used for experimental purposes.) Now, look at the first and seventh shapes. Where would we put the next (eighth) shape if we were to continue the sequence, and what would it look like?

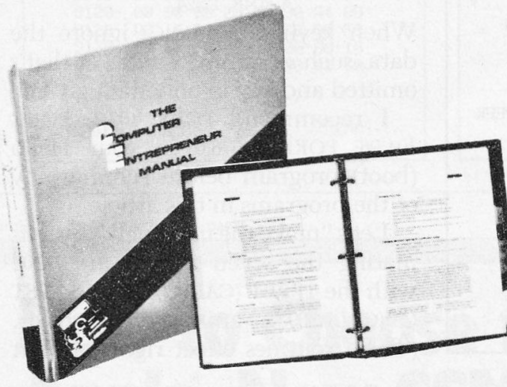
Well, first notice that each shape is being moved 2 dots to the right of the previous one. Then observe that we'll be looking for a shape like sequence number 1 to continue the "movement." Also note that X=49 will be the horizontal coordinate of the center of the next shape, so the first shape in the sequence will end up centered exactly on the line again, just as it is in its diagram position. Since the figure in the block shape sequence will move over exactly 2 bytes and the step value, in dots, between each of the figures in the shape sequence is 2, then that means the step value is equal to the required horizontal byte-coordinate increment we'll be using just before starting the sequence over.

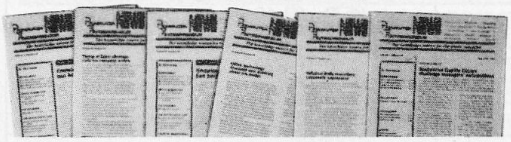
What this means is that during the display of the seven shapes shown, all block shape coordinates stay exactly the same. It's only just before the sequence restart that the horizontal byte coordinate gets increased by 2. So what's happening, in effect, is that most of the movements of the block shape figures take place within the boundaries of the block shape, and not by coordinate manipulation. Incidentally, all shape numbers given in MANA and MANC are taken from MAN, a vector shape table for a man walking. The actual shape numbers you'll refer to as you build and use a block shape table such as MANC are shapes 1–7, equivalent to sequence numbers 1–7.

Two-Page Flipping

When you use pre-shifted shapes of the block shape sequence variety and then go for unflickering smoothness by use of two-page flipping animation, the level of complexity goes up by several orders of magnitude. Check out the table and you'll see that things can get awkward pretty quickly. You draw on one screen and display the other. One screen will get the sequence 1, 3, 5, 7, 2, 4, 6, 1, 3, etc., while the other screen is getting, alternately, 2, 4, 6, 1, 3, 5, 7, etc.

The strangest part is when <single asterisk> HL <horizontal left byte coordinate> is 0 while erasing shape 6, 1 while drawing shape 1, back to 0 for erasing shape 7 on the opposite screen (double asterisks) and up to 1 again for drawing shape 2. It's important to keep good charts of what's happening when coding such animation routines.


THE COMPUTER ENTREPRENEUR MANUAL



```
0  ONERR  GOTO 63990
1  POKE 8,0: REM  8 MUST BE 0ED FOR THIS PROG. TO WORK!!!!!!!!!!!!!
5  HIMEM: 36864
10 D$ =  CHR$ (4)
25  TEXT : INPUT "SHAPE TABLE NAME: ";N$: IF  LEN (N$) = 0 THEN 25
27  PRINT : INPUT "YOU WANT YOUR SHAPE TO TRAVEL:              (1) ----> RIGHT
     WARDS                        (2) <---- LEFTWARDS                     (1-2
     ):";Q: IF Q < 1 OR Q > 2 THEN 27
28  IF Q = 1 THEN  PRINT  CHR$ (4);"BLOADTEST F (CALL36934)"
29  IF Q = 2 THEN  PRINT  CHR$ (4);"BLOADTEST G (CALL36934)"
30  PRINT : PRINT "INSERT YOUR SHAPE TABLE DISK NOW:": GET A$: PRINT  CHR$
     (13): CALL 1002: PRINT D$"BLOAD";N$: PRINT "ADDRESS: " PEEK (43634) +
      PEEK (43635) * 256: PRINT "LENGTH: " PEEK (43616) +  PEEK (43617) *
     256
31  INPUT "WIDTH:";WD: INPUT "HEIGHT:";HT: INPUT "STEP SIZE:";SS: INPUT "R
     IGHT BOUNDARY OF LEFT SIDE OF SHAPE:";RB: IF Q = 1 THEN  POKE 235,WD -
     SS: POKE 29,(RB - SS) + WD: POKE 30,RB - SS
32  INPUT "# OF 1ST SHAPE IN SEQUENCE: ";SH: POKE 239,WD: POKE 238,RB: POKE
     237,HT: POKE 236,SS: IF Q = 2 THEN  POKE 25,39 - WD: POKE 235,SS + 39

33  IF Q = 1 THEN  POKE 36955,SH + 1: POKE 36987,SH: POKE 36994,SH + 1: POKE
     37029,SH + 5: POKE 37046,SH: POKE 37080,SH - 1: POKE 37084,SH + 6: POKE
     37133,SH + 1
34  IF Q = 2 THEN  POKE 37092,SH + 7: POKE 37096,SH: POKE 37152,SH + 5: POKE
     36955,SH + 5: POKE 36987,SH + 6: POKE 36994,SH + 5: POKE 37029,SH + 1
     : POKE 37046,SH + 6
35  TEXT : INPUT "DELAY LOOP HI BYTE (1-255):";A: IF A < 1 OR A > 255 THEN
     35
36  POKE 9,A
37  PRINT : INPUT "DELAY LOOP LO BYTE (1-255):";B: IF B < 1 OR B > 255 THEN
     37
38  POKE 31,B
40  CALL 36934
42  HOME
43  TEXT : PRINT "$EF:" PEEK (239): PRINT "$EE:" PEEK (238): PRINT "$ED:" PEEK
     (237): PRINT "$EC:" PEEK (236): PRINT "$EB:" PEEK (235): PRINT "$1D:"
      PEEK (29): PRINT "$1E:" PEEK (30): PRINT : PRINT
45  TEXT : INPUT "DO YOU WANT TO SEE IT SOME MORE? (Y/N):";Q$: IF  LEN (Q$
     ) = 0 THEN 45
46  IF  ASC (Q$) <  > 89 THEN  END
50  GOTO 31
63990  POKE 216,0
63991  ONERR  GOTO 63990
63992 PK =  PEEK (222): IF PK = 254 THEN  RESUME
63995  GOTO 0
```

Listing 2. ASMINPUT.

Now, if you key in TEST F(CALL 36934) and TEST G (CALL36934) in Listings 4 and 5, and then MANC and ASMINPUT in Listings 3 and 2, you'll have a two-page flipping block shape sequence using routine for moving left or right using seven-shape block sequences. Here are some BSAVE addresses and lengths for various files in these listings:

Listing 3	MANC,A$900,L1646 (step 2, 21 high, 4 wide, 7 shapes)
Listing 4	TEST F (CALL36934), A 36864, L324
Listing 5	TEST G (CALL36934), A 36864, L342
Listing 6	TEST H (CALL2186), A 2048, L224

When keying in MANC, ignore the data, such as from $970 to $9FF, that's omitted and key in only data given.

I recommend POKE 103,1: POKE 104,96: POKE 24576,0 in your Hello (boot) program before running any of the programs in this article.

Let's "make the man walk" by animating the seven shapes in MANC with the TEST F (CALL36934) and TEST G (CALL36934) animation routines. (These routines effect right and left

```
*900.96F


0900- 00 00 00 00 00 00 00 00
0908- 30 08 00 00 00 10 04 00
0910- 00 00 08 08 00 00 00 08
0918- 10 00 00 00 04 10 00 00
0920- 00 04 20 00 00 00 02 20
0928- 00 00 00 02 40 00 00 00
0930- 03 40 00 00 00 13 44 00
0938- 00 00 0B 48 00 00 00 07
0940- 50 00 00 00 03 60 00 00
0948- 00 03 40 00 00 00 03 40
0950- 00 00 00 03 40 00 00 00
0958- 01 00 00 00 00 03 00 00
0960- 00 00 03 00 00 00 00 00
0968- 00 00 00 00 00 00 B8 0B
*A00.A6F

0A00- 00 00 00 00 00 00 00 00
0A08- 00 00 00 00 00 61 00 00
0A10- 00 00 20 40 00 00 00 21
0A18- 00 00 00 00 12 00 00 00
0A20- 00 12 00 00 00 00 0A 00
0A28- 00 00 00 0A 00 00 00 00
0A30- 4E 20 00 00 00 2E 40 00
0A38- 00 00 1E 40 00 00 00 0F
0A40- 00 00 00 00 0E 00 00 00
0A48- 00 0E 00 00 00 00 0E 00
0A50- 00 00 00 0E 00 00 00 00
0A58- 04 00 00 00 00 0C 00 00
0A60- 00 00 0C 00 00 00 00 00
0A68- 00 00 00 00 00 00 00 00
*B00.B6F

0B00- 00 00 00 00 00 00 00 01
0B08- 48 00 00 00 00 44 00 00
0B10- 00 00 48 00 00 00 00 50
0B18- 00 00 00 00 30 00 00 00
0B20- 00 30 00 00 00 00 30 00
0B28- 00 00 00 30 00 00 00 00
0B30- 38 00 00 00 01 3A 00 00
0B38- 00 00 7C 00 00 00 00 38
0B40- 00 00 00 00 38 00 00 00
0B48- 00 38 00 00 00 00 38 00
0B50- 00 00 00 38 00 00 00 00
0B58- 10 00 00 00 00 30 00 00
0B60- 00 00 30 00 00 00 00 00
0B68- 00 00 00 00 00 00 00 00
*C00.C6F
0C00- 00 00 00 00 00 00 00 00
0C08- 00 00 00 00 01 50 00 00
0C18- 00 00 00 00 60 00 00 00
0C20- 01 40 00 00 00 02 40 00
0C28- 00 00 01 40 00 00 00 01
0C30- 68 00 00 00 05 70 00 00
0C38- 00 03 60 00 00 00 01 60
0C40- 00 00 00 01 60 00 00 00
0C48- 01 60 00 00 00 01 60 00
0C50- 00 00 01 60 00 00 00 00
0C58- 40 00 00 00 01 40 00 00
0C60- 00 01 40 00 00 00 00 00
0C68- 00 00 00 00 00 00 00 00
*D00.D6F

0D00- 00 00 00 00 00 00 00 0E
0D08- 00 00 00 00 06 00 00 00
0D10- 00 0A 00 00 00 00 0A 00
0D18- 00 00 00 0A 00 00 00 00
0D20- 0A 00 00 00 00 06 00 00
0D28- 00 00 06 00 00 00 00 0F
0D30- 40 00 00 00 07 00 00 00
0D38- 00 07 00 00 00 00 07 00
0D40- 00 00 00 07 00 00 00 00
0D48- 07 00 00 00 00 07 00 00
0D50- 00 00 07 00 00 00 00 02
0D58- 00 00 00 00 06 00 00 00
0D60- 00 06 00 00 00 00 00 00
0D68- 00 00 00 00 00 00 00 00
*E00.E6F

0E00- 00 00 00 00 00 00 00 06
0E08- 00 00 00 00 12 00 00 00
0E10- 00 0A 00 00 00 00 12 00
0E18- 00 00 00 22 00 00 00 00
0E20- 24 00 00 00 00 14 00 00
0E28- 00 00 14 00 00 00 00 1C
0E30- 00 00 00 00 1C 00 00 00
0E38- 00 1C 00 00 00 00 1C 00
0E40- 00 00 00 1C 00 00 00 00
0E48- 1C 00 00 00 00 1C 00 00
0E50- 00 00 1C 00 00 00 00 08
0E58- 00 00 00 00 18 00 00 00
0E60- 00 18 00 00 00 00 00 00
0E68- 00 00 00 00 00 00 00 00
*F00.F6F

0F00- 00 00 00 00 00 00 00 00
0F08- 00 00 00 06 08 00 00 00
0F10- 02 04 00 00 00 02 08 00
0F18- 00 00 01 10 00 00 00 01
0F20- 10 00 00 00 00 50 00 00
0F28- 00 00 50 00 00 00 04 72
0F30- 00 00 00 02 74 00 00 00
0F38- 01 74 00 00 00 00 78 00
0F40- 00 00 00 70 00 00 00 00
0F48- 70 00 00 00 00 70 00 00
0F50- 00 00 70 00 00 00 00 20
0F58- 00 00 00 00 60 00 00 00
0F60- 00 60 00 00 00 00 00 00
0F68- 00 00 00 00 00 00 00 00
```

Listing 3. MANC.

```
*9000.9144

9000- A0 09 A6 07 CA E0 00 F0
9008- 04 C8 4C 04 90 98 85 FB
9010- A9 00 85 FA A5 FD 85 06
9018- A2 00 A0 00 20 11 F4 A4
9020- FE A2 00 A1 FA 51 26 91
9028- 26 88 18 E6 FA D0 02 E6
9030- FB C0 FF F0 04 C4 FF B0
9038- EA C6 06 A5 06 C9 FF F0
9040- 04 C5 FC B0 D3 60 20 E2
9048- F3 A9 00 8D 52 C0 20 D8
9050- F3 A9 40 85 E6 A9 00 8D
9058- 54 C0 A9 02 85 07 A9 00
9060- 85 FF A5 EF 85 FE A9 00
9068- 85 FC A5 ED 85 FD 20 00
9070- 90 A9 00 8D 55 C0 A9 20
9078- 85 E6 A9 01 85 07 20 00
9080- 90 A9 02 85 07 A9 20 85
9088- E6 A9 00 8D 55 C0 20 A0
9090- 90 A9 40 85 E6 A9 00 8D
9098- 54 C0 20 A0 90 4C 85 90
90A0- C6 07 A5 07 C9 06 D0 2D
90A8- 20 00 90 A4 09 A6 1F CA
90B0- D0 FD 88 D0 F8 A9 01 85
90B8- 07 A4 EC E6 FE E6 FF 88
90C0- D0 F9 18 A5 FF C5 EE 90
90C8- 08 A5 EF 85 FE A9 00 85
90D0- FF 20 00 90 60 A5 07 C9
90D8- 00 D0 48 A9 07 85 07 A4
90E0- EC C6 FE C6 FF 88 D0 F9
90E8- A5 FE C5 EB D0 08 A5 1E
90F0- 85 FF A5 1D 85 FE 20 00
90F8- 90 A4 09 A6 1F CA D0 FD
9100- 88 D0 F8 A4 EC E6 FE E6
9108- FF 88 D0 F9 A9 02 85 07
9110- 18 A5 FF C5 EE 90 08 A5
9118- EF 85 FE A9 00 85 FF 20
9120- 00 90 60 20 00 90 A4 09
9128- A6 1F CA D0 FD 88 D0 F8
9130- E6 07 E6 07 20 00 90 18
9138- AD 00 C0 C9 80 B0 01 60
9140- 68 28 60 00 00
```

Listing 4. TEST F (CALL36934).

```
9000- A0 09 A6 07 CA E0 00 F0
9008- 04 C8 4C 04 90 98 85 FB
9010- A9 00 85 FA A5 FD 85 06
9018- A2 00 A0 00 20 11 F4 A4
9020- FE A2 00 A1 FA 51 26 91
9028- 26 88 18 E6 FA D0 02 E6
9030- FB C0 FF F0 04 C4 FF B0
9038- EA C6 06 A5 06 C9 FF F0
9040- 04 C5 FC B0 D3 60 20 E2
9048- F3 A9 00 8D 52 C0 20 D8
9050- F3 A9 40 85 E6 A9 00 8D
9058- 54 C0 A9 06 85 07 A5 19
9060- 85 FF A9 27 85 FE A9 00
9068- 85 FC A5 ED 85 FD 20 00
9070- 90 A9 00 8D 55 C0 A9 20
9078- 85 E6 A9 07 85 07 20 00
9080- 90 A9 06 85 07 A9 20 85
9088- E6 A9 00 8D 55 C0 20 A0
9090- 90 A9 40 85 E6 A9 00 8D
9098- 54 C0 20 A0 90 4C 85 90
90A0- E6 07 A5 07 C9 02 D0 39
90A8- 20 00 90 A4 09 A6 1F CA
90B0- D0 FD 88 D0 F8 A9 07 85
90B8- 07 A4 EC C6 FE C6 FF 88
90C0- D0 F9 A5 FE C5 EF B0 15
90C8- 18 A5 FE 65 EC 85 1D A5
90D0- FF 65 EC 85 1E A9 27 85
90D8- FE A5 19 85 FF 20 00 90
90E0- 60 A5 07 C9 08 D0 4E A9
90E8- 01 85 07 A4 EC E6 FE E6
90F0- FF 88 D0 F9 A5 FE C5 EB
90F8- D0 0F 38 A5 1D E5 EC 85
9100- 08 A5 1E 85 FF A5 1D 85
9108- FE 20 00 90 A4 09 A6 1F
9110- CA D0 FD 88 D0 F8 A4 EC
9118- C6 FE C6 FF 88 D0 F9 A9
9120- 06 85 07 A5 FE C5 08 D0
9128- 08 A9 27 85 FE A5 19 85
9130- FF 20 00 90 60 20 00 90
9138- A4 09 A6 1F CA D0 FD 88
9140- D0 F8 C6 07 C6 07 20 00
9148- 90 18 AD 00 C0 C9 80 B0
9150- 01 60 68 28 60 00 00 00
```

Listing 5. TEST G (CALL36934).

```
*800.8E2

0800- A0 09 A6 07 CA E0 00 F0
0808- 04 C8 4C 04 08 98 85 FB
0810- A9 00 85 FA A5 FD 85 06
0818- A2 00 A0 00 20 11 F4 A4
0820- FE A2 00 B1 26 81 FA 88
0828- 18 E6 FA D0 02 E6 FB C0
0830- FF F0 04 C4 FF B0 EC C6
0838- 06 A5 06 C9 FF F0 04 C5
0840- FC B0 D5 60 A0 09 A6 07
0848- CA E0 00 F0 04 C8 4C 48
0850- 08 98 85 FB A9 00 85 FA
0858- A5 FD 85 06 A2 00 A0 00
0860- 20 11 F4 A4 FE A2 00 A1
0868- FA 51 26 91 26 88 18 E6
0870- FA D0 02 E6 FB C0 FF F0
0878- 04 C4 FF B0 EA C6 06 A5
0880- 06 C9 FF F0 04 C5 FC B0
0888- D3 60 A5 FD 85 06 A2 00
0890- A0 00 20 11 F4 18 A4 FF
0898- A9 00 85 08 85 CF B1 26
08A0- 2A 91 26 B0 02 90 02 E6
08A8- 08 C9 80 B0 02 90 02 E6
08B0- CF A5 08 D0 09 B1 26 29
08B8- 7F 91 26 4C C4 08 B1 26
08C0- 09 80 91 26 C4 FE F0 09
08C8- C8 18 A5 CF C9 01 4C 98
08D0- 08 C6 06 A5 06 C9 FF F0
08D8- 04 C5 FC B0 B1 60 00 00
```

Listing 6. TEST H (CALL2186).

movement.) The Basic *driver* program we'll use (to be RUN now) is ASMINPUT. Give the shape table name of MANC. (I'm assuming you've saved the necessary files.) Say RIGHTWARDS for direction of travel. Specify a width of 4, a height of 21, a step size of

2, a right boundary of 34, and a first shape in sequence of 1. Then give a delay loop high byte of 70 and a delay loop low byte of 255.

This results in a realistic walking speed and no flicker problems (due to the two-page flipping animation in these routines), but the two-step movements are somewhat noticeable. A good way to improve this would be to have 14 shapes in the sequence and do one-step movements, still incrementing the horizontal byte coordinate by 2 at the end of the sequence, but using only one half the delay time. Notice that when HR (horizontal right coordinate) gets to 34 the sequence restarts. That's the 34 you input above. Hit any key and do it again, but use a delay loop high byte of 35 so things move twice as fast. (Ignore the $EE:34 and other data given on the screen; this just indicates how the variables are doing once the action stops.)

Now he's marching along. You can see that the step movements within block shape parameters are no longer noticeable. Now try a step value of 1 and a delay loop high byte of 70 again. Notice how block shape sequences meant for a step value of 2 do weird things with a value of 1. But also notice that the intra-block step movements, being only 1 dot each, are much less obvious with a step value of 1. These latter two experiments should support the idea that 14 one-step shapes with a horizontal byte coordinate increment of 2 will yield the smoothest results. (Incidentally, when moving to the left, choose LEFTWARDS, but for the other inputs choose the same as you did for RIGHTWARDS.)

Sequence Creation

Now key in TEST H (CALL2186) and SEQUENCE CREATOR, Listings 6 and 1. (Don't forget about the POKE 104,96, etc., as advised previously.) Then RUN SEQUENCE CREATOR, hitting return upon entry into the program. Choose (1) LOAD IN BLOCK SHAPE TABLE and give MANC as the shape table name. Then specify shape number 1, VTOP of 10, VBOT of 31, HRIGHT of 5, HLEFT of 1, and no, you don't need any more shapes (when asked).

Now, in the menu, choose (3) DEFINE BLOCK SHAPE WITH PADDLES, read the instructions, and move the paddles to find out which one makes the dot cursor move horizontally. We'll call this paddle your X paddle and the other your Y paddle. Move the dot cursor just outside the upper left corner of the imaginary rectangular block around the man shape, and hit the X paddle button. Now move to the lower right corner and hit the Y paddle button—*but not until you've moved at least 7 dots to the right of that position, to make room for intra-block step movements.* (Use a 14-dot offset if your step value is 2 and a 21-dot offset if your step value is 3, and so on.) Seven times the step value to the right (lower) of your block shape is where you'll hit the Y button.

When asked if the rectangle (which defines the block shape parameters) is okay, answer yes or no. No gets you another chance. Now choose (2) GIVE HORIZONTAL STEP SIZE FOR BLOCK-SHAPE SEQUENCE & SAVE ENTIRE SEQUENCE, and specify 1 for step size, 7 for number of shapes in sequence, 1 for number of first block shape in sequence to be saved, and Y (yes) for "Are you ready for this sequence?" Keep your eyes peeled, and you'll see all seven shapes made by shifting (after which each in turn will be scanned and the resultant data saved in memory). When asked for file name use TEST and give 7 as the number of the last shape in the shape table.

Once the sequence is saved, use it when you RUN ASMINPUT to check the latter out. Step size must be 1, unless you used something greater than that in your sequence creation. Unless you've made a mistake the man will float very smoothly.

If all this sounds like it's right up your alley, drop me a line for more information on routines and utilities for graphics, sounds and more.

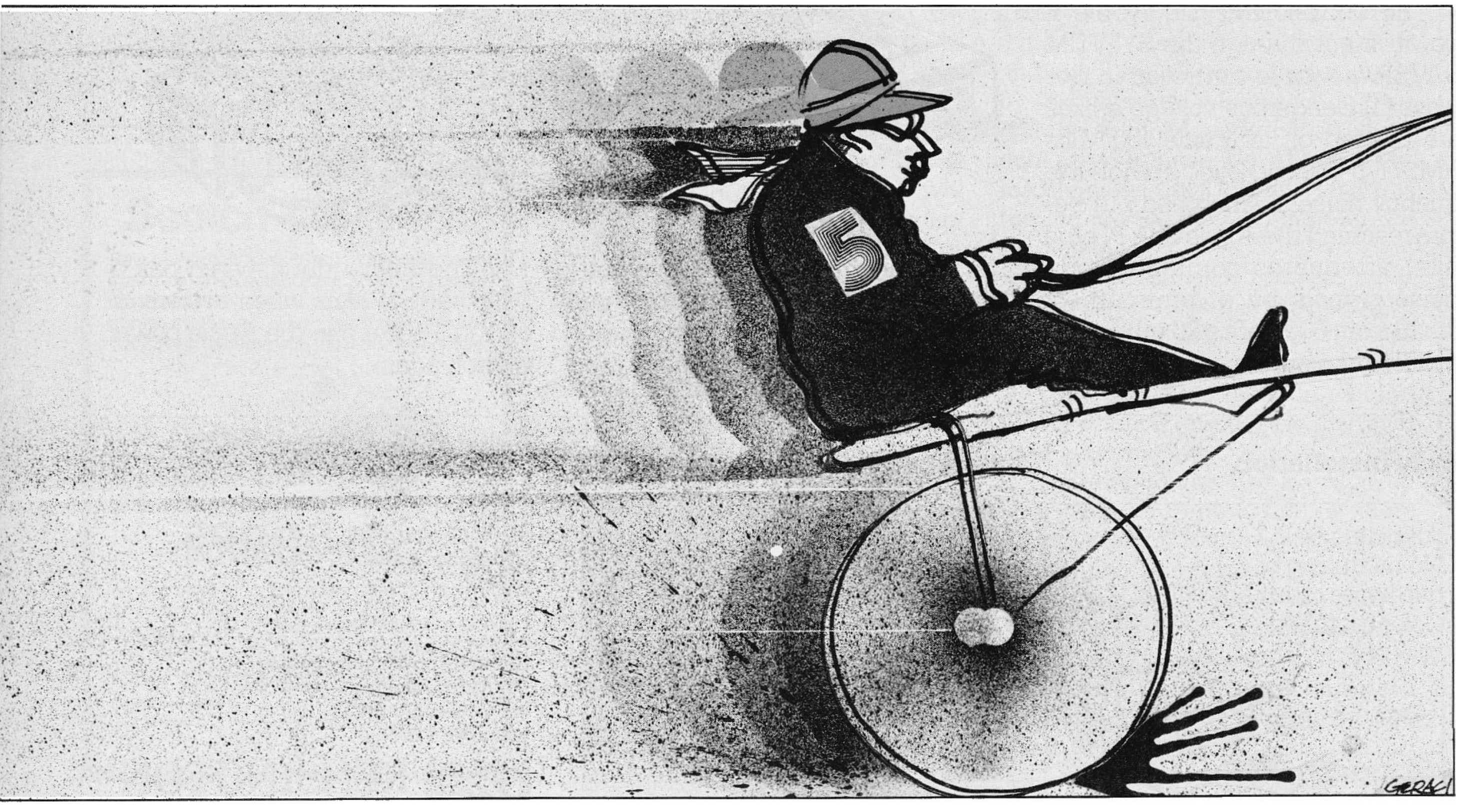
Next time I'll dissect the fastest color-fill algorithm around to show how it works. You'll get a chance to save it, a program to use it with and a palette of colors for posterity. See you then! ■

Harness Race Handicapper

Picture this: A late race at The Meadows, with unbelievable odds; a sudden snow squall slows the track. What to do? Go for it!

by Steven A. Schwartz

Attending the harness races can be a pleasurable leisure activity and, for a select set of individuals, a profitable pursuit. Gambling for profit, however, involves special learned skills, not just "luck" (as many would have you believe). Individuals who are consistent winners and who profit from the harness races year after year are of two primary types: those with a thorough knowledge of the sport who have developed the necessary personal skill to allow them to predict the outcome of races (to handicap); and those who rigorously employ a proven system. Consistently betting the horse with the lowest odds (the favorite), your favorite number, the age of your youngest child, your apartment number, and others, may all be rigorously followed and do qualify as systems.



Son of "The Inflation Fighter"

The *Orange+Two*" $1095

Suggested List Price

New Math from Orange+ Computer Systems

Z80A CPU + 6502 CPU = *"Orange+Two"*

We give you two computers for the price of one, because we've put them both in the same machine. The "Orange+Two" is the latest "state of the art" personal business/home computer with two micrprocessors, each operating independently. Consider: CP/M 3.0, Digital Research's new enhanced CP/M version, AND 6502. That means the "Orange+Two" will run over 20,000 currently available programs, including some of the most popular business and game software on the market today. And, it uses the new easy-to-learn, easy-to-program language: FORTH-79.

STANDARD FEATURES:

— Z80A plus 6502 microprocessing units
\+ 64K RAM Bank Switchable, fully expandable
\+ Built-in disk drive controller for two drives
\+ Programmable ASCII keyboard with numeric keypad and auto repeat
\+ Built-in control-reset function
\+ Audio volume control switch
\+ High resolution video display with graphics
\+ Cassette interface for tape backup unit
\+ Game input and output connector for joystick
\+ Operating languages: FORTH-79, BASIC, CP/M 3.0
\+ Streamlined, high-impact case with removable lid, allowing full access to interior
\+ Metal base plate with motherboard connectors for fast board removal and replacement (no screws)
\+ 110/220 switch selectable power supply. Operating range 47-65 Hz
\+ Programmable EPROMS (2764). Allows user to do custom ROM programming

The "Orange+Two" is the pacesetter of the future. Why buy 1970's technology at 1983 prices, when you can purchase two computers for the cost of one. Ask your favorite dealer for the "Orange+Two".

The "Orange+Two" carries a full 90 day warranty. Nine-month extended warranty is available for an additional $99.00.

Prices & Specifications are subject to change without notice. Limited quantities are subject to availability.

If "Orange+Two" is not available locally, you can order factory direct. To order factory direct, call

(213) 999-5210

Allow 30 days for delivery from receipt of order.
*California residents add 6½% sales tax.

or send bank check or money order for $1095* to Collins International Trading Corporation, 23801 Calabasas Road, Suite 2050, Calabasas, California 91302

DEALER INQUIRIES INVITED.

Circle 14 on Reader Service card

"Before trying the program, be sure to read the whole article and work through some examples."

However, if you place large enough bets, each may eventually have you filing for bankruptcy. The point is that whether you develop finely honed handicapping skills or bet a system, the sole object is to reduce or, if possible, eliminate the "luck factor" and decrease the amount of risk involved in placing a bet.

Harness Race Handicapper is based on a statistical technique called multiple linear regression. Basically, a formula is created that is composed of a weighted combination of important predictor variables like post position, speed in recent races, etc. The weights are determined by the aforementioned statistical technique. This particular system was developed after extensive computer data analysis of an entire season's racing forms for the Windsor Raceway in Ontario. When the weighted variables are combined, a predicted finish score is obtained for each horse in the race. The lower the finish score, the greater the likelihood that the horse will finish "in the money."

There are several potential ways to use the program/system. First, if you are a relative newcomer to harness racing, it can be used to select probable winners in each race. No handicapping skill is required. Simply follow the program's instructions and place a bet on the horse with the lowest predicted finish position—the "top pick."

Second, if you are a casual bettor who attends races more for the action and excitement than for the potential of making a modest profit in the long run, you can use the system to help you place a bet for *every* race in an evening or afternoon's program. Although certain types of races are not easily predictable (as evidenced by the fact that professional gamblers seldom bet more than the few most "handicappable" races on the program), the system will generate predicted finishes for any race you care to enter.

Finally, if you do possess some knowledge of handicapping, you may find that the greatest use for the system will be in the elimination of probable losers. In most races you will find that the race winner will be one of the system's top three choices. Rather than simply betting the top pick each race, you can use your handicapping expertise to select among the three top picks for the bet with the largest profit potential.

Before trying the program, be sure to read the whole article and work through the examples. Although some of the preparations for coding may seem complex at first, as you become more familiar with the system and harness racing terminology, you should be able to handicap any race in 5 to 10 minutes.

Address correspondence to Dr. Steven A. Schwartz, 9226 Vantine St., Pittsburgh, PA 15235. Enclose S.A.S.E. for a reply.

Input Preparation

The bulk of the information used in the handicapping will be obtained from the racing form lines for each horse's last six races. From this material you will be required to *rank* the horses on several important variables—that is, compare each horse against the others in the race.

Adjustments to the Form

Because horses are driven at tracks other than the present one, occasionally have incomplete information for one or more races, and so on, it is often necessary to make adjustments to the form prior to entering information into the computer program. To accomplish this, follow these steps:

1) *Draw a line through each of the following types of* previous *races.* Those that are eliminated will not be considered in handicapping today's race.

a) Qualifiers (qua);

b) Any race that indicates *no betting* (NB) was allowed;

c) Any race in which the distance run was not equal to one mile (m);

d) Any race in which the horse was slowed by an equipment break (ex), an interference break (ix), or an accident break (ax). Note that the emphasis is on the word "slowed." If the horse

Track Abbreviations and Comparative Speed Ratings

Balmoral (Ill.)	Blmp	2:06
Batavia (N. Y.)	Btva	2:05^1
Bay Meadows (Calif.)	BM^1	2:04
Blue Bonnets (Que.)	BB$\frac{5}{8}$	2:03^1
Brandywine (Dela.)	Brd$\frac{5}{8}$	2:02^3
Buffalo (N. Y.)	BR	2:05^1
Delaware (Ohio)	Dela	2:04^3
Detroit-Wolverine (Mich.)	Det^1	2:01^2
Dover (Dela.)	DD$\frac{5}{8}$	2:04^2
DuQuoin (Ill.)	DuQ^1	2:01^1
Fairmount (Ill.)	FP^1	2:03^3
Freehold (N. J.)	Fhld	2:04^3
Greenwood (Ont.)	GrR$\frac{5}{8}$	2:04
Hawhtorne (Ill.)	Haw^1	2:03^4
Harrington (Dela.)	Har	2:05^2
Hazel Park (Mich.)	HP$\frac{5}{8}$	2:04
Hollywood (Calif.)	Hol^1	2:01^4
Indianapolis (Ind.)	Ind^1	2:01^4
Jackson (Mich.)	Jack	2:07
Latonia (Ky.)	Lat^1	2:03
Laurel (Md.)	Lau$\frac{5}{8}$	2:03^1
Lebanon (Ohio)	Leb	2:06^2
Lexington (Ky.)	Lex^1	2:01^2
Liberty Bell (Pa.)	LB$\frac{5}{8}$	2:02^3
Los Alamitos (Calif.)	LA$\frac{5}{8}$	2:03^4
Louisville (Ky.)	LouD	2:05^2
Maywood (Ill.)	May	2:04
Meadowlands (N. J.)	M^1	2:01^2
Meadows (Pa.)	Mea$\frac{5}{8}$	2:03
Midwest (Ky.)	Mid	2:05^3
Mohawk (Ont.)	Moh$\frac{5}{8}$	2:04^1
Monticello (N. Y.)	MR	2:03^1
New England (Mass.)	NE$\frac{5}{8}$	2:03
Northfield (Ohio)	Nfld	2:04^3
Northville (Mich.)	Nor	2:05
Ocean (Md.)	OD	2:04^3
Pocono (Pa.)	PcD$\frac{5}{8}$	2:03^1
Pompano (Fla.)	PPk$\frac{5}{8}$	2:03^2
Raceway (Ohio)	RP$\frac{5}{8}$	2:03^3
Richleu (Que.)	Rich	2:04^4
Rockingham (N. H.)	Rock	2:04^4
Roosevelt (N. Y.)	RR	2:03^2
Rosecroft (Md.)	RcR	2:03^4
Sacramento-Cal Expo	Sacr^1	2:03^2
Saratoga (N. Y.)	Stga	2:03^2
Scioto (Ohio)	ScD$\frac{5}{8}$	2:02^3
Seminole (Fla.)	Sem$\frac{5}{8}$	2:03^2
Sportsman's (Ill.)	Spk$\frac{5}{8}$	2:02^4
Springfield (Ill.)	Spr^1	2:01
Syracuse (N. Y.)	Syr^1	2:01^4
Vernon (N. Y.)	VD$\frac{5}{8}$	2:01^4
Windsor (Ont.)	WR$\frac{5}{8}$	2:03^1
Yonkers (N. Y.)	YR	2:04

Figure 1. Speed ratings.

"The issue is whether the equipment, interference or accident affected the horse's performance."

managed to finish in the money and in reasonable time, you may opt to leave the race as is. The issue is whether the equipment, interference or accident appears to have significantly affected the horse's performance. (This adjustment assumes that the breaks were not caused by or the fault of the horse, its driver, or the individual responsible for the horse's equipment, and do not reflect poor performance. Isolated incidents of this nature are not unusual. However, if a horse's past six races show a large number of such incidents, it may well reflect poor driver judgment, a difficult-to-control animal, or poor equipment maintenance procedures—and it may be best to leave the races in.) In Figure 2, Seventh Race at The Meadows, four races were eliminated for horse 3 on the basis of step 1.

2) *Any previous race run at a track other than the present one may require an adjustment to the horse's previous finish time.* This procedure will change a horse's time to what we might expect him to have run if the race had been held at the present track. Pinpoint all such races and do the following:

a) Locate the speed rating of the present track. This information, along with comparable figures for other major tracks, is typically printed in the daily racing form. See Figure 1 for an example.

b) Locate the speed rating for the other track.

c) Subtract the smaller speed rating from the larger speed rating. Note that all times on the form and the speed rating chart are given in minutes, seconds and *fifths* of a second. The latter are shown as smaller raised numbers. It will be easier to perform the calculations if you change the fifths to tenths (decimal notation) beforehand.

d) If the other track's speed rating is the *slower* of the two (larger number), *subtract* the difference between the two from each of the previous races run at the other track.

e) If the other track's speed rating is the *faster* of the two (smaller number), *add* the difference between the two to each of the previous races run at the other track.

In Figure 2, times were adjusted for horses 1, 3 and 6. The first converted times listed are actual times from the form listed in fifths of a second. The second set of times is in tenths of a second (decimal notation) and are expressed as *deviations from 2 minutes*. Performing all calculations on tenths as differences from 2 minutes will simplify the following steps. However, be *very* careful here! The conversion calculations should be closely checked. Be certain, for example, that your conversions from fifths to tenths of a second are correct and that you have not added

"A single horse in a given race with virtually no prior codable races makes handicapping very difficult."

when you should have subtracted.

Special note: The program uses only the *four most recent* codable finish times when making its calculations. For horse 1 (Figure 2), all six times have been converted for the purpose of illustration—even though only the times on 6/7, 5/29, 5/21 and 5/15 will be used to handicap the race. Thus, unless one of the four most recent races has been eliminated in step 1 or altered during the next step, you need not worry about time conversions for the fifth or sixth most recent races.

3) *In any race in which the horse was distanced (dis) or did not finish (DNF)* and *a finish time was not recorded, take the winning time for the race and add 10 seconds to it.* This will serve as an approximation of the horse's finish time for that race.

Final Input Preparation

Here, and throughout the remainder of the article, the term "codable" race is synonymous with "usable." It refers to any race among the previous six that has not been eliminated due to adjustment step 1.

1) *Sum of the last three finishes.* Add the finish positions for the three most recent codable races for each horse. If only two usable races exist for a given horse (see horse 3 in Figure 2, for example), take the two codable finish positions that remain, add them together, average them, and add the average to the total. For horse 3, this would be:

5 + 4 = 9 (add the two usable finish positions)
9 / 2 = 4.5 (calculate the average)
5 + 4 + 4.5 = 13.5 (add the average to the total)

For a horse with only a single codable race, simply multiply the horse's finish position by three. Be aware, however, that a single horse in a given race with virtually no prior codable races makes handicapping very difficult. Horse 3 in Figure 2 is a good example of this. If it appears that the horse may have potential and you do not feel there is sufficient information for an accurate handicapping, it might be best to pass on the race altogether. If you do not feel competent to make this judgment, I would suggest that you pass on any race that has a horse with fewer than three codable prior races.

This method of replacing missing information (adding, averaging, and adding the average to the total enough times to make up the required number of data elements) is used throughout, with the sole exception of approximating the finish time of a horse who was distanced or did not finish (adjustment step 3). Once the sum of the last three finishes has been calculated for each horse, place the horses in rank order (from 1 to the number of post positions in the race). The horse with the best performance on this measure (the

1	6.40	4.60	3.60
3	8.60	4.20	
2	3.00		

PACE — 1 MILE
Purse $4 400
* If nine starters purse $4600

SEVENTH RACE

Warmup Saddle Pad Color — Black

OPEN

TRIFECTA RACE — **Quinella Wagering Race**

Times for #1
Sacr = 2:03^4
M' = 1:59^2
" = 2:03^2
" = 2:00
" = 2:00
" = 2:08^3

1 — OSBORNES KNIGHT (Red, 4-1)

br g, 76, Osborne Creed—Tina Hill—Knight Time
Archie McNeil & Jerry Osborne, Fla. & Ohio

ARCHIE McNEIL (160 lbs.) (8-13-45) Green-Brown-Gold
10 - 3 - 1 - 2 — .422

Trainer—T. Sullivan $6,745.00 (3, 2:01 ⅝)
1980 14 4 0 0 $14,225.00 (1:58^2 M^1)
1979 14 6 0 0 $ 6,745.00 (2:01 PPk⅝)

6- 7 Sac	ft	Inv	7500	m	29^4	1:01	1:31^3	2:02	5	5°	2°	3°	5^{4½}	6^{11½}	2:04^1	*.85	(A.McN)	BaronsImage SaulsPride RangerBill	
5-29 M^1	ft	cd	15000	m	30	59	1:28	1:57^2	1	3	5	7	6^4	5^2	1:57^4	59.60	(A.McN)	TaurusRomeo ChetHnvrN HornbyThorpe	
5-21 M^1	gd	cd	15000	m	29^4	59^2	1:28^4	1:58	8	4°	2	5	5^6	6^{19}	2:01^4	12.20	(A.McN)	TheBattler TexasTea DellsCourt	
5-15 M^1	ft	cd	14000	m	29^4	1:00	1:29^1	1:58	3	4	1°	2°	2^{1½}	5^{2¼}	1:58^2	5.70	(A.McN)	RobbyTanner Tweety EdensImp	
5- 5 M^1	ft	cd	9500	m	29^4	59^1	1:29^3	1:58^2	7	4°	1	1	1^{2¼}	1^{1¼}	1:58^2	*2.10	(A.McN)	OsbornesKnight KenwdBtsy TwinBBret	
4-28 M^1	sl	cd	9500	m	31	1:01^4	1:34^2	2:04^3	x2x	10	9	9	7^{6½}	5^{12½}	2:07	6.90	(A.McN)	HurricaneSquire Baronessa NiteCrawler	

2 — PLEASURE POINT (White, 5-1)

b h, 77, Adios Vic—Lindys Aqua Marine—Bye Bye Byrd
Albert L. Mourer, Chicora, Pa.

DAN ALTMEYER (135 lbs.) (12-13-54) Red-White-Black
175 - 28 - 22 - 18 — .264

Trainer—D. Altmeyer $14,985.00 (2, 2:02^4 ⅝)
1980 15 3 3 2 $21,433.00 (2:00^2 Mea⅝)
1979 23 12 4 4 $14,985.00 (2:02^4 Mea⅝)

6- 7 Mea⅝7	ft	Open	4800	m	30^2	1:02^3	1:31^4	2:02^2	1	4	4	2i°°	2^{½}	1^{nk}	2:02^2	*1.20	(D.Alt)	PleasurePoint RealQuick KreuBoy	69
5-31 Mea⅝8	gd	LC	12600	m	30^2	1:01^2	1:31^1	2:01	6	7	7	6°°	5^{3½}	2^1	2:01^1	1.70	(D.Alt)	Connoisseur PleasurePoint LdyNnSctt	60
5-23 Mea⅝12	ft	3yrStk	32966	m	28^2	58^1	1:28^1	1:58^2	4	x6	10	9	9^{10}	5^{1½}	1:58^3	12.50	(D.Alt)	AlbaTime JDsBuck DearStar	71
5-18aMea⅝10	sy	LC	4000	m	31^1	1:02^3	1:33^2	2:02^4	3	6	6	2°°	2^{½}	2^{¾}	2:03	3.10	(D.Alt)	Connoisseur PleasurePoint BlackBart	65
5-11aMea⅝9	gd	LC	4000	m	29^1	59^1	1:29	2:00^2	8	9	9	7°°	5^4	1^{ns}	2:00^2	11.50	(D.Alt)	PleasurePoint Connoisseur SwiftAdios	68
5- 3 Mea⅝8	ft	Open	4800	m	29^3	1:00^1	1:31	2:00^3	1	5	5	5	°4^3	dh3^{¾}	2:00^4	3.10	(D.Alt)	ParLk ShowUp PlsrePnt-dh-BlvdSkppr	68

3 — SILVER TONGUE (Blue, 6-1)

gr h, 76, Bye Bye Byrd—Dear Chunga—Storm Cloud
Betty G. Ratner, Pompano Beach, Fla.

DOUG SNYDER (155 lbs.) (7-10-51) Brown-Gold
42 - 4 - 5 - 5 — .201

Trainer—Dane Snyder $20,319.00 (3, 1:59^1 ¾)
1980 14 3 4 1 $13,050.00 (1:59^4 PPk⅝)
1979 17 4 2 2 $12,347.00 (1:59^1 VD¾)

qua

6-14 Mea⅝7	ft	Inv-hcp	7000	m	29^3	59^4	1:28^2	1:57	1	5	6	5°°	5^{10½}	5^{9¾}	1:59	19.90	(D.Sny)	TheConsigliori Connoisseur Justassmng	76
6- 7 Mea⅝7	ft	qua		m	31	1:02^3	1:31^4	2:02^3	7	1	1	1	1^{10}	1^{18¾}	2:02^3	NB	(D.Sny)	SilverTongue RaceyCasey LaredoVic	75
5-14 Mea⅝8	gd	quaDR		m	31^2	1:03^2	1:34^2	2:07^1	6	2°	2	2	2^{1½}	2^{nk}	2:07^1	NB	(DaSny)	Shadrack SilverTongue FourMe	50
5- 6 Mea⅝5	ft	quaDR		m	30^2	1:02^3	1:33^1	2:04^1	5	2	2	1°	1^4	1^{7½}	2:04^1	NB	(DaSny)	SilverTongue St.ClairIdyll PhntmImage	65
5- 2 Mea⅝6	ft	quaH-DR		m	30^4	1:02^3	1:34	2:04^1	3	4	4	4	4^{5½}	4^{4¼}	2:05	NB	(DaSny)	Justassuming JmboDeacn FlashRoger	57
3-22 PPk⅝	ft	cd	9000	m	28	57^3	1:26^2	1:56^4	6	6	6	5	5^{3¼}	4^5	2:15 4	11.50	(R.Mac)	DistantThndr WzrdAlmhrst BillyOLynch	

Time for #3 = (PPK) 2:15^2

4 — REAL QUICK (Yellow, 12-1)

b h, 70, Quick Pick—Lady Dudrock—Dudley Hanover
Vincent L. Conforte, Chesterland, Ohio

HERMAN HYLKEMA (160 lbs.) (9-2-30) Green-Gold
89 - 12 - 19 - 5 — .272

Trainer—H. Hylkema $131,120.00 (7, 1:58^3 ¾)
1980 21 2 7 2 $14,130.00 (2:00^1 Mea⅝)
1979 27 3 5 8 $24,887.00 (2:02 M^1)

6-14 Mea⅝8	ft	Open	4800	m	30^1	1:00^4	1:29^3	1:59	2	1	3	3	3^{1½}	3^2	1:59^2	9.30	(H.Hyl)	LadigoHnvr NaturalAbility RealQuick	76
6- 7 Mea⅝7	ft	Open	4800	m	30^2	1:02^3	1:31^4	2:02^2	3	2	2	3i	1^{½}	2^{nk}	2:02^2	6.50	(H.Hyl)	PleasurePoint RealQuick KreuBoy	69
5-31 Mea⅝9	gd	Open	5000	m	29^2	59^4	1:29^1	2:00	2	2	4	2	2^2	2^{1¾}	2:00^2	5.80	(H.Hyl)	KiddooSimmons RealQuick KreuBoy	60
5-23 Mea⅝8	ft	nw400ps	4200	m	29^3	1:01^2	1:31	2:00^1	6	2	2	3	2^{1½}	1^2	2:00^1	4.60	(H.Hyl)	RealQuick FloydHnvr SundustPrince	71
5-17 Mea⅝9	sy	Open	5000	m	29^1	1:00^4	1:31^3	2:02^4	2	2	3	4	6^{3½}	5^3	2:03^2	5.80	(H.Hyl)	StarRacer KaboomMinbar ArmbroTroy	56
5-10 Mea⅝6	ft	Open	4800	m	29^4	1:02	1:32^2	2:02^2	4	5	5	5°°	6^{3½}	4^{4¾}	2:03^2	5.10	(H.Hyl)	NaturalAbility KreuBoy St.ClairJulep	61

5 — NATURAL ABILITY (Brown, 3-1)

b g, 1969, Stephan Smith—Indian Pence—Scottish Pence
Jack Roseman & Jim Hodder, Pittsburgh, Pa.

DEAN ZAIMES (155 lbs.) (5-24-53) Blue-White
110 - 15 - 17 - 22 — .289

Trainer—D. Zaimes $252,827.00 (5, 1:58^4 ½)
1980 8 1 3 1 $ 7,445.00 (2:02^2 Mea⅝)
1979 27 6 11 4 $28,065.00 (2:00^2 Mea⅝)

6-14 Mea⅝8	ft	Open	4800	m	30^1	1:00^4	1:29^3	1:59	4	6	6	5	5^{3½}	2^{1¾}	1:59^2	9.00	(D.Zai)	LadigoHnvr NaturalAbility RealQuick	76
6- 6 Mea⅝7	©ft	clm250000	4200	m	29^2	1:00^3	1:30	1:59^4	7	1	2	4	3^{1½}	2^2	2:00^1	*2.10	(B.Zen)	LadigoHnvr NatralAbility ProfitLeader	74
5-24 Mea⅝6	sy	Open	4800	m	28^4	59^2	1:30^1	2:01^1	5	4	4	4	°4^{2½}	3^{2¾}	2:01^4	*1.60	(B.Zen)	DenEDoughboy IronDale NaturalAblty	62
5-17 Mea⅝6	ft	Inv-hcp	7000	m	29^2	1:00^4	1:30^1	1:59^3	1	3	4	5	4^3	4^2	2:00	22.20	(B.Zen)	DallasSpur ELVon TheConsigliori	56
5-10 Mea⅝6	ft	Open	4800	m	29^4	1:02	1:32^2	2:02^2	5	1	1	1	1^{½}	1^{1½}	2:02^2	4.30	(B.Zen)	NaturalAbility KreuBoy St.ClairJulep	61
5- 3 Mea⅝8	ft	Open	4800	m	29^3	1:00^1	1:31	2:00^3	7	2	3	4	5^{3½}	5^2	2:01	9.20	(B.Zen)	ParLk ShowUp PlsrePnt-dh-BlvdSkppr	68

6 — DEN E DOUGHBOY (Green, 6-1)

br g, 76, Breadwinner—Lady Gamecock—Gamecock
4-D Ranch & Stable Inc., Monroeville, Pa.

RICHARD WOOD (155 lbs.) (5-17-38) Green-White
103 - 22 - 13 - 10 — .316

Trainer—J. Wood $4,493.00 (3, 2:04^4 ⅝)
1980 18 3 2 1 $21,434.00 (2:00 M^1)
1979 8 5 1 1 $ 4,493.00 (2:04^4 Mea⅝)

Times for #6
M' = 2:04^3
" = 2:01^2

6-14 Mea⅝8	ft	Open	4800	m	30^1	1:00^4	1:29^3	1:59	3	4	5	4°°	4^2	4^2	1:59^2	9.30	(R.Woo)	LadigoHnvr NaturalAbility RealQuick	76
6- 7 Mea⅝7	ft	Inv	7000	m	28^4	1:00^1	1:29^3	2:00^2	1	5	5	4°°	5^3	5^{1¼}	2:00^3	6.60	(R.Woo)	Justassuming ELVon DallasSpur	69
5-31 Mea⅝6	gd	Inv	7000	m	30^1	1:01^3	1:31^1	2:00^2	5	5	5	6	6^{5½}	5^{2¼}	2:00^4	12.30	(R.Woo)	StarRacer DallasSpur ELVon	60
5-24 Mea⅝6	sy	Open	4800	m	28^4	59^2	1:30^1	2:01^1	6	5	5	5	5^3	1^{nk}	2:01^1	6.40	(R.Woo)	DenEDoughboy IronDale NaturalAblty	62
5-12 M^1	sy	cd	11000	m	30	1:00	1:30^2	2:00	8	8	7°	5°°	5^{3¾}	8^{15½}	2:03	15.10	(J.Par)	Pipeto FlyFlyMike PTBret	
5- 1 M^1	ft	cd	8500	m	29^2	58^4	1:29^3	1:59^2	8	8	9	8°	4^{3½}	5^{1¾}	1:59^4	*1.40	(J.Par)	YnkeeCmmndo MnnrtMnsht TrprtCarmel	

7 — BIG BILL M (Black, 20-1)

br g, 1974, Battleground—Linda Holmes—Rdrk. Hlms
William R. Iman, Callery, Pa.

WM. McMILLAN, Jr. (200 lbs.) (11-6-42) Gold-Red-White
51 - 7 - 7 - 10 — .279

Trainer—W. McMillan, Jr. $13,943.00 (5, 2:02^2 ⅝)
1980 7 1 2 1 $ 4,887.00 (2:05^2 Mea⅝)
1979 31 2 6 3 $ 7,876.00 (2:02^2 Mea⅝)

6-13 Mea⅝9	ft	nw400ps	4400	m	28^2	59	1:28^3	2:00	4	6	6	8	°6^{5½}	3^2	2:00^2	26.90	(McMJr)	ProfitLeader BlackBart BigBillM	66
6- 8 Mea⅝7	ft	nw425ps	4200	m	30^3	1:00^4	1:31^3	2:01^3	1	3	3	5	3^4	2^{3½}	2:02^1	16.80	(McMJr)	FloydHnvr BigBillM BelovedSkipper	52
5-31 Mea⅝9	sy	cd	3900	m	29^4	1:02^1	1:32^2	2:01^4	3	5	5	8	8^7	4^{4¾}	2:02^4	4.10	(McMJr)	BaronOfArizona Justagadbt JJsFerro	60
5-24 Mea⅝7	sy	cd	3700	m	31^1	1:00^4	1:32^3	2:03^3	5	5	5	5°°	5^5	2^{¾}	2:03^4	8.50	(McMJr)	SpeederBaron BigBillM Justagadabout	62
5-17 Mea⅝7	ft	cd	3700	m	29^1	59^1	1:29^2	2:01^4	3	5	5	5	5^5	4^{1¾}	2:02^1	22.80	(McMJr)	FloydHnvr BionicBeauty SpeederBrn	56
5-11aMea⅝7	sy	nw250ps	3200	m	30^2	1:02^4	1:32^4	2:05^2	5	6	6	6°°	4^{2½}	1^{nk}	2:05^2	11.30	(F.Gle)	BigBillM Washable CindyGiggles	68

8 — FLOYD HANOVER (Grey, 5-1)

br h, 76, Tar Heel—First Kiss—Shadow Wave
Herman J. & Josephine C. Lombardi, Houston, Pa.

RICHARD STILLINGS (145 lbs.) (3-7-44) Purple-Gold-White
265 - 52 - 53 - 40 — .358

Trainer—L. Waugh $6,467.00 (3, 2:01^4 1)
1980 6 3 1 0 $ 5,950.00 (2:01^3 Mea⅝)
1979 7 1 1 0 $ 6,247.00 (2:01^4 M^1)

dis = 2:04^2
+ 0:10
2:14^2

6-14 Mea⅝8	ft	Open	4800	m	30^1	1:00^4	1:29^3	1:59	5	7	7°	8	7^8	7^{10}	2:01	10.00	(B.Zen)	LadigoHnvr NaturalAbility RealQuick	76
6- 8 Mea⅝7	ft	nw425ps	4200	m	30^3	1:00^4	1:31^3	2:01^3	4	1°	1	1	1^{2½}	1^{3½}	2:01^3	*.80	(R.Sti)	FloydHnvr BigBillM BelovedSkipper	52
5-30 Mea⅝8	sy	nw425ps	4200	m	30^3	1:03^4	1:34^2	2:04^2	8	x8	x8	8	8dis	8dis		*1.40	(R.Sti)	LadigoHnvr SpeederBaron SndstPrince	74
5-23 Mea⅝8	ft	nw400ps	4200	m	29^3	1:01^2	1:31	2:00^1	8	1°	1	1	1^{1½}	2^2	2:00^3	2.30	(R.Sti)	RealQuick FloydHnvr SundustPrince	71
5-17 Mea⅝7	ft	cd	3700	m	29^1	59^1	1:29^2	2:01^4	6	4	2°	1°	1^2	1^{ns}	2:01^4	*.70	(R.Sti)	FloydHnvr BionicBeauty SpeederBrn	56
5- 6 Mea⅝8	ft	nw1500cd	1900	m	30^1	1:01^2	1:31^3	2:02	2	1	1	1	1^2	1^{2½}	2:02	*1.20	(R.Sti)	FloydHnvr McArthrMmntum BrnChris	59

9 — SHOW UP (Orange, 12-1)

b h, 76, Good Show—Palajean—Palachuck
Richard Lee Morgan & Deane H. Northrup, Ohio

RICHARD MORGAN (185 lbs.) (2-14-40) Red-White
88 - 15 - 8 - 18 — .289

Trainer—R. Morgan $2,688.00 (3, 2:03^2 ⅝)
1980 12 3 1 1 $ 9,690.00 (2:01 Mea⅝)
1979 4 3 0 1 $ 2,688.00 (2:03^2 Mea⅝)

6-14 Mea⅝8	ft	Open	4800	m	30^1	1:00^4	1:29^3	1:59	6	8	8	7	6^5	5^{3½}	1:59^3	50.90	(R.Mor)	LadigoHnvr NaturalAbility RealQuick	76
6- 7 Mea⅝7	ft	Open	4800	m	30^2	1:02^3	1:31^4	2:02^2	5	5	5	6i	°5^{4½}	4^{1¾}	2:02^4	25.40	(R.Mor)	PleasurePoint RealQuick KreuBoy	69
5-31 Mea⅝8	gd	LC	12600	m	30^2	1:01^2	1:31^1	2:01	4	1	2	2	3^2	5^{6½}	2:01^3	3.30	(R.Mor)	Connoisseur PleasurePoint LdyNnSctt	60
5-25aMea⅝7	ft	LC	4000	m	31^1	1:00^4	1:30^4	2:01	5	6	6	7°	7^7	6^{4½}	2:01^4	13.70	(R.Mor)	ShowUp LadyNanScott Shadrack	75
5-18aMea⅝10	sy	LC	4000	m	31^1	1:02^3	1:33^2	2:02^4	2	3	4	5	3^4	1^{1¼}	2:01	2.20	(R.Mor)	Connoisseur PleasurePoint BlackBart	65
5-11aMea⅝9	gd	LC	4000	m	29^1	59^1	1:29	2:00^2	2	1	3	5	5^{3½}	5^{4½}	2:03^3	11.90	(R.Mor)	PleasurePoint Connoisseur SwiftAdios	68

MORNING LINE SELECTIONS—5 - 1 - 2 - 8

Figure 2. Seventh race at The Meadows, 6-22-80 (left).

Figure 3. Sixth race at The Meadows, 6-22-80 (below).

PACE — 1 MILE
Purse $3,800

SIXTH RACE

Warmup Saddle Pad Color — Green

CONDITIONED — All ages. For NW $400 per start in 1979-80 or 1980. AE NW 4 pari-mutuel races life. AE NW $1600 in last 4 starts that are NW $15,000 in 1980. 3 year olds allowed 25%. 4 year olds 15%.

1	16.20	7.20	4.00
4		5.40	3.20
3			2.60

Perfecta = $92.00

PERFECTA RACE

HEAD NO. | Date | Trk | Con | Class | purse | Dis | Time ¼ ½ ¾ | Time | PP | ¼ | ½ | ¾ | Str | LgnFin | Ind Tm | Odds | Driver | ORDER OF FINISH: First | Second | Third | Temp

1 — Red — 8-1

FLYING STONE

b h, 77, Keystone Pat—Circle All—Storm Cloud

Wm. C. McMillan, Sr., Pittsburgh, Pa.

WM. McMILLAN, JR. (200 lbs.)

(11-6-42) Gold-Red-White

51 - 7 - 7 - 10 — .279

Trainer—W. McMillan, Jr. $7,213.00 (2, 2:12^4 ½)

1980 11 4 0 0 $ 7,232.00 (2:03^3 Mea⅝)

1979 20 2 6 4 $ 7,213.00 (2:12^4 Lsbn)

6-15 Mea⅝8 sy cd 3700 m 30^4 1:03^4 1:33^2 2:03^4 4 5 3 1°°1½ 1^2 2:03^4 5.00 (McMJr) FlyingStone Britz McArthurMomntum67

Convert ~~6- 5 Nfld ft 3yrStk 9980 m 29^1 1:00^3 1:31^1 2:00^4 1x 8 8 x8 8^{15} 8^{21¼}~~ 2:05 18.80 (McMJr) MarshalsStar SteadyRich BoxSeat

5-30 Mea⅝8 sy nw200ps 2900 m 29^3 1:02^2 1:33^3 2:03^3 1 4 4 1 1^{2½} 1^{2¾} 2:03^3 1.70 (McMJr) FlyingStone TomMarBill EAMomntum74

5-23 Mea⅝8 ft nw400ps 4200 m 29^3 1:01^2 1:31 2:00^1 5 8 8 8 8^{9½} 8^{10} 2:02^1 17.80 (McMJr) RealQuick FloydHnvr SundustPrince 71

5- 9 Mea⅝8 ft nw400ps 4200 m 30^1 1:02^3 1:32^4 2:02^4 6 8 8 8 7^8 4^{3½} 2:03^2 10.40 (McMJr) TrvlnMckey KahluaAndCrm VckiElctra47

5- 2 Mea⅝7 ft nw400ps 4200 m 31 1:02^1 1:32^3 2:02^1 6 6 6 7 6^6 4^{3¼} 2:02^4 24.80 (McMJr) KaboomMinbar BrnTartar DoodlesVon61

2 — White — 4-1

LADY CHRIS TIME

br m, 76, Chris Time—Miracle Bess—Miracle Knight

Olga B. Corrado, Vienna, Ohio

RON ROSS (150 lbs.)

(9-11-56) Gold-Brown

56 - 9 - 5 - 10 — .270

Trainer—J. Ross, Sr. $9,695.00 (3, 2:05^4 ⅝)

1980 18 4 1 1 $ 6,944.00 (2:03^2 Mea⅝)

1979 20 1 3 1 $ 3,009.00 (2:05^4 Mea⅝)

6-12 Mea⅝8 ft OpenF-M 4000 m 29^2 1:00^4 1:30^2 2:00 2 3 3 4 3^3 5^{4½} 2:00^4 32.50 (R.Ros) LadyNanScott WeDoHappy Quintross 63

6- 1 Mea⅝9 ft nw250ps 3400 m 29^3 1:00^3 1:30^2 2:00^4 7 8 8 6 6^{5½} 5^6 2:02 28.60 (R.Ros) RaceTimeBill ArmbroUgnda AmbroVvd69

5-22 Mea⅝7 ft nw200ps 2900 m 30 1:02^4 1:32^2 2:03^2 1 1 1 1 1^3 1^2 2:03^2 *2.00 (R.Ros) LadyChrisTime CindyGiggles Tresie 72

5-15 Mea⅝8 ft OpenF-M 4000 m 31^3 1:03^2 1:32^2 2:01^4 7 7 7 6 °7^{7½} 8^{13} 2:04^2 59.10 (R.Ros) VickiElectra WeDoHppy St.ClairJulep52

5-11aMea⅝7 sy nw250ps 3200 m 30^2 1:02^4 1:32^4 2:05^2 3 4 4 2°°2½ 4^{4½} 2:06^1 5.60 (R.Ros) BigBillM Washable CindyGiggles 68

5- 2 Mea⅝8 ft nw200ps 2900 m 30 1:03^2 1:33^4 2:04^3 5 5 5 4°°3^{1½} 1^1 2:04^3 3.10 (R.Ros) LdyChrisTime OmahaShdw MxVnSwkr61

3 — Blue — 3-1

BLACK BART

b g, 77, Adios Vic—Ada Lobell—Overtrick

Robert S. & Mary M. Hardy, Upper St. Clair, Pa.

LEWIS KEITH (155 lbs.)

(5-27-51) Red-White-Blue

222 - 45 - 29 - 33 — .325

Trainer—L. Keith $2,095.00 (2, 2:05^3 ⅝)

1980 13 3 2 2 $ 8,011.00 (2:02 Mea⅝)

1979 10 2 2 2 $ 2,095.00 (2:05^3 Mea⅝)

6-13 Mea⅝9 ft nw400ps 4400 m 28^2 59 1:28^3 2:00 7 8 8 6 °5^5 2¾ 2:00^1 24.60 (L.Kei) ProfitLeader BlackBart BigBillM 66

6- 8 Mea⅝7 ft nw425ps 4200 m 30^3 1:00^4 1:31^3 2:01^3 2 4 4 2°xx7x^{147}dis 2:10^3 6.90 (L.Kei) FloydHnvr BigBillM BelovedSkipper 52

5-31 Mea⅝8 gd LC 12600 m 30^2 1:01^2 1:31^1 2:01 8 4° 2° 4° 8^9 8^{21} 2:05^1 28.90 (L.Kei) Connoisseur PleasurePoint LdyNnSctt60

5-25aMea⅝7 ft LC 4000 m 31^1 1:00^4 1:30^4 2:01 3 4 5 4°°5^5 5^{6½} 2:02^1 *1.40 (L.Kei) ShowUp LadyNanScott Shadrack 75

5-18aMea⅝10sy LC 4000 m 31^1 1:02^3 1:33^2 2:02^4 5 7 7 4°°3^2 3^{3½} 2:03^2 9.20 (L.Kei) Connoisseur PleasurePoint BlackBart 65

5-10 Mea⅝8 ft cd 3700 m 30 1:02 1:32 2:02 1 5 3 1 1^3 1^{3½} 2:02 *2.10 (L.Kei) BlackBart BaronOfArizona SpeedrBrn61

4 — Yellow — 8-1

SHADRACK

ch h, 77, Timely Shadow—Caroga Jewel—Gene Abbe

Frank Elden Crandell, Thornville, Ohio

WM. FAHY (150 lbs.)

(5-28-54) Brown-Orange-White

212 - 39 - 25 - 24 — .287

Trainer—D. Crandell $3,815.00 (2, 2:05^1 ⅝)

1980 4 1 0 1 $ 3,398.00 (2:02^1 Mea⅝)

1979 13 4 2 0 $ 3,815.00 (2:05^1 Mea⅝)

6- 8 Mea⅝7 ft nw425ps 4200 m 30^3 1:00^4 1:31^3 2:01^3 5 6 6 6 °4^7 5^{5½} 2:02^3 3.90 (W.Fah) FloydHnvr BigBillM BelovedSkipper 52

5-31 Mea⅝8 gd LC 12600 m 30^2 1:01^2 1:31^1 2:01 4 1° 1 1 1^1 4^3 2:01^3 8.90 (W.Fah) Connoisseur PleasurePoint LdyNnSctt60

5-25aMea⅝7 ft LC 4000 m 31^1 1:00^4 1:30^4 2:01 5 2° 1 1 1^{2½} 3^{2¼} 2:01^2 5.80 (W.Fah) ShowUp LadyNanScott Shadrack /5

5-18aMea⅝9 ft nw250ps 3400 m 30^2 1:01^1 1:31 2:02^1 4 1° 2 1 1^1 1^1 2:02^1 6.50 (W.Fah) Shadrack ArmbroStoney JstaswllBrn 65

qua ~~5-14 Mea⅝8 gd qua m 31^2 1:03^2 1:34^2 2:07^1 1 4 4 4 3^{3½} 1^{nk} 2:07^1 NB (W.Fah) Shadrack SilverTongue FourMe 50~~

9-27 Mea⅝ ft cd 2000 m 29^2 1:00^1 1:30^2 2:02^3 2 4 2 3° 4^{5½} 5^{15¼} 2:05^3 21.40 (D.Cra) Bulgari DIRebellious TootsReid

5 — Brown — 9-2

SUNDUST PRINCE

b g, 74, Arrivederci Byrd—Jung Frau—Adios Boy

Beverley A. & Bud D. Foster (lessees), Fla.

BUD FOSTER (200 lbs.)

(10-31-30) Blue-White-Red

19 - 3 - 1 - 4 — .257

Trainer—Bev Foster $00.00

1980 10 2 1 4 $ 4,320.00 (2:03^4 Mea⅝)

1979 First Start

6-18 Mea⅝6 ft nw3500cd 2700 m 29^2 1:00 1:30^3 2:00^4 4 3 3 3 4^2 5^{2¼} 2:01^1 15.10 (B.Fos) FernCliffAlbie EAMmntum TownRyl 67

6-12 Mea⅝7 ft nw3500cd 2700 m 29^4 1:02 1:33^2 2:02^3 1 3 3 3 3^3 3^{¾} 2:02^4 3.50 (B.Fos) TownRoyal RchmndSpur SundstPrince63

6- 5 Mea⅝7 ft nw3500cd 2700 m 30^2 1:03 1:32 2:02^1 6 6 6 7 6^{10} 7^{12½} 2:04^3 3.70 (B.Fos) FrnCliffAlbie WnterLghtning NightMd69

5-30 Mea⅝8 sy nw425ps 4200 m 30^3 1:03^4 1:34^2 2:04^2 4 2 2 3 2^{1½} 3^2 2:04^4 8.30 (B.Fos) LadigoHnvr SpeederBaron SndstPrince74

5-23 Mea⅝8 ft nw400ps 4200 m 29^3 1:01^2 1:31 2:00^1 2 4 4 5 4^{3½} 3^{3¾} 2:01 21.50 (B.Fos) RealQuick FloydHnvr SundustPrince 71

5-15 Mea⅝8 ft nw2cd 1900 m 30^2 1:01^3 1:32^4 2:03^4 2 2 2 2 4^2 1^{1½} 2:03^4 1.90 (B.Fos) SundustPrince MissMargo JoyfulBret 52

6 — Green — 7-2

J. J.'s FERRO

b g, 76, Albatross—Valentine Dream—Knight Dream

John L. Denny, Alliance, Ohio

BILL ZENDT (170 lbs.)

(7-28-49) Blue-Gold-Red

87 - 21 - 17 - 11 — .392

Trainer—B. Zendt $6,510.00 (3, 2:03^4 ⅝)

1980 8 3 0 3 $ 5,730.00 (2:02 Mea⅝)

1979 29 2 6 3 $ 6,285.00 (2:03^4 Mea⅝)

6-15 Mea⅝8 sy cd 3700 m 30^4 1:03^4 1:33^2 2:03^4 3 4 4 5 °5^4 4^{2½} 2:04^1 *1.00 (B.Zen) FlyingStone Britz McArthurMomntum67

6- 7 Mea⅝8 ft cd 3700 m 29^1 1:01^2 1:31^1 2:02^2 6 6 6 5 °4^3 3^{1½} 2:02^3 2.80 (B.Zen) Oripahs DamonAlmahurst JJsFerro 69

5-31 Mea⅝9 sy cd 3900 m 29^4 1:02^1 1:32^2 2:01^4 9 1° 1 2 2^{1½} 3^4 2:02^3 10.00 (B.Zen) BaronOfArizona Justagadbt JJsFerro60

5-25aMea⅝9 ft nw250ps 3400 m 31 1:02^3 1:34 2:04 4 5 5 4°°1½ 1^1 2:04 *2.00 (B.Zen) JJsFerro FlyingGoozly FlashRoger 75

5-16 Mea⅝8 ft nw200ps 2900 m 30^4 1:02 1:32^1 2:02^2 1 4 4 3°°3^{1½} 1^2 2:02^2 *1.40 (B.Zen) JJsFerro FlashRoger ArmbroVivid 59

5-10 Mea⅝8 ft nw150ps 2000 m 29^1 1:01^1 1:31 2:02 5 2 2 2 2^1 1^{2¾} 2:02 2.60 (B.Zen) JJsFerro BaronAbbot JustaswellBaron61

7 — Black — 5-1

PATS COMEDIAN

b h, 77, Keystone Pat—Anna Verna—Meadow Skipper

Robert C. Beatty, Washington, Pa.

DAN ALTMEYER (135 lbs.)

(12-13-54) Red-White-Black

175 - 28 - 22 - 18 — .264

Trainer—R. Murin $8,042.00 (2, 2:02^4 ½)

1980 7 1 1 1 $ 4,681.00 (2:00^4 ScD⅝)

1979 12 2 2 0 $ 8,042.00 (2:02^4 Nfld)

ix ~~6- 5 Nfld ft 3yrStk 9680 m 30^3 1:02^3 1:32^3 2:02^4 6ix 5 6 6 5^{10} 5^{7¾}p4 2:04^2 41.90 (R.Bet) CountryPreacher BretonOak BrtsGrndSn~~

5-31 ScD⅝ ft Stk 15000 m 29^3 59^4 1:29^3 1:58^4 4 4° 4° 6° 3^{1¼} 8^{14} 2:01^3 8.10 (R.Bet) SafetySkipper EvergreenBeck SlyBeau

5-26 ScD⅝ ft 3yrStk 10250 m 28^3 57^3 1:29^3 2:00^1 8 6 8° 8°°5^{2½} 3^{7¾} 2:01^4 6.50 (R.Bet) SteadyCloud CountyPrchr PatsComedian

5-19 ScD⅝ ft nw3cd 2300 m 29^1 1:01^4 1:31^4 2:00^4 2 2 2 3 3^{1¼} 1^2 2:00^4 28.40 (R.Bet) PatsComedian SteadyRich KashTree

Convert 5-14 Leb ft 3yrStk 11403 m 30^3 1:01^3 1:32^1 2:03^1 6 6 6° 6° 6^{7½} 5^3 2:03^4 50.90 (J.Ada) SeaRanchBeach MotorHome BretonOak

5- 6 Mea⅝9 ft nw3500cd 2900 m 29^1 59^2 1:30^1 2:02^3 2 2 1 2 2½ 2^{2½} 2:03 *1.30 (D.Sny) BittASpring PatsComedian TDLine 59

MORNING LINE SELECTIONS—3 - 6 - 2

"When you begin to code your own races, you may find it simpler to use a similar table and fill in the blanks as you go along."

Race #________ Date of Race ___/___/___ Track____________________________

Post No.	Sum of Last 3 Finishes	RANK	Sum of Last 4 Times	RANK	Sum of Best 4 Finishes	RANK	KICK	NO. OF RACES RUN IN YEAR	MONEY WON THIS YEAR
1									
2									
3									
4									
5									
6									
7									
8									
9									

Note: Ignore extra post positions if fewer horses than nine are entered in the race. Only those variables whose headings are in capital letters will be keyed into the program.

Figure 4. Sample race coding form.

smallest sum) is assigned the lowest rank number. Check the examples provided for assigning ranks when scores are tied.

Note: To assist you, the calculations for the race in Figure 2 have been provided in Table 1. When you begin to code your own races, you may find it simpler to use a similar table and fill in the blanks as you go along. This will make keying in the data easier as well. A prototype form for this purpose is provided in Figure 4.

2) *Sum of the last four times.* Add the finish times for the four most recent codable races for each horse. As mentioned earlier, converting fifths of a second to tenths and expressing each time as a deviation from 2 minutes will make this easier for you. Any number *under* 2 minutes should be used as a negative time. Examples of time conversions include:

"All you need do now is run the program and enter the information for each post position."

Race #________ Date of Race ___/___/___ Track____________________

Post No.	Sum of Last 3 Finishes	RANK	Sum of Last 4 Times	RANK	Sum of Best 4 Finishes	RANK	KICK	NO. OF RACES RUN IN YEAR	MONEY WON THIS YEAR
1	17	9	6.6	6	16	8	−7	14	14225
2	8	3	5.2	4	6	2	0.5	15	21433
3	13.5	5	29.6	9	18	9	0.75	14	13050
4	7	1.5	2.4	3	8	4	−0.5	21	14130
5	7	1.5	1.4	1	8	4	1.75	8	7445
6	14	6.5	2.0	2	15	6.5	0.00	18	21434
7	9	4	9.2	7	8	4	3.50	7	4887
8	16	8	17.6	8	5	1	−2.0	6	5950
9	14	6.5	5.8	5	15	6.5	1.50	12	9690

Predicted Finish Scores: Post 1 = 5.645; Post 2 = 3.353; Post 3 = 5.434; Post 4 = 4.093; Post 5 = 2.928; Post 6 = 4.602; Post 7 = 5.396; Post 8 = 6.162; Post 9 = 6.267

Table 1. Calculations for seventh race.

$2{:}05^2 = 5.4$
$2{:}17^1 = 17.2$
$1{:}58^1 = -1.8$
$1{:}59^4 = -0.2$
$1{:}57 = -3.0$

When ranking horses on this variable, the horse with the *lowest* total time receives the lowest (smallest) rank.

3) *Sum of best four finish positions.* Add the four *best* (lowest) finish positions for each horse—again, only from codable races. Place the horses in rank order by assigning the lowest rank to the horse with the *smallest* total.

4) *Kick.* This number represents the number of lengths gained on or lost to the race leader between the stretch and the finish line in the horse's *most recent race.* Lengths behind the leader are shown as the small number written after the horse's position at the stretch and finish. If, at either of

Win Percentage Adjustments

The win percentages, variable AV(I), in lines 370–460 of the program represent appropriate figures for Windsor Raceway at the time the program was developed. However, such figures will vary from track to track and, to a lesser degree, from season to season at the same track. Therefore, to increase the accuracy of the program at your primary track, these figures should be adjusted each racing season. Quite often the percentages of wins from each post position are reported routinely in the racing form.

The proper means of calculating this figure for each post position is as follows:

1) Record the number of starts during the season from each post.

2) Record the number of wins during the season for each post.

3) Divide the result of step 2 by the result of step 1 as. . .

$$\frac{\text{number of wins}}{\text{number of starts}}$$

4) Multiply the result by 100.

Thus, if the result from step 3 is .256, when we multiply this by 100 we obtain a result of 25.6 (25.6 percent). This must be done *separately* for each post position. For example, if the maximum number of horses that your track allows in a race is nine, then you should end up with nine win percentage figures.

Next, after loading the program, type LIST 370–460. Change each of the Windsor win percentages to the new ones that you just calculated. Be certain, however, to add them in the same fashion as they were written in your program master—25.6, not 25.6 percent or .256.

If you must calculate the above figures by hand (if they are not available in the form), you may have luck requesting the information from your local track secretary. If neither of these sources can supply the necessary figures (that is, either the complete win percentages *or* the wins and starts from each post position), three options remain:

1) Use the figures that are on the master program. This will reduce the accuracy of the system, but it will effectively weight positions 1–5 higher than 6–9, which is generally appropriate.

2) Scan through some current books on harness racing. Since this tends to be a relatively important factor in most handicapping approaches, you may find one or more sources that list *U.S. average win percentages for each post position.*

3) Obtain forms for the previous season and calculate the statistics from scratch. You will have to count the number of starts and wins from each position by hand. Although much more time-consuming than the first two options, accuracy will be enhanced.

If these figures are published routinely in the form, be very certain that they are calculated as shown above. Occasionally they will be calculated by:

a) Recording the number of wins from each post position.

b) Recording the total number of races run, *regardless of post.*

c) Dividing as:

$$\frac{\text{number of wins}}{\text{number of races}}$$

4) Multiplying the result by 100 to obtain a percentage for each post.

Note that the difference occurs in step 2. Although this approach may appear correct, the error is found in the fact that there will be more races run with seven or more horses than with nine horses, for example. Let's create a hypothetical case. Let's say that at track Z there were 100 five-horse races, 100 seven-horse races, and 100 nine-horse races, for a total of 300 races during the season. Furthermore, let us assume that:

Wins In...	5-Horse Races	7-Horse Races	9-Horse Races	Total
From..				
Post 5	10	10	10	30
Post 7	—	10	10	20
Post 9	—	—	10	10

First, let's perform the calculations the *correct* way. For post 5 this is:

$$\frac{\text{number of wins} \quad (30)}{\text{number of starts} \quad (300)} \times 100 = 10.0 \text{ (or } 10\%)$$

For post 7 this is:

$$\frac{\text{number of wins} \quad (20)}{\text{number of starts} \quad (200)} \times 100 = 10.0 \text{ (or } 10\%)$$

For post 9 this is:

$$\frac{\text{number of wins} \quad (10)}{\text{number of starts} \quad (100)} \times 100 = 10.0 \text{ (or } 10\%)$$

The *incorrect* way for post 5 would yield:

$$\frac{\text{number of wins} \quad (30)}{\text{total races} \quad (300)} \times 100 = 10.0 \text{ (or } 10\%)$$

correct

For post 7 this would be:

$$\frac{\text{number of wins} \quad (20)}{\text{total races} \quad (300)} \times 100 = 6.67 \text{ (or } 6.67\%)$$

3.33% too low

For post 9 this would be:

$$\frac{\text{number of wins} \quad (10)}{\text{total races} \quad (300)} \times 100 = 3.33 \text{ (or } 3.33\%)$$

6.67% too low

As I'm sure you can see, the latter procedure for calculating win percentages *penalizes* all horses running from post 7 or 9 simply because there are more races where *at least* five horses are entered than there are races in which *at least* seven or nine horses are entered. These figures in our example would be 300, 200 and 100 races, respectively. ■

Race # ________ Date of Race ___/___/___ Track ________________

Post No.	Sum of Last 3 Finishes	RANK	Sum of Last 4 Times	RANK	Sum of Best 4 Finishes	RANK	KICK	NO. OF RACES RUN IN YEAR	MONEY WON THIS YEAR
1	11	2.5	11.8	4	10	2.5	1.50	11	7232
2	11	2.5	10.6	3	11	5	−1.5	18	6944
3	17	7	18.2	7	11	5	4.25	13	8011
4	12	4.5	7.8	2	13	7	1.50	4	3398
5	15	6	13.4	5.5	10	2.5	−.25	10	4320
6	10	1	13.4	5.5	6	1	1.50	8	5730
7	12	4.5	5.8	1	11	5	−12.75	7	4681

Predicted Finish Scores: Post 1 = 3.394; Post 2 = 4.518; Post 3 = 4.838; Post 4 = 3.582; Post 5 = 4.474; Post 6 = 4.051; Post 7 = 5.098

Table 2. Calculations for sixth race.

these times, the horse was *the* leader (indicated by a large 1), then the small number after the 1 is the number of lengths he *lead* by.

For all horses *other than the leader at the stretch or the winner*, kick is simply the number of lengths gained on or lost to the leader, as follows:

Stretch	Finish	Calculation
3^5	3^5	5 − 5 = 0
4^7	5^{12}	7 − 12 = −5
2^3	$2^{1\,1/2}$	3 − 1.5 = 1.5

For the *leader at the stretch who did not win* (for example, $1^3\ 2^2$) kick is equal to the total lengths lost. In this case he lost his original three-length lead plus two more lengths for an answer of −5.

For a *horse who won, but did not lead in the stretch* (for example, $3^3\ 1^4$), kick is equal to the total lengths gained—in this case, 3 plus 4, or 7.

For a *horse who both led in the stretch and won*, kick is equal to the number of lengths that he increased his lead (for example, $1^1\ 1^2 = 1$). If he did *not* improve his lead or it even decreased (yet he still won), score his kick as 0. We do not wish to penalize a horse/driver for running a judicious race. A win by ten lengths and a win by a nose both achieve the same result.

Finally, whenever lengths are indi-

cated as a neck, nose, etc., count the number of lengths as 0. For example, $3^{7}\ 2^{ns} 7 - 0 = 7$. If no lengths were recorded (horse distanced or did not finish), record kick as – 10.

5) *Races run this year.* Enter the number of races run by the horse in the current year.

6) *Money won this year.* Record the number of dollars won during the current year. However, for this and the previous variable, if this is the beginning of the racing year for a given horse (that is, the horse's first race of the year), you may substitute last year's figures. If you make this substitution, be *certain* that both figures come from the *same* racing year!

Adjustments to the Computer Program

One of the internal program variables concerns win percentage for each post position. To be correct for your track, consult the sidebar for a nine to ten line program modification. This update should be made at least once a year.

The Results

All you need do now is run the program and enter the information for each post position. After the last data item has been entered, the program will pause, perform the necessary calculations, and then generate the predicted finish scores sorted from first to last. Results are printed on your screen in the following manner:

THE MEADOWS 5/23/81
PREDICTED FINISH POSITION—RACE 3

PRED.FINISH	POST	FINISH SCORE
1	5	3.05238154
2	1	3.28356012
3	2	3.49912041
4	4	3.98590018
5	3	4.56078723

Calculation Practice

To get used to the method, work through the first example (Figure 2 and Table 1) and familiarize yourself with the calculation and adjustment methods. An additional sample race has been included for more practice (Figure 3). Although all necessary calculations are shown in Table 2, don't look at it until you have first tried it on your own and have run the figures through the program.

Program listing.

```
10  REM  TO BE CORRECT FOR YOUR T
     RACK, BE CERTAIN TO CHANGE '
     AV' VARIABLES TO APPROPRIATE
      POSITION WIN PERCENTAGES
20  REM  'AV' VARIABLES ARE IN LI
     NES 370-460
30  REM *************************
     ****
40  REM (C) DR. STEVEN SCHWARTZ,
     1982
50  REM *************************
     ****
60  GOSUB 1480
70  CLEAR
80  HOME : PRINT : INVERSE : HTAB
     8: PRINT "HARNESS RACE HANDI
     CAPPER"
90  VTAB 4: PRINT "  (C) STEVEN A
     . SCHWARTZ, PH.D., 1982  ":

100  PRINT "DO YOU NEED INSTRUCTI
     ONS (Y/N)? ";: GET A$: IF A$
      = "Y" THEN  GOSUB 720
110  CLEAR
120  DIM K(10),MO(10),MNR(10),KIC
     K(10),Z(10),AV(10),R3F(10),R
     4T(10),ZR(10),ZK(10),RB4F(10
     ),R80(10),PP(10),FINISH(10)
130  CLEAR
140  PRINT : PRINT : PRINT
150  PRINT "    ENTER '99' WHEN A
     SKED FOR RACE"
160  PRINT "           NUMBER TO END
      PROGRAM.": PRINT : PRINT
170  INPUT "RACE NUMBER:
        ";X$:RC =  VAL (X$): IF R
     C = 99 THEN 700
180  INPUT "DATE OF RACE:
        ";DT$: INPUT "NAME OF TRA
     CK:            ";TR$
190  INPUT "# OF HORSES IN RACE:
        ";X$:N =  VAL (X$): IF N >
     10 OR N < 2 THEN  PRINT "<<
     MAXIMUM HORSES = 10 >>": GOTO
     190
200  FOR I = 1 TO N
210  HOME : PRINT : PRINT "INPUT
     INFORMATION FOR HORSE #";I
220  PRINT "===== =========== ===
      ===== =="
230  VTAB 5: PRINT "1. POST POSIT
     ION **************  ";I:PP(I
     ) = I
240  VTAB 6: HTAB 1: CALL  - 958:
      INPUT "2. RANK SUM OF LAST
     3 FINISHES ? ";X$:R3F(I) =  VAL
     (X$)
250  IF R3F(I) < 1 OR R3F(I) > N THEN
      PRINT "<< MUST BE BETWEEN 1
      AND ";N;" >>": FOR PA = 1 TO
     2000: NEXT PA: GOTO 240
260  VTAB 7: HTAB 1: CALL  - 958:
      INPUT "3. RANK SUM OF LAST
     4 TIMES ?     ";X$:R4T(I) =  VAL
     (X$)
270  IF R4T(I) < 1 OR R4T(I) > N THEN
      PRINT "<< MUST BE BETWEEN 1
      AND ";N;" >>": FOR PA = 1 TO
     2000: NEXT PA: GOTO 260
280  VTAB 8: HTAB 1: CALL  - 958:
      INPUT "4. RANK SUM OF BEST
     4 FINISHES ? ";X$:RB4F(I) =
      VAL (X$)
290  IF RB4F(I) < 1 OR RB4F(I) >
     N THEN  PRINT "<< MUST BE BE
     TWEEN 1 AND ";N;" >>": FOR P
     A = 1 TO 2000: NEXT PA: GOTO
     280
300  VTAB 9: HTAB 1: CALL  - 958:
      INPUT "5. KICK IN LAST RACE
      ?             ";X$:K(I) =  VAL
     (X$)
310  VTAB 10: HTAB 1: CALL  - 958
     : INPUT "6. RACES RUN THIS Y
     EAR ?          ";X$:R80(I) =
      VAL (X$)
320  IF R80(I) < 0 THEN  PRINT "<
     < MINIMUM POSSIBLE IS ZERO (
     0) >>": FOR PA = 1 TO 2000: NEXT
     PA: GOTO 310
330  VTAB 11: HTAB 1: CALL  - 958
     : INPUT "7. MONEY WON THIS Y
     EAR ?          ";X$:MO(I) =  VAL
     (X$)
340  IF MO(I) < 0 THEN  PRINT "<<
      MINIMUM POSSIBLE IS ZERO (0
     ) >>": FOR PA = 1 TO 2000: NEXT
     PA: GOTO 330
350  IF R80(I) = 0 AND MO(I) <  >
     0 THEN  PRINT "<< CANNOT WIN
      MONEY WITHOUT RACING >>": FOR
     PA = 1 TO 2000: NEXT PA: GOTO
     310
360  ON I GOTO 370,380,390,400,41
     0,420,430,440,450,460
370 AV(I) = 15.8: GOTO 470: REM
     PP #1
380 AV(I) = 12.9: GOTO 470: REM
     PP #2
390 AV(I) = 13.6: GOTO 470: REM
     PP #3
400 AV(I) = 14.0: GOTO 470: REM
     PP #4
410 AV(I) = 16.3: GOTO 470: REM
     PP #5
420 AV(I) = 9.9: GOTO 470: REM  P
     P #6
430 AV(I) = 8.4: GOTO 470: REM  P
     P #7
440 AV(I) = 6.5: GOTO 470: REM  P
     P #8
450 AV(I) = 3.9: GOTO 470: REM  P
     P #9
460 AV(I) = 0.0: REM  PP #10 ---
     NO PP #10 AT WINDSOR RACEWAY
     ; HENCE, NO WEIGHT
470  IF R80(I) <  > 0 THEN 490
480 MNR(I) = 0: GOTO 500
490 MNR(I) = MO(I) / R80(I)
500 SK = SK + K(I):SR = SR + MNR(
     I):SS = SS + K(I) ^ 2:S2 = S
     2 + MNR(I) ^ 2:KICK(I) = K(I
     ):Z(I) = MNR(I)
510  NEXT I
520 MK = SK / N
530 M2 = MK ^ 2
540 QK =  SQR ( ABS ((SS - N * M2
     ) / (N - 1.0)))
550 R = SR / N
560 R2 = R ^ 2
570 QR =  SQR ( ABS ((S2 - N * R2
     ) / (N - 1.0)))
580  FOR J = 1 TO N
590 ZK(J) = (KICK(J) - MK) / QK
600  IF QR <  > 0 THEN 620
610 ZR(J) = 0: GOTO 630
620 ZR(J) = (Z(J) - R) / QR
630 FINISH(J) = 4.719629 + AV(J) *
     ( - .1419352) + R3F(J) * (.1
     171093) + R4T(J) * (.1521627
     ) + ZR(J) * ( - .305517) + Z
     K(J) * ( - .2700057) + RB4F(
     J) * (.0814399)
640  NEXT J
650 TB =  INT ((40 -  LEN (DT$ +
     "    " + TR$)) / 2) + 1
660  HOME : PRINT  TAB( TB)TR$;"
        ";DT$: PRINT : PRINT "**PR
     EDICTED FINISH POSITION - RA
     CE ";RC;"**": PRINT : PRINT

670  GOSUB 1700
680  VTAB 24: HTAB 8: INVERSE : PRINT
     "PRESS ANY KEY TO CONTINUE";
     : GET A$: NORMAL
690  HOME : GOTO 130
700  HOME : VTAB 12: PRINT "*****
     ******* END OF PROGRAM *****
     ******"
710  END
720  REM   INSTRUCTIONS
730  HOME : PRINT
740  PRINT "  THIS PROGRAM AND TH
     E EQUATION USED TO"
750  PRINT "PREDICT FINISH POSITI
     ONS ARE THE RESULT"
760  PRINT "OF EXHAUSTIVE STATIST
     ICAL ANALYSES OF A"
770  PRINT "SEASON'S RACES AT THE
      WINDSOR RACEWAY"
780  PRINT "(WINDSOR, ONTARIO). A
     LL INFORMATION IN-"
790  PRINT "CLUDED ON THE RACING
     FORM WAS EXAMINED"
800  PRINT "FOR USE AS PREDICTOR
     (INDEPENDENT)"
810  PRINT "VARIABLES. A SUBSET O
     F SIX OF THE MOST"
820  PRINT "POWERFUL PREDICTORS W
     AS CHOSEN THROUGH"
830  PRINT "THE STATISTICAL TECHN
     IQUE OF MULTIPLE"
840  PRINT "LINEAR REGRESSION AND
      IS PRESENTED IN"
850  PRINT "THIS PROGRAM."
860  PRINT : PRINT "  OVER A SEAS
     ON'S RACING, THIS EQUATION"
870  PRINT "SHOWED A PROFIT OF 20
```

Listing continued.

On Betting

Although tempting, we suggest that you do *not* bet the top three picks of the system as a Trifecta/Traitor combination—that is, the bet where you are asked to pick the win, place and show horses in a race. Even though you will often find that two of the system's top three picks are within the money winners, you will seldom find all three. Perhaps this accounts for the generally large pay-offs in Trifectas—they are difficult to predict.

Perfecta, Exacta, Exactor and Quinella bets (win/place combinations in a single race), on the other hand, seem to be quite acceptable with the system. As an example, the system successfully selected the Perfecta combination (Figure 3 and Table 2) *in order* for a pay-off of $92 on a $2 bet! (Note: both races included in this article were selected at random from a Meadows racing program.)

The 20 percent *long-run* profit mentioned in the program (which was initially achieved by the system) was obtained by consistently betting the system's top pick to win. We suggest that you try the system out at home a bit before venturing out to the track and the betting windows. Purchase forms for a few days, code all bettable races (see the program for additional tips), and check the next day's paper for the results. When you're ready, bet the system . . . *don't play hunches*! (If your hunches were that accurate, you wouldn't need a system, now would you?)

Finally, just to excite you a bit, let me tell a little story. Shortly after the system was developed, a friend and I were testing it at the Windsor Raceway. The final race of the evening took place in a blizzard and the system's first and second picks had incredibly high odds on the tote board. Therefore we decided to sit out the storm. When the race was over, the winner paid approximately 120 to 1 and the place horse about 60 to 1. Both horses were correctly picked by the system—in order! Coupled with a third horse, this race represented the largest Triactor pay-off in the history of the track. $35,000 was split between two lucky patrons who held $6 box tickets on the combination.

Good luck! ■

Listing continued.

```
     % BY CONSISTENTLY"
880  PRINT "BETTING THE TOP PICK;
      I.E., THE HORSE"
890  PRINT "WITH THE LOWEST PREDI
     CTED FINISH POSI-"
900  PRINT "TION. ALTHOUGH THE TO
     P PICK IS NOT AL-"
910  PRINT "WAYS THE WINNER, EXPE
     RIENCE HAS SHOWN"
920  PRINT "THAT ONE OF THE THREE
      TOP PICKS TYPI-"
930  PRINT "CALLY IS!"
940  VTAB 24: INVERSE : PRINT "PR
     ESS ANY KEY TO CONTINUE";: GET
      A$: NORMAL : HOME : PRINT
950  PRINT "FOR MAXIMUM EFFICIENC
     Y, CLOSELY HEED "
960  PRINT "THE FOLLOWING SUGGEST
     IONS:"
970  PRINT : PRINT "1. CHANGE THE
      PERCENTAGES IN LINES 370"
980  PRINT "   THROUGH 460 (VARIA
     BLE 'AV'). THESE"
990  PRINT "   ARE THE WIN PERCEN
     TAGES FOR EACH "
1000  PRINT "   POST POSITION (1
     THRU 9--IN ORDER) "
1010  PRINT "   AT WINDSOR RACEWA
     Y. TO INCREASE ACC-"
1020  PRINT "   URACY, REPLACE TH
     EM WITH THE ONES "
1030  PRINT "   FOR YOUR RACE TRA
     CK. IT IS SIMPLEST"
1040  PRINT "   TO USE THE FIGURE
     S FROM THE LAST DAY"
1050  PRINT "   OF THE END OF THE
      MOST RECENT SEASON"
1060  PRINT "   OR, IF THE PRESEN
     T SEASON HAS BEEN"
1070  PRINT "   ON FOR SEVERAL MO
     NTHS, USE CURRENT"
1080  PRINT "   FIGURES. <WINS BY
      POST POSITION ARE"
1090  PRINT "   GENERALLY AVAILAB
     LE ON THE FORM>"
1100  PRINT : PRINT "   THE PROGR
     AM PROVIDES FOR A MAX. OF 10
        HORSES PER RACE, EVEN THO
     UGH WINDSOR    ONLY ALLOWS U
     P TO NINE HORSES."
1110  VTAB 24: INVERSE : PRINT "P
     RESS ANY KEY TO CONTINUE";: GET
      A$: NORMAL : HOME : PRINT
1120  PRINT "2. DO NOT HANDICAP T
     ROTTERS, MAIDEN"
1130  PRINT "   RACES (OR RACES F
     OR TWO YEAR OLDS),"
1140  PRINT "   OR CHEAP CLAIMERS
      (WHERE HORSES MAY"
1150  PRINT "   BE CLAIMED FOR LE
     SS THAN $5,000)."
1160  PRINT "   HORSES IN SUCH RA
     CES TEND TO BE VERY"
1170  PRINT "   INCONSISTENT AND,
      HENCE, DIFFICULT"
1180  PRINT "   TO HANDICAP.": PRINT
1190  PRINT "3. CONVERSELY, THIS
     SYSTEM HAS SHOWN"
1200  PRINT "   GREATEST SUCCESS
     WITH HIGHER PRICED"
1210  PRINT "   CLAIMERS; I.E., $
      10,000 - $20,000."
1220  PRINT : PRINT "4. ONE OF TH
     E SYSTEM'S TOP THREE PICKS"
1230  PRINT "   IS TYPICALLY THE
     WINNER. THUS, THE"
1240  PRINT "   SYSTEM EFFECTIVEL
     Y ELIMINATES THE"
1250  PRINT "   HORSES WITH A HIG
     H POTENTIAL FOR "
1260  PRINT "   LOSING. HOWEVER,
     SELECTING AMONG THE"
1270  PRINT "   TOP THREE PICKS I
     S UP TO YOU."
1280  VTAB 24: INVERSE : PRINT "P
     RESS ANY KEY TO CONTINUE";: GET
      A$: NORMAL : HOME : PRINT
1290  PRINT "**** HINT ****": PRINT

1300  PRINT "   THE CLOSER THE PR
     EDICTED FINISH"
1310  PRINT "   SCORES ARE FOR TH
     E TOP THREE, THE"
1320  PRINT "   MORE NEARLY IDENT
     ICAL ARE THEIR"
1330  PRINT "   CHANCES OF WINNIN
     G--LOOK FOR LARGE"
1340  PRINT "   SPREADS!"
1350  PRINT : PRINT "5. THE PROGR
     AM CALCULATES THE PREDICTED"

1360  PRINT "   FINISH POSITION F
     OR EACH HORSE AND"
1370  PRINT "   PRINTS THEM IN OR
     DER (1ST, 2ND, --)."
1380  PRINT "   THE LOWER THE NUM
     BER, THE MORE LIKE-"
1390  PRINT "   LY THE HORSE WILL
      WIN (OR FINISH IN"
1400  PRINT "   THE MONEY). "
1410  PRINT : PRINT "6. IN HANDIC
     APPING THE RACES, THE MORE"
1420  PRINT "   CURRENT AND COMPL
     ETE THE INFORMATION"
1430  PRINT "   FOR EACH HORSE--T
     HE MORE ACCURATE "
1440  PRINT "   THE PREDICTIONS.
     EARLY IN EACH SEAS-"
1450  PRINT "   ON, NEITHER OF TH
     ESE CONDITIONS TYP-"
1460  PRINT "   ICALLY EXIST. BEW
     ARE!"
1470  VTAB 24: INVERSE : PRINT "P
     RESS ANY KEY TO BEGIN PROGRA
     M";: GET A$: NORMAL : HOME :
      PRINT : RETURN
1480  REM  GRAPHICS INTRODUCTION
1490  HOME : GR
1500  COLOR= 12
1510  FOR I = 0 TO 39
1520  HLIN 0,39 AT I
1530  NEXT I
1540  COLOR= 0
1550  PLOT 3,6: PLOT 6,6: HLIN 8,
     11 AT 6: HLIN 13,16 AT 6: HLIN
      18,21 AT 6: HLIN 23,26 AT 6:
      HLIN 28,31 AT 6: HLIN 33,36
      AT 6
1560  PLOT 3,7: PLOT 6,7: PLOT 8,
     7: PLOT 11,7: PLOT 13,7: PLOT
     16,7: PLOT 18,7: PLOT 21,7: PLOT
     23,7: PLOT 28,7: PLOT 33,7
1570  HLIN 3,6 AT 8: HLIN 8,11 AT
     8: HLIN 13,15 AT 8: PLOT 18,
     8: PLOT 21,8: HLIN 23,25 AT
     8: HLIN 28,31 AT 8: HLIN 33,
     36 AT 8
1580  PLOT 3,9: PLOT 6,9: PLOT 8,
     9: PLOT 11,9: PLOT 13,9: PLOT
     16,9: PLOT 18,9: PLOT 21,9: PLOT
     23,9: PLOT 31,9: PLOT 36,9
1590  PLOT 3,10: PLOT 6,10: PLOT
     8,10: PLOT 11,10: PLOT 13,10
     : PLOT 16,10: PLOT 18,10: PLOT
     21,10: HLIN 23,26 AT 10: HLIN
     28,31 AT 10: HLIN 33,36 AT 1
     0
1600  HLIN 10,13 AT 18: HLIN 15,1
     8 AT 18: HLIN 20,23 AT 18: HLIN
     25,28 AT 18
1610  PLOT 10,19: PLOT 13,19: PLOT
     15,19: PLOT 18,19: PLOT 20,1
     9: PLOT 23,19: PLOT 25,19
1620  HLIN 10,12 AT 20: HLIN 15,1
     8 AT 20: PLOT 20,20: HLIN 25
     ,27 AT 20
1630  PLOT 10,21: PLOT 13,21: PLOT
     15,21: PLOT 18,21: PLOT 20,2
     1: PLOT 23,21: PLOT 25,21
1640  PLOT 10,22: PLOT 13,22: PLOT
     15,22: PLOT 18,22: HLIN 20,2
     3 AT 22: HLIN 25,28 AT 22
1650  FOR PAUSE = 1 TO 3000: NEXT
     PAUSE
1660  FOR Y = 0 TO 39
1670  HLIN 0,39 AT Y
1680  NEXT Y
1690  TEXT : RETURN
1700  REM  SORT BY PREDICTED FINI
     SH POSITION
1710  FOR J = 1 TO N - 1
1720 L = J
1730  FOR K = J + 1 TO N
1740  IF FINISH(L) < FINISH(K) THEN
     1760
1750 L = K
1760  NEXT K
1770 TEMP = FINISH(L):T1 = PP(L)
1780 FINISH(L) = FINISH(J):PP(L) =
     PP(J)
1790 FINISH(J) = TEMP:PP(J) = T1
1800  NEXT J
1810  REM  PRINT RESULTS
1820  INVERSE : PRINT "PRED. FINI
     SH";: NORMAL
1830  HTAB 18: INVERSE : PRINT "P
     OST";: NORMAL : PRINT "
     ";
1840  HTAB 27: INVERSE : PRINT "F
     INISH SCORE": NORMAL : PRINT
     : PRINT
1850  FOR I = 1 TO N
1860  PRINT  TAB( 6)I; TAB( 19)PP
     (I); TAB( 27)FINISH(I)
1870  NEXT I
1880  PRINT : FOR I = 1 TO 40: PRINT
     "-";: NEXT I
1890  RETURN
```

The Power Intensifies

Our readers demand more info on the UCSD p-System and how they can write software that runs on 40 different computers!

by R. John Buczek

My article "The Power" on the UCSD p-System in the January *inCider* brought in a lot of mail to which I would like to respond. Most of the inquiries can be condensed into four general questions. The first two can be answered in a few paragraphs; I'll devote the remainder of the article to detailed answers to the third and fourth questions.

How do I join the UCSD p-System users group?

I wish they were all this easy. Just send $20 per year to:

Chip Chapin, Secretary
USUS
PO Box 1148
La Jolla, CA 92038

The twenty dollars entitles you to the *USUS News and Report*, which usually comes out quarterly, although since the editorial staff is all volunteer it is sometimes late. Also thrown in is a listing of your name and address in the *International Directory*, and the opportunity to keep up on what's happening. Most of all you get to express your opinions about the products and policies of Softech, Apple and others. They all have permanent liaison representatives to the USUS group and will at least listen to your comments. I highly recommend joining. The present membership numbers about 1500, heavily weighted toward system people with philosophical interests. There's nothing wrong with that, but I think that the interests of end users need a little more representation.

Will the p-System allow me to run a 317 user network on a Sinclair 1000?

Most of the inquiries of this type are unanswerable because they are too broad. A reasonable response would require much more information than people gave me or more time than I can afford. All I can say here is that the p-System does not suddenly endow your micro with the abilities of an IBM 370, nor does it magically make a hundred-thousand-dollar custom programming job into a thousand-dollar job.

I would suggest, once again, that you join the USUS and try to find someone near you in the *Directory*. When you send a membership check in to Mr. Chapin, ask him to tell you how to come up on the USUS net. You can hang your questions on a network bulletin board (taking care to formulate them carefully and to include all pertinent information) and see if anyone is interested enough to help (no guarantee). If no one volunteers, offer money. I'll be glad to send my fee schedule to anyone upon request.

If the p-System is so great, how come the [graphics, word processing, database, etc.] program I wrote runs [faster, slower, bigger, smaller, etc.] than under the [Apple or Radio Shack, CP/M, PCDOS, XENEX, etc.] version of [Pascal, Basic, Fortran, Algol, Forth, Lisp, etc.]?

Please understand that I make no claim that the p-System is the be-all and end-all of software systems. I claim only that its performance is usually equal to or better than the majority of systems on the market.

Admittedly there are some problems. At the time I wrote "Power" (July '82) Softech had just announced version IV.0 for the Apple and I hadn't seen it, although I had used IV.0 on other machines. I am now aware that the presently (January '83) available version of IV.0 for the Apple II is ridiculously slow, and uses more RAM than does Apple's (II.1) version. An associate recently benchmarked some simple test programs as 300 percent slower than Applesoft Basic. At this time I don't recommend that anyone purchase Softech's version IV.0.

All is not lost, however. Good old Apple Pascal II.1 does a fine job, and any program written under Apple Pascal (remember, it's still UCSD) can be recompiled under IV.0. The extra IV.0 features will be available for Apple users soon, probably by the

Address correspondence to R. John Buczek, PO Box 893, LaGrande, OR 97850.

time this article is in your hands. There is some hot gossip going around that a number of p-System licensees are now in the final stages of a complete rewrite of the Apple SYSTEM INTERP that speeds Apple IV.0 up a lot, allows you to use add-on memory as RAM disks and, on the theory that if they're going to go to all that trouble anyway why waste the opportunity, probably more new features.

Keep in mind that unlike any other system that I know of, the p-System is not dependent on just one vendor. It is, therefore, a growing and evolving creature, with sufficient flexibility to quickly respond to problems and often surprise us with new or improved features. The implementations for other machines are okay, especially those based on 8088, 8086 and M68000 chips. So, look at the forest and have faith.

Incidentally, you might wonder, as I did, why a supposedly forward-thinking outfit like Softech would release such a dog as the current IV.0 for Apple. The answer boils down to money. It seems that Apple got their UCSD license from the University of California before the state of California forced the University to turn commercial distribution over to Softech. Therefore, Softech makes next to nothing on each copy of the Apple Pascal II.1 system sold, and Apple makes nothing at all on copies of Softech's newer versions sold for Apple computers. Consequently, nobody cooperates, and nobody puts in much effort. It's much more profitable for Softech to invest its efforts in the versions for the Osborne or IBM PC. Since there is a good Apple market out there, however, other p-System licensed vendors are making the necessary improvements.

Back to the "how come" questions. A number of people said that portability is great for commercial software distributors, but why should a little guy invest the extra money when he can use another DOS that's easier and cheaper now. This was the entire thrust of my last piece, but let me try once more, briefly, to state my point.

Five years ago micros were brand new. They existed totally by accident (but that's another story). Considering their cost in comparison to any-

```
UNIT <some name>;
INTERFACE

     TYPE A_SET = [X,Y,Z];
          SIGNS = [+,-];
          DEGREES = 1..360;

PROCEDURE ROTATE (AXIS:A_SET;DIR:SIGN;AMOUNT:DEGREES);

------------------------------------------------------------------------------------

IMPLEMENTATION

PROCEDURE ROTATE;
BEGIN
     .
     .
     <programming as necessary>
     .
     .
END;

BEGIN
END.
```

Listing 1.
Library entry format.

thing on the market at the time, the fact that these little machines could do anything at all was so amazing that no one's expectations were very high. The average operating system at that time had around a half a man-year of effort invested in it and the earliest applications programs marketed often represented a few man-weeks' work.

We've come through several generations of development now. More than 60 percent of the companies in our industry that have reached a multimillion dollar sales volume are already out of business. The hobby people are out and the new market is everybody, and our expectations are high and getting higher every day. Today's newest operating systems represent many man-years of effort and most applications programs are equivalent. We can still afford these systems because the sales volume is high enough to permit recovery of development costs.

The newest processors, especially the 32-bit processors now in prototype, are really amazing. They have capacities that cannot be equaled by "old school" technology. Up until now operating systems have, at least in part, been inspired by existing mini and mainframe software. With new processors that can do more than those old machines, developers will have to break new ground at significantly higher cost. These new operating systems may require hundreds of man-years of development, and new applications programs that fully exploit all the new features may require scores. If this work is duplicated a dozen times for a dozen different systems, the divided volume will not be enough to get the price down to the level that everyone can afford.

We are going to have to settle down and support a small number of less-than-perfect software systems now, to get much greater improvements tomorrow. Three years ago Adam Osborne predicted that within a decade or so one "super chip" would capture 95 percent of the market, controlling everything from microwave ovens to supermini computers, depending on how much of the capability was used. There's every chance that economic forces may do the same for operating systems. If you could have only one operating system, what would *you* want? You can influence that decision by what you support now.

Another variation on the "how come" theme is: Why fool with a complicated, memory-intensive pseudo-system? What's wrong with good ol' CP/M? It's portable isn't it?

I realize that there is no answer to this question that will please everyone. Please think about my reasoning before you jump down my throat.

First of all, CP/M is processor ori-

"In any system you can write such a routine and copy it into any future program, but with the p-System you can put it into the language permanently."

ented. There is no guarantee that any one processor will be around for more than a few more years and a system that requires redesign from the bottom up, while attempting to carry old familiar features that may not necessarily be pertinent to new equipment, can hardly be described as portable. Most CP/M systems use the Z80 chip, but most CP/Ms only give you the good old 8080 ASM.COM assembler. It'll work, since the Z80 accepts the 8080 codes as a subset, but it wastes most of the added capability you've paid for.

There is a philosophical problem, too. CP/M was originally designed/defined to be a lowest common denominator, supporting the minimal systems then on the market. There are no standards for new features such as graphics, audio, control, etc. Therefore, every hardware OEM offers "extended CP/M" with their own extensions to handle these features. These extensions conform to no standards, and effectively create a dozen or more custom CP/Ms with no portability at all. Any program written under one variation must be extensively rewritten for another.

In the p-System, "standards" is the keyword. We already have standards for most of these new features and the people involved are busy keeping ahead of new developments. In addition, we have easy methods for implementing many new custom hardware features at the applications level, not at the system level. An example is the modem communications standard that USUS created and formally recognized last year.

This is, of course, an over-simplification of an exceedingly complex issue, but I think the major points are valid.

With regard to the "frames" structure briefly mentioned in the January article—Huh??

If you have ever written a program in any language, how many times have you said, "Boy, if I had written this language I'd certainly have done such-and-such differently." Experienced programmers have long known that they can effectively do things differently by accumulating an extensive series of subroutines for operations such as screen control, disk ops, printer control, error handling, etc, that work just the way they want. The advantage of the p-System SYSTEM LIBRARY facility is that, having written such subroutines, you can effectively make them a permanent part of the programming languages applicable to any future programming.

Let's try an example. This will be in Pascal since few people work in Fortran and I firmly discourage anyone from doing anything in Basic. The system does work the same way with any of the languages (now including Lisp, Forth, and Fermat, and soon to include ADA).

Everyone seems to be interested in graphics, so let's say that you want to be able to rotate an image around any axis. To do this you need a new subroutine:

```
Procedure Rotate(axis:a_set;direction:sign;
amount: degrees)
```

where you have previously defined:

```
TYPE A_SET = [X,Y,Z] i.e. a set containing only x,y,z
  SIGN = [+,-]
  DEGREES = 1..360 i.e. an integer in that range
```

With this subroutine, you simply specify which axis you want as the center of rotation (X,Y or Z), which direction (+ or −), and how many degrees of rotation.

Now, of course, in any system you can write such a routine and copy it into any future program, but with the p-System you can put it into the language permanently.

The format for a library entry is demonstrated in Listing 1. The first part, the INTERFACE, is the communications bridge between a program using the new command and the library. It may seem redundant, but it is this feature that permits the

"You need only replace the old SYSTEM.LIBRARY entry with the new one and all existing programming will still work."

```
Unit <some name>;
INTERFACE

        TYPE A_SET = [X,Y,Z];
             SIGNS = [+, -];
             DEGREES = 1..360;

PROCEDURE ROTATE (AXIS:A_SET;DIR:SIGN;AMOUNT:DEGREES);
----------------------------------------------------------------------
IMPLEMENTATION

PROCEDURE ROTATE;
BEGIN
        .
        <programming as necessary>
        .
END;

PROCEDURE DECODE(LISC:INTEGER);
BEGIN
        .
        <routine to decode registered owners name from license number, or coded entry in
SYSTEM.MISCINFO. Returns name in global string S>
        .
END;

FUNCTION
CHECK_CURRENT_PASSWORD(ATTEMPTED:STRING):BOOLEAN
BEGIN
        .
        <routine that checks password entered against variable password calculated from
        date. Software owner informs user of new password by postcard each January or by
        phone if identification is proper>
        .
END;

BEGIN
        PAGE(OUTPUT);
        WRITELN('THIS GRAPHICS UNIT BY:');
        WRITELN('WAYNE GREEN SOFTWARE');
        WRITELN('LICENSED ONLY TO:');
        DECODE(LICENSE);
        WRITELN(S); (*S is name returned from DECODE*)
        WRITELN('ENTER PASSWORD FOR CURRENT YEAR');
        READLN(S);
        IF NOT(CHECK_CURRENT_PASSWORD(S)) THEN EXIT
END.
```

Listing 2. Inserting a password.

second part, the IMPLEMENTATION, to be written in another language or even in assembly language. The BEGIN END at the end is a dummy to keep the compiler happy, but if you wish to insert a commercial or password here, it will run once only, when the calling program first boots and links up the library. See Listing 2.

Let's say that you create this instruction and install it in your SYSTEM.LIBRARY, then proceed to write dozens of programs all using the new command. After some time you decide that the existing command is just too slow, so you rewrite the command IMPLEMENTATION in assembly language. As long as you don't change any part of the INTERFACE, you need only replace the old SYSTEM.LIBRARY entry with the new one and all existing programming will still work as before (only faster).

Listing 3.
Example of a frame.

```
PROGRAM STATISTICS;
USES STATS;   (* <--- this hooks in the library *)
VAR M,V,S:REAL;
      DATA:LIST_OF_INFO; (* defined in STATS*)
      ANSWER:CHAR;

BEGIN
      REPEAT
            DATA : = COLLECT_DATA;
            M : = MEAN;
            V : = VARIANCE;
            S : = STANDARD_DEVIATION;
            REPORT_DATA(M,V,S);
            WRITE('ARE YOU DONE ? <Y,N>');
            READLN(ANSWER);
      UNTIL ANSWER = 'Y';
END.
```

The IMPLEMENTATION can be in p-code, compiled code or assembly language, or be changed from one to another, without affecting the programs using the command. The data types A_SET, SIGN, and DEGREES also become permanent parts of the language.

Now let's consider how we could use this capability to create what I call a *frame*. Say you've written a hot new statistical package that collects data and then calculates mean, variance and standard deviation, and reports these parameters on the screen. Obviously, such a program would be of use to lots of customers, but how about the customer who wants to use it on an existing database of 20,000 records. Okay, you say, I'll add the ability to accept data from a file. But what format do you expect the data to be in. If you make one choice and the customer's data is in another, he must write a program to reformat his data, *if* he has room. You could write a general purpose file reader, but this might take twice as much effort as the original program. Instead, let's do it this way.

Compile and install your procedures (and functions) MEAN, VARIANCE, STANDARD_DEVIATION, COLLECT_DATA and REPORT_DATA into a library unit called STATS.

Now sell your clients the library code, and the program code and source listed in Listing 3. This will work exactly like your original program, but if your client wishes he can ignore your command COLLECT_DATA and write his own, maybe called GET_DATA, exactly configured as he needs. Maybe he wants to write a more extensive program that automatically calculates these parameters on all data, but only reports if certain limits are exceeded. No problem.

This program is what I call a *frame*. It protects your software source code, while permitting your customer to modify the way he applies it to his problems. The beauty of the frame is that you can include a commercial in the IMPLEMENTATION, as in Listing 2, that will show up every time the program is booted. Someone might rip off a copy of your library, but it could be made very difficult to remove the commercial, thereby letting every user know where the material came from.

The same commands could be called from a program written in p-Basic or p-Fortran giving your customer even more flexibility. If your customer uses this library to write a 10,000-line program and then you rewrite the library to work faster or with more precision, there is still no problem; the customer's program will continue to work.

Don't forget portability. You can write this program on an Apple, and sell it for an IBM PC or an Osborne I, or any other machine for which UCSD IV.0 (or later) is available.

I hope this follow-up meets most of your needs. If you still have questions, drop me a line and I'll try to be helpful. I would also like to hear complaints about the system, especially bugs, and I'll pass them on to USUS. ■

Super-Text 40-56-70

Simplified Version of Super-Text 40/80: Bargain or Blunder?

by Leslie R. Schmeltz

Word processing programs for the Apple II and other personal computers have followed an interesting course of evolution. First, limited feature programs were introduced and rapidly fell by the wayside as newer, more efficient versions became available. The list of features offered by word processing programs has grown like topsy, as have the price tags. Recently, producers of word processing programs have realized there is a large potential market for programs with fewer features and proportionately smaller price tags. A few manufacturers have elected to create special versions of their word processing systems to fill this void in the market.

The amateur versions of several popular word processing systems have incorporated an impressive array of features while emphasizing user friendliness. Easy Writer, Personal Secretary and Super-Text 40-56-70 are prime examples of programs derived from their more professional counterparts—Easy Writer Professional, Executive Secretary and Super-Text 40/80. This review will take a close look at Super-Text 40-56-70 and, of course, compare it to Super-Text 40/80 that was reviewed in this magazine a short time ago (Jan. 83).

System Requirements

Super-Text 40-56-70 requires an Apple II or Apple II Plus with 48K and at least one disk drive. No further hardware is necessary to utilize either the upper/lowercase or various screen column-width options in the program. If your Apple has not yet been modified for operation of the shift key, Muse Software has provided the necessary wire and instructions to handle the modification.

Documentation

The documentation accompanying Super-Text 40-56-70 is excellent. The basic format is exactly like that furnished with other versions of Super-Text—a loose-leaf binder divided into functional sections. Explanations are, in many cases, the same as those found in the Super-Text 40/80 manual. Where these manuals differ, the information contained in the Super-Text 40-56-70 manual is geared more toward the inexperienced user.

Interfacing

Interfacing Super-Text 40-56-70 is exactly the same as Super-Text 40/80. Since the display formats are in software, no particular problems should be encountered here. Communicating with your printer is simply a matter of following the completely explicit instructions in the program manual. Interfacing the program to your printer is a matter of selecting the proper ΛParams file from the group offered: Epson MX-80 or MX-80 with Graftrax, Diablo compatible, Centronics Parallel (using the standard Apple card), NEC and IDS. Control character sequences and default parameters for printing are easily set or modified by anyone, regardless of the level of computer sophistication.

Operations and Features

Four modes of operation provide access to all the capabilities of the system. Rather than dwell on a lengthy description of each, let me just hit the high points.

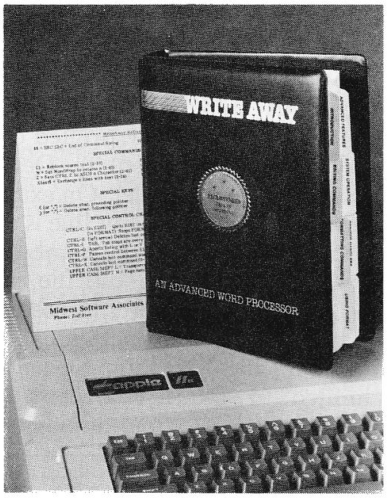
Cursor Mode

Cursor mode is used for loading and saving files, scrolling text, deleting text, find and replace and block operations. The file system is used to provide the display mode, as we will discuss in more detail later. The disk catalog display shows which slot and drive is being accessed, which file is currently in memory, those files changed since the last save, free space remaining on the disk, and the operation currently in progress. Of course, the names of the files contained on the disk are also displayed! A new feature allows the user

Address correspondence to Leslie R. Schmeltz, 3224 Magnolia Court, Bettendorf, IA 52722.

NOW, USE ALL THE POWER OF THE APPLE *IIe* FOR FAST, EASY WORD PROCESSING

Write Away is a new, full-function word processing system designed specifically for the Apple *IIe*.



It will also run faster than any other word processor when using 48K or 64K Apple *II* systems.

Write Away is ideal for both the advanced professional or novice user. It is fast, versatile, powerful and easy to use . . . Just what you need to produce *any type of document*. Write Away has *all* of the advanced features you would expect from a professional word processing system.

Write Away is easy to learn and use . . . the manual is organized by chapters, and has a complete index as well as reference sections, in which each command is carefully explained with examples. Write away has screen tutorials, a "Help" screen and a handy stand-up command reference card. You'll also use single keystrokes for deleting, searching, inserting, cursor movement and other commands. Write Away allows conditional text ("If . . . Then" text selection).

It's fast . . . Write Away uses S&H's TDE ("the DOS enhancer") for fast system loading. Write Away programs are written in 6502 machine language using advanced programming techniques. As a result, commands are executed and files are loaded with amazing speed.

Extra features . . . Write Away uses standard Apple DOS text files for easy interfacing with popular programs . . . VisiCalc, Sensible Speller and more. It also has a DIF file translator and other utilities for creating a mailing list from DB Master and other data bases . . . mail merge system included.

Automatically utilizes RAM card, upper/lower case and recognizes Videx, Smarterm, Vision 80, Wizard 80, Sup 'R Term, Full View 80, Viewmax-80 and both Apple *IIe* 80-column boards. Write Away will work with any printer.

Awarded the recommended seal of approval by the Professional Software Programmers Association.

ONLY

$175

WRITE AWAY

Midwest Software Associates
P.O. Box 301 St. Ann, MO 63074
Call toll free 1-800-835-2246/Ext. 467
(In Missouri 1-800-362-2421/Ext. 467)

to select whether binary or text files will be saved, providing the means to use files with other programs. DOS 3.2 is used in this mode, but files may be muffined to DOS 3.3 if desired.

Cursor and text movement are also handled in the cursor mode. Easy to use commands provide character, word, line, video screen or entire file. Find and Replace, a really strong point in all previous Super-Text versions, provides all the goodies we expect in a quality word processing program. Wild card characters, space matching and multiple locate and replace are all present. Replacement occurrence and word count are also provided.

A full set of block operation capabilities are included in Super-Text 40-56-70. After you have marked a block of text (using control-V), it may be saved, copied, moved, deleted or unmarked. Change markers may be inserted into the text to identify the location of the last change made to the file in memory.

System Query, accessed from Cursor mode, provides selection of various options and current status of the system. Autolink (a method of constructing and utilizing linked files) On or Off, status of The Key (a method of defining a character or series of characters automatically printed in the text when the colon is pressed), number of occurrences replaced in the last Replace operation, and Print status are displayed. Print status may be Form (continuous paper), Sheet (single sheet), Preview (preview printed output on the video screen) or Design (character design module). On exiting System Query, a File Status line is displayed. This line shows the number of characters currently in the file, the name of the file currently in use and the number of characters that can be added to the file. Maximum file length is limited to 14904 characters, at which point the program displays the **MEMORY FULL** prompt.

Add Mode

Add mode is used to create a new file or add text to one that already exists. This mode is entered by pressing control-A and exited by pressing escape-escape. If you have modified your Apple, the shift keys will control the selection of upper and lowercase letters. A type ahead buffer is provided so the program will keep up with all but the fastest typists, but see the Impressions section for more on this.

Add mode is also used to enter formatting information in a file, set tabs, define headers and footers and numerous other functions incorporated into this program.

Change Mode

Change mode is used to change a character at the cursor to one from the keyboard. An echo sound verifies the fact that a change has indeed been made. Cursor movement capabilities are active in this mode, so multiple changes can be made in a file without having to exit and reenter the mode.

> "Although there are many users who really like the alternate character sets and are willing to live with the speed limitations, my preference lies elsewhere."

Print Mode

Print mode provides for either hard copy on your printer or preview material on the video display. You may print multiple files or mulitple copies of a file using either continuous or single sheet paper. Print may be interrupted or terminated at any point. Unlike other versions of Super-Text, this program does provide video display during print operations. Lines are shown on the screen according to the method of display you have selected.

Autolink printing provides the means to automatically sequence a number of files for printing or previewing operations. It is not possible, however, to print multiple copies of linked files if the Autolink feature is in use.

Display Options

Super-Text 40-56-70 is billed as a program that allows the user to select 40, 56, or 70 character lines (hence the name) for display with no additional hardware required. Many programs now on the market have gone a similar route, sporting displays that range between 40 and 70 characters. The selection process in Super-Text 40-56-70 is quite interesting—the display file is loaded from the disk in exactly the same fashion as any other file (control-L) and may be changed at any time. Text in each file is not formatted when saved and it adapts itself readily to changes in display mode.

Muse Software has provided the three character sets in software. You may, if desired, design and save your own character set using the Design mode. To use this feature, load the @Design file (it occupies the same memory area as Print, thus the two cannot be resident in memory at the same time). You may design characters for a 40, 56, 70 column display, depending on the mode that is selected when the Design mode is entered. The full character set currently in use is displayed along with a large matrix used for design of your own characters. After specifying the particular character to be designed, using the cursor movement and dot on-off commands provided allows you to set your own. Once completed, your character set may be saved on disk and is identified by an @ preceding its name.

A few of the characters used in all displays may not be familiar to you. The carriage return, for instance, is a curved arrow. Control characters have also changed from the simple inverse video used in previous versions of Super-Text (except for the 80-column user of Super-Text 40/80).

What's Missing?

Super-Text 40-56-70 is primarily intended as a simplified version of Super-Text 40/80. What features are missing in this program? Split-screen editing has not been included. Noticeably absent is a Math mode that provides for calculations, columnar addition and alignment of numbers in a text file or entered from the keyboard.

Impressions Of Super-Text 40-56-70

Perhaps I should preface my remarks in this section by telling you that I have used every version of Super-Text. My applications for word processing are rather numerous, so each of these versions has seen quite a bit of use.

Super-Text 40-56-70 has faithfully incorporated all of the desirable features of previous versions of Super-Text. Each new version shows an improvement in the documentation and user friendliness of the program. In most respects, I find this program to be a very capable word processing system. Super-Text 40-56-70 will, by the way, interface with both Form Letter Module and Address Book from Muse to provide personalized form letter and document capabilities.

The software generated character sets provide some interesting display options. While all three of them are adequate for the intended purpose, I found the 70-column set somewhat hard to read (especially after an hour or two at the keyboard) and the 56-character set especially easy in that regard.

If you have ever attended a conference at school for your youngster, you know the usual format is to say all the good things first and then (with the proverbial *but*) lay on the bad news. Well folks, here is the *but* in this review. The switch to software generated characters has bogged down the video display operations of Super-Text to a pronounced degree. Although there is a type-ahead buffer so characters are not lost, the screen is rarely able to keep up with all but the most mediocre typist. You will often find it lagging several characters behind you. This really takes some getting used to! Even though I am far from the world's fastest typist, I find it necessary to pause after every few characters to let the screen catch up. The same problem is noted using text movement operations. Rather than scrolling in the normal fashion, each line ripples as it is updated on the screen.

Since I am accustomed to the speed and screen efficiency of Super-Text 40/80 (the same in both 40 and 80 column modes, by the way), Super-Text 40-56-70 really wouldn't convince me to switch to a software-generated character set program. Although there are many users who really like the alternate character sets and are willing to live with the speed limitations inherent in their use, my preference lies elsewhere.

It would be nice if some provision were made to print the characters you design using the program. No reference is made to that capability at this time, although some enterprising Apple users will likely find a method of doing so.

Summary

Super-Text 40-56-70 and Super-Text 40/80 are advertised as high quality word processors, combining different features for different user preferences. There is no doubt their ad copy is quite accurate—both programs are indeed very capable word processors for the Apple II. This user, however, much prefers the speed and efficiency of Super-Text 40/80.

Super-Text 40-56-70 sells for $125 and is published by Muse Software Inc., 347 N. Charles St., Baltimore, MD 21201. ■

The Echo II Speech Synthesizer

An electronic Laurence Olivier it's not, but this exciting piece of hardware does put useful speech synthesis within reach of most Apple owners. There, you see, talk is still cheap.

by Chuck Doherty

Although it is not presently possible for a home computer to produce a convincing, totally human voice, it *is* possible to acheive some uncannily realistic results with speech simulation systems already on the market.

Of course, developing any speech synthesizer for a home computer entails many compromises. Processor speed, memory limitations and input/output architecture must all be considered in the design process. Price, too, is a big factor; a speech board costing more than the computer itself would be no bargain.

Speech synthesizers fall into two broad categories: those which are primarily hardware-based, and those which are built around software. There are advantages and drawbacks to both designs. One product which falls into the "hardware" category is the Echo II Speech Synthesizer from Street Electronics, 1140 Mark Ave., Carpinteria, CA 93103 for $149.95.

The Echo II is based on Texas Instruments' TMS 5200 speech processor. (Street describes its system as an "upgraded version of the one used in the Speak and Spell.") The TMS 5200 uses a system known as "linear predictive coding," which reduces the parameters of speech to a form of "shorthand," allowing an unusually large amount of speech information to be stored in a relatively small portion of memory.

As with most add-ons for the Apple, installation is made via the main bus. The Echo II is not fussy about which slot you choose—it will work in all of them. In addition to the board itself, the system includes a small, enclosed loudspeaker for audio output. Double-sided tape on the speaker box allows mounting inside the computer case.

A disk, which can be easily copied, contains the software portion of the package. Along with demo software, there are programs which facilitate the translation of written English into speech. There is also a provision for entering data in phonetic code for a more exact pronunciation.

The heart of the system is a program called Speakeasy. Used by most of the other programs on the disk, Speakeasy allows speech to be generated from phonetic code data. In many cases, this phonetic code does not resemble the spelling of the word it represents. For instance, although the phonetic code for the word "yes" is "yes," the code for the word "thousand" is ")#zs'nd,"

)#zs'nd," Island dressing, Sir?

When using the Speakeasy program, data is spoken by "printing" a string containing the code after entering a control-V. As an example, the manual provides the following short program which demonstrates Speakeasy's operation; when run, it will produce the words "Echo Speech."

```
20    V$5CHR$(22)
30    PRINT V$"EKO"
40    PRINT V$"SP&C"
50    "DONE".
```

Line 20 assigns the variable V$ to the code for a control-V so that we may easily call for it when needed. Line 30 begins by sending out the signal to turn speech on, control-V, and then sends the data to be spoken; in this case that is the string "EKO", which is the phonetic code for the word "ECHO."

Unless a print statement ends with a

Address correspondence to Chuck Doherty, 32 Meadowood Drive, So. Dartmouth, MA 02748.

semicolon, it is automatically followed by a carriage return, which turns the speech mode off again. In this case, line 30 ends with an implied carriage return, forcing us to again send a control-V in line 40 to engage the speech mode. Line 40 also contains the phonetic code for the word "Speech," which is "SP&C". At the beginning of line 50 there is no control-V, so any data following the print command will not be spoken but will merely be written to the screen. Running this program results in the words "Echo Speech" being spoken, followed by the word "DONE" being printed to the screen.

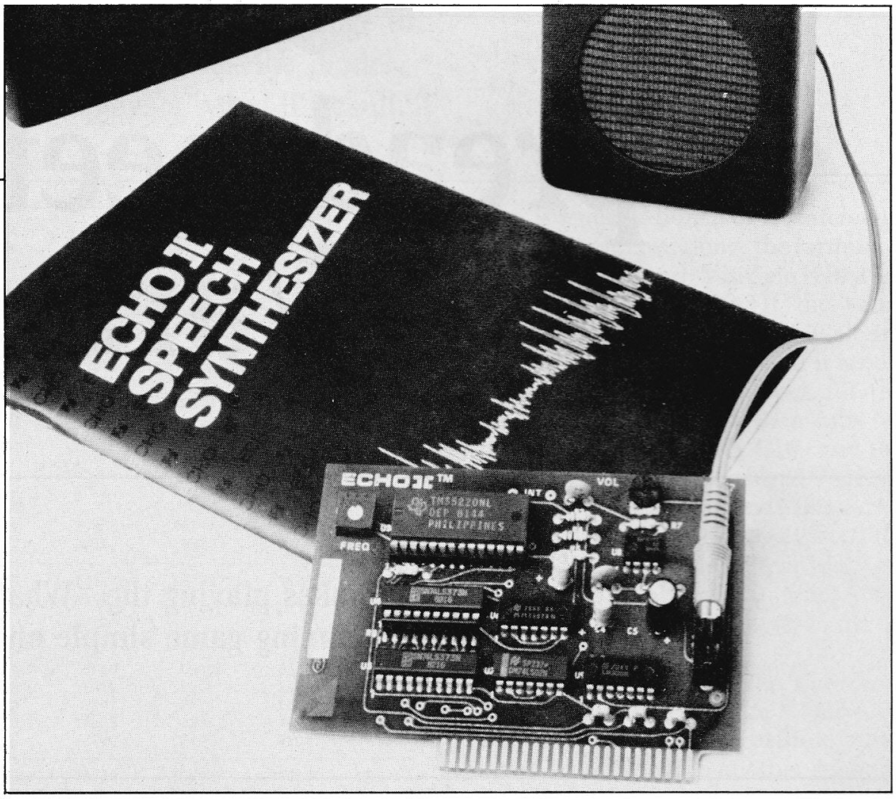
One very helpful mode of Speakeasy's operation is the phoneme editor which allows the direct input of phonetic code, which is spoken immediately. The editor allows for easy experimentation with the Echo system

There is much more to speech than the pronunciation of words. There is also stress, rate, pitch and volume. Speakeasy provides control of all of these modifiers, even within a single word. Subtle changes in modulation can result in very different meanings and make the difference between "canned" and natural-sounding speech.

All of us unconsciously use changes of pitch and inflection. A good example is the change of pitch we use at the end of a sentence when asking a question. Without inflection, speech becomes very literal and the definition of a phrase stays the same. For instance, by changing the emphasis within a simple phrase such as "My Apple is very versatile," the meaning varies widely.

Speakeasy provides several different means of changing pitch within a phrase. One is by using the numbers 1–9 which, when inserted in the proper place, will raise or lower pitch. When the program encounters fricatives (such as *th*) or stop consonants (like k from kick) the pitch is reset to the value held before it was changed.

Volume, too, may be altered with Speakeasy. The symbols + and – increase and lower the volume. The Echo II won't shout, but it will go all the way down to a whisper.



The Echo II Speech Synthesizer.

Also on the main disk is a series of programs titled Textalker. The main Textalker program is designed to translate ordinary written English into a form that can be processed by Speakeasy, a sub-program of Textalker.

The many exceptions to logical pronunciation may be second nature to us, but programming a computer to handle words such as "debut" is another thing altogether. According to Street, Textalker uses over 400 rules of pronounciation to give as accurate a response as possible to the input. Although this system works fine in many cases, you may find yourself spelling certain words in a way that Textalker can better understand. One example given in the manual is the word "robot." Entered without change, the first "o" is pronounced, as in the word "rob." In order for Textalker to handle this word correctly, you must enter it spelled "rowbot."

As with the Speakeasy program, data is converted from text to speech by placing it within quotes and giving a print command. Textalker allows for three different modes of output: speech only, write to the screen (as in an ordinary print command), or both.

Textalker has variable pitch and volume levels and will even automatically adjust the pitch to coincide with punctuation, lowering pitch at the end of a sentence and raising it when it sees a question mark.

If desired, the program can pronounce individual letters rather than entire words. The program will pronounce some of the punctuaton marks it encounters, but if you wish, it will say *most* of the punctuation or even all of it (including line feeds and carriage returns).

I was most impressed by one particular program on the disk, "Textalker.Blind." This routine allows sight-impaired persons to use a computer with an ease not possible before. With it, the operator can review the contents of the screen by moving an "audio cursor" in any direction and having whatever it encounters spoken aloud.

The manual supplied with the Echo II could have been better. Some portions are almost patronizing in their simplicity, while the technical and advanced information is sparse and vague. Also, the English-to-phoneme dictionary of only 35 words could stand some enlargement.

The future will undoubtedly hold great advances in the speech synthesis field, but until then, the Echo offers some outstanding performance. Although Street Electronics has made speech for the Apple II simple to implement and quite affordable, you will still have to do your share of the work for the product to give its all. ■

Spreadsheet Variables

VisiCalc makes playing the "What If?" business planning game simple and fast.

by Archie Mason

One of the most significant and exciting features of VisiCalc, the first of the spreadsheet programs, is its ability to "what-if" with ease and speed. You probably know the joy of preparing a large and complex matrix of sales, expenses, seasonal factors, personnel requirements, and so on, and then being able, with a few keystrokes, to change a few critical entries to see the effect on other entries.

This month I am going to explain a method to speed up this process, enabling you to keep track of the variables you have already "what-iffed" and to quickly re-enter them for another review. Instead of tracking around the model entering the new variables to be examined in the various places they occur, you will be able to preload groups of variables and speed up the printing of the various versions of the matrix.

I am also going to mention some of the more useful functions of VisiCalc: nested If statements, a variation on the @SUM formula, and some other VisiTips (the little shortcuts and hints we feature in *Spreadsheet*, the InterCalc Users Group monthly newsletter).

Varying Variables

To speed up "what-iffing" you need an area in the model, preferably at the top, for "matched" groups of variables. Look at Figure 1. It represents the top of a sales profit and loss model. No criticisms, please, of the bookkeeping validity of the model—it's just for illustration purposes.

In preparing for a meeting our operator identified a group of factors to be examined. The first was the sales in the first month of a new sales operation. Then came the cost of the merchandise slated to be sold—there were various possibilities. The rate at which sales were projected to grow each month was another variable, as was the percent of expected sales to be budgeted for advertising, the commission to be paid the salespeople, and the related cost of making telephone sales, an important contributor to sales and overhead.

The operator in our example made provision for four "sets" or combinations of these variables, and there could have been more (you'll see the limit in a moment). Notice that under each column of factors there is a location labelled Choose, and that in Column B there is an asterisk. It comes from the entry of a 1 in B10, with the format set /F*. The locations C10 and E10 are similarly set, but are empty. The reason the asterisk is on the left is a peculiarity of VisiCalc—there are some formatting commands that are mutually exclusive. You cannot have an asterisk flush right, for instance, or an integer value flush left. If the offset bothers you, don't use the asterisk, just

	A	B	C	D	E	F
1		VARIABLE	TABLES			CHOSEN
2		1	2	3	4	FACTOR
3		----------	----------	----------	----------	----------
4	START SALES	22000	46000.00	51000.00	22000.00	22000
5	COST OF GDS	0.30	0.35	0.40	0.40	.3
6	INCREASE RT	.05	.075	.12	.15	.05
7	ADVERTISING	.075	0.15	0.22	0.17	.075
8	COMMISSION	0.10	0.17	0.45	0.22	.1
9	TELEPHONE	0.05	0.20	0.23	0.11	.05
10	CHOOSE->	*				
11						

Figure 1. The sets of variables.

Address correspondence to Archie Mason, c/o InterCalc, PO Box 254, Scarsdale NY 10583. A stamped, self-addressed envelope will speed a reply.

"Every time you change the variable set, the model displays a 'what-if' result."

	A	B	C	D	E	F	G	H
1		VARIABLE	TABLES			CHOSEN		
2		1	2	3	4	FACTOR		
3								
4	START SALES	22000	46000.00	51000.00	22000.00	22000		
5	COST OF GDS	0.30	0.35	0.40	0.40	.3		
6	INCREASE RT	.05	.075	.12	.15	.05		
7	ADVERTISING	.075	0.15	0.22	0.17	.075		
8	COMMISSION	0.10	0.17	0.45	0.22	.1		
9	TELEPHONE	0.05	0.20	0.23	0.11	.05		
10	CHOOSE->	*						
11								
12								
13		JANUARY	FEBRUARY	MARCH	APRIL	MAY	JUNE	TOTAL
14								
15	GROSS SALES	22000	23100	24255	25468	26741	28078	149642
16	COST OF GDS	6600	6930	7277	7640	8022	8423	44893
17								
18	GR. PROFIT	15400	16170	16979	17827	18719	19655	104749
19								
20	EXPENSES							
21								
22	OFFICE RENT	2200	2200	2200	2200	2200	2200	13200
23	SALARIES	6500	6500	6500	6500	6500	6500	39000
24	ADVERTISING	1650	1733	1819	1910	2006	2106	11223
25	COMMISSION	2200	2310	2426	2547	2674	2808	14964
26	TELEPHONE	1100	1155	1213	1273	1337	1404	7482
27	ALL OTHER	2200	2200	2200	2200	2200	2200	13200
28								
29	TOTAL EXP.	15850	16098	16357	16630	16917	17218	99069
30								
31	NET PROFIT	-450	73	621	1197	1802	2437	5680
32								
33	"CASH FLOW"	-450	-378	244	1441	3243	5680	

Figure 2. Model with the first set selected.

	A	B	C	D	E	F	G	H
1		VARIABLE	TABLES			CHOSEN		
2		1	2	3	4	FACTOR		
3								
4	START SALES	22000	46000.00	51000.00	22000.00	22000		
5	COST OF GDS	0.30	0.35	0.40	0.40	.4		
6	INCREASE RT	.05	.075	.12	.15	.15		
7	ADVERTISING	.075	0.15	0.22	0.17	.17		
8	COMMISSION	0.10	0.17	0.45	0.22	.22		
9	TELEPHONE	0.05	0.20	0.23	0.11	.11		
10	CHOOSE->			*				
11								
12								
13		JANUARY	FEBRUARY	MARCH	APRIL	MAY	JUNE	TOTAL
14								
15	GROSS SALES	22000	25300	29095	33459	38478	44250	192582
16	COST OF GDS	8800	10120	11638	13384	15391	17700	77033
17								
18	GR. PROFIT	13200	15180	17457	20076	23087	26550	115549
19								
20	EXPENSES							
21								
22	OFFICE RENT	2200	2200	2200	2200	2200	2200	13200
23	SALARIES	6500	6500	6500	6500	6500	6500	39000
24	ADVERTISING	3740	4301	4946	5688	6541	7522	32739
25	COMMISSION	4840	5566	6401	7361	8465	9735	42368
26	TELEPHONE	2420	2783	3200	3681	4233	4867	21184
27	ALL OTHER	2200	2200	2200	2200	2200	2200	13200
28								
29	TOTAL EXP.	21900	23550	25448	27630	30139	33025	161691
30								
31	NET PROFIT	-8700	-8370	-7991	-7554	-7052	-6475	-46142
32								
33	"CASH FLOW"	-8700	-17070	-25061	-32615	-39667	-46142	

Figure 3. Model with the second set selected.

use 1 for yes (chosen) and 0 for no (not chosen). Then you can have them neatly lined up in their respective columns!

On the right side of this area, in Column F, you'll find the factors chosen for review—the ones from Column B. It is the formula in F that brought them there:

@IF(B10 = 1,B4,@IF(C10 = 1,C4,@IF(D10 = 1,D4,@TF(E10 = 1,E4,0))))

This is a nested bunch of If statements and, if you follow along, the formula is easy to understand. VisiCalc is first instructed to assess B10. If the value there is equal to 1, then VisiCalc inserts the value found in B4. If it is not, the program goes on to look for the next instruction, which then calls for the same judgement on C10, and the same subsequent procedure. VisiCalc keeps checking until it finds an asterisk (1), or it finally defaults to a zero. This formula is /Replicated down the column, with the Row 10 items (N)o change, and the bringover items as (R)elative.

In effect, then, you can move any one of columns B through E into Column F by blanking the existing asterisk and inserting it in another column. You are probably way ahead of me by now—the model is instructed to use whatever factor it finds in F for calculating, and every time you change the variable set, the model displays a "what-if" result.

The limit in the number of variable sets I mentioned is, of course, the maximum number of characters in a formula permitted by VisiCalc on the Apple. Assuming that you are working in Rows 1 through 9, which means that the location references in the formula are the minimum length (A1 instead of

"When you finally get the variable sets you want available for review plugged into the top, making printouts of the various combinations is easy."

A12, two characters instead of three), then you can have ten columns of variables using the nested @IF style.

VisiTips

The rest of the model illustrates some VisiTips you can use every day. The entry in B15 comes down from F4, the first of the "variable" variables. The February sales apply the Increase Rate multiplier from F6, the formula being (preceding month + (preceding month times F6)). A five percent increase results. The cost of goods and other variables perform the same way.

Two "what-ifs" appear in Figure 2 and 3. In Figure 2, the result of spending too much advertising money against too few sales—you lose your shirt! When you finally get the variable sets you want available for review plugged into the top, making printouts of the various combinations is easy. Choose the first column of variables, print, choose another, print, and so on, and the job is done in a few moments.

First VisiTip: The formula in Row 29 is @SUM (Row 20. . . Row 28). So why do I include all that "white space" and the underlines? In model development it is often necessary to insert rows to add new items. If you start the @SUM range at a point you know you'll never exceed, and end it similarly at a level you will never go below, then inserted rows will always be included in the VisiCalc update of the formula. On the other hand, using an exact @SUM (Row 22. . . Row 27) means you cannot insert a new item before Row 22 or after Row 27 without remembering to change the summing formula and to replicate it across.

Second VisiTip: See that "Cash Flow" line? It is a running total, adding the earnings for the current month to the accumulated net profit. The first item, in Column B, is a simple bring-down of the net, and the second could be B33 (last month's total) plus C31 (the net for this month). But there is a simpler way. Since in doing an @SUM replication, VisiCalc asks that the relationship be used for both ends of the range, just use (N)o change for the first one and (R)elative for the second. In this way, each successive @SUM makes the range one location longer.

Incidentally, this tip can be used for a number of the VisiCalc functions that ask for ranges. By employing it, for instance, in an @MAX application, you can ask if this is the biggest value so far, rather than merely if this is the biggest item in the range.

Thanks to all of you who have sent letters. I am up-to-date on the replies (to those who sent stamped self-addressed envelopes). Please continue to write if there are things I can help you with. ■

Franklin's ACE 1000 Runs With The Best!

VisiCalc®, DB Master®, Desktop Plan®—they are all running on the Franklin ACE 1000. Cash flow, budgets, word processing or data base management, business or pleasure, the ACE 1000 runs with the best.

The Franklin ACE 1000 is hardware and software compatible with the Apple® II. Franklin users can choose from an enormous selection of programs—programs that run better on the ACE because it includes 64K of RAM, upper and lower case, VisiCalc keys, a numeric pad and an alpha lock key.

Run with the best. Call or write today for the name of your local authorized Franklin dealer.

Franklin ACE is a trademark of Franklin Computer Corporation.
Apple is a registered trademark of Apple Computer Inc.
VisiCalc and Desktop Plan are registered trademarks of Visi Corp.
DB Master is a registered trademark of Stoneware.

FRANKLIN
COMPUTER CORPORATION
2138 Route 38
Cherry Hill, NJ 08002
609-482-5900
Telex: 837-385

Circle 184 on Reader Service card.

All-American Program

by Jeffrey A. Mills

This program has been written in the "Spirit of '76," Mom, and Apple pie for your July Fourth celebration!

```
10  HGR2
20  FOR S = 6 TO 5 STEP  - 1
30  HCOLOR= 5 + (S = 5) * 2
40  FOR A = 0 TO S:C = B + 12
50  FOR Y = B TO C
60  HPLOT 0,Y TO 278,Y
70  NEXT
80 B = B + 28
90  NEXT
100 B = 14
110  NEXT
120  HCOLOR= 6
130  FOR Y = 0 TO 82
140  HPLOT 0,Y TO 92,Y
150  NEXT
160  HCOLOR= 7
170  FOR Y = 15 TO 71 STEP 14: FOR
     X = 10 TO 82 STEP 8
180 Z = X + 1:T = Y + 1
190  HPLOT X,Y TO Z,Y TO Z,T TO X
     ,T
200  NEXT : NEXT
210  FOR V = 770 TO 795: READ N: POKE
     V,N: NEXT
220  FOR A = 1 TO 6: READ D,F: POKE
     768,D: POKE 769,F: CALL 770:
      NEXT
230  DATA 172,1,3,174,1,3,169,4,3
     2,168,252,173,48,192,232,208
     ,253,136,208,239,206,0,3,208
     ,231,96,7,100,5,70,29,18,10,
     70,9,100,25,148
```

Address correspondence to Jeffrey A. Mills, 789 Ebner St., Columbus, OH 43206.

Save It!

Making the most of DIF in VisiCalc, VisiPlot & pfs:graph

by Trish McClelland

A DIF (Data Interchange Format) file is a feature found in many software packages. It lets you transfer data from one file to another within a program, or between different software packages. It is especially useful when you want to save only a portion of the file for insertion in another file.

Using DIF in VisiCalc

For example, let's say that you have created a file with column headings that read "Category" and "Jan" through "Aug" (see Table 1). In the Category column you have inserted category names and, across those rows (in the columns identified as months), you have inserted values. You then saved the file with the /SS command.

Now you wish to begin another file, but the column headings and categories remain the same. You can input that information over again manually, or you can load the old file, change the numbers, and save it under a different name. Or you can utilize DIF. This would save only the portions you wish to save, requiring less effort but increasing your productivity.

To do this, you would simply position your cursor at the upper left corner of the rectangle you wish saved. Then type /S#. The prompt will read:

DATA: SAVE LOAD

Since you are saving a file, you will type the letter S. You are then asked to give a file name. After typing the name of your choice, I recommend that you add the letters .DIF to identify it as a DIF file. This file can only be accessed by the /S# command. After typing the file name, press return. You are prompted for the lower right corner. This is the lower right of the portion you wish saved, not necessarily the lower right of the worksheet. Type in the coordinate or move your cursor to that position, and press return. The prompt now reads:

DATA SAVE: R, C, OR RETURN

This is asking whether you want your data to be saved by rows, columns or return for the default. You would type C if you were saving a column, such as the column of categories in our example; R would be used if you wanted a row saved; return is used if it doesn't matter whether the data is saved in rows or in columns.

Now you are ready to load it into another file. A DIF file can be inserted at any point in your spreadsheet. Simply position your cursor at the desired coordinate and type /S#, this time typing the letter L for Load. You are then asked the file name, and if you want it loaded by rows, columns or return for the default. This gives you the opportunity to save a column but load it as a row.

A DIF file can be as large or as small as you desire. Values can also be saved, but algorithms will be erased and only the result (the number) will be saved. This uses less of your valuable memory, and therefore increases file capacity.

Transferring Files from VisiCalc to VisiPlot

A common use of DIF files is storing data in VisiCalc and loading it into VisiPlot. Before doing this, however, you should know how the various packages define the data.

VisiCalc is a calculations package. It operates with a collection of interrelated labels, numbers and formulae. It contains a lot of detail, and usually gives an end result such as totals, net present value, variances, etc.

VisiPlot is a graphics package. Graphs are normally concerned only with the end result (such as totals). It sees all numbers as time related, and interprets all labels (except the first in a series) as zeros. Blanks are also considered zeros. The label at the beginning of the series is interpreted as the name of the file.

Patricia R. McClelland (8283 Greensboro Drive, McLean, VA 22102) writes poetry, murder mysteries and women's features in addition to computer articles.

1983 HEADCOUNT PLANNING MODEL

CATEGORY	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG
LEGAL	30	30	30	31	32	33	33	33
MKTG.	50	55	56	56	56	57	58	60
ENG.	70	70	70	72	72	73	73	74
FINANCE	22	23	24	24	24	24	24	24
PERS.	10	12	12	15	15	15	16	16
ADMIN.	25	25	30	32	32	32	32	33

Table 1.
Example of a data file.

First you must have the data points you wish to use in VisiCalc as a DIF file. Multiple rows or columns can be stored. Each row or column (depending on the command you gave when storing it) will be considered an independent series.

Let's take another look at our example (see Table 1). If you were interested in plotting this data, you would position the cursor at the word LEGAL before typing the /S# command. If the coordinate you chose as the lower right corner is Administration's headcount for August (33), you will have six independent series. The labels in the category column will become the names of the respective files.

Then load your VisiPlot program with the VisiCalc data disk in Drive 2. When your Main Storage Menu appears, choose LOAD and select the DIF file saved previously.

The program will ask for the periodicity (such as 12 for annually, 4 for quarterly, etc.), the major start (the year), and the minor start (the month).

The example we are working with would be defined as the following:

PERIODICITY	12
MAJOR START	1983
MINOR START	1

You have successfully transferred your file from VisiCalc to VisiPlot at this point. Now you can remove your VisiCalc data disk from Drive 2 and replace it with the VisiPlot data disk. Once it has been inserted, you can save your data in a normal VisiPlot file. Do not save data from two different programs on the same disk.

You are now able to treat this file as you would any other VisiPlot file, and edit or plot your series accordingly.

"You are now able to treat this VisiCalc file as you would any other VisiPlot file."

Transferring Files from VisiCalc to pfs:graph

The pfs:graph program, like VisiPlot, is a graphics package. Although manufactured by a different company (Software Publishing Corporation), it is compatible with VisiCalc.

We know that VisiCalc operates with a collection of interrelated labels, numbers and formulae; pfs:graph breaks these down into X and Y axes, with the X axis identified as time-related (day, month, year, etc.), as a number or as an identifier (which enables the insertion of words). The Y data is always a number.

First save the VisiCalc data in a DIF file. Then load your pfs:graph with the VisiCalc data disk in Drive 2. When the program is loaded, you will choose option #1 (GET/EDIT DATA) from the main menu. A new menu will appear and you will choose #2 (GET VISICALC FILE). Enter the additional information and press control-C to continue.

The Get VisiCalc File menu will now be displayed on your screen. There is certain information which you still must enter before the transfer of data is complete. The prompt for this information is displayed on your screen at this point. You are asked for the file name (enter it exactly as it was originally typed in VisiCalc), the X and Y data and the X data format.

The X and Y data prompts refer to the location of the data in the DIF file. It does *not* refer to the original worksheet. For instance, if we saved the first two rows from the example in Table 1, our DIF file would begin at JAN and end at 33, ignoring the category column altogether. The first row is the X data and the second row is the Y data. Therefore,

X DATA:	1
Y DATA:	2

The last prompt, X Data Format, is asking you to define the X data. Is it a month, year, number, identifier, or something else? For our example, we will enter M. Then press control-C to continue.

The message READING DATA will appear on your screen, and your data points will be counted. When the data has been successfully transferred to pfs:graph, you will be returned to the main menu. You can then edit, display or print your graph.

When you are storing your data in pfs:graph, remember to remove your VisiCalc data disk and insert the pfs:graph data disk, or your VisiCalc data will be erased!

DIF files can be helpful in many different programs. Some advantages are speed, efficiency, error reduction and simplicity. When running out of memory in VisiCalc, I've saved parts of the file in DIF and overlaid them on the spreadsheet, letting me complete the project. DIF files can be used in a multitude of ways, given a lot of imagination and a little flexibility. ■

Viva Las Vegas

by Larry Bohn

While watching a friend play a matching game on a TI-99, I decided to write a similar game for my Apple II. But after many changes, my game did not resemble my friend's at all. I had created a slot machine game. It's not a traditional slot machine—you don't use coins—but it is fun anyway.

Begin the game by entering your name and the amount of gambling money you have. This can be any amount. Be prepared to lose it, though. As in Las Vegas, you don't win often.

Your bet must be entered next. A bet of 0 allows you to exit the program giving you your winnings or losses. You cannot bet more money than you have.

The pictures are now selected and displayed in low-resolution graphics. These pictures include oranges, cherries, lemons and, of course, a bag of money. The payoff is calculated, displayed and the new amount of your gambling money is displayed for your convenience. If you have lost all of your money, you are given the opportunity to play again or quit. If you still have some money, you can play slot

Address correspondence to Larry Bohn, 2925 Foxcroft Court One, Topeka, KS 66614.

Inside Apple

Apple Computer Inc., 20525 Mariani Avenue, Cupertino, California 95014 Vol. 1 No. 2

For the authorized Apple dealer nearest you, call **800-538-9696** (**800-662-9238** in California.)

Fruitful Connections.

There are more people in more places making more accessories and peripherals for Apples than for any other personal computer in the world.

Thanks to those people—in hundreds of independent companies—you can make the humblest 1978 Apple II turn tricks that are still on IBM's Wish List for 1984.

But now we're coming out with our very own line of peripherals and accessories for Apple® Personal Computers.

For two very good reasons.

First, compatibility. We've created a totally kluge-free family of products designed to take full advantage of all the advantages built into every Apple.

Second, service and support.

Now the same kindly dealer who keeps your Apple PC in the pink can do the same competent job for your Apple hard-disk and your Apple daisywheel printer.

So if you're looking to expand the capabilities of your Apple II or III, remember:

Now you can add Apples to Apples.

A joy to behold.

The new Apple Joystick II is the ultimate hand control device for the Apple II.

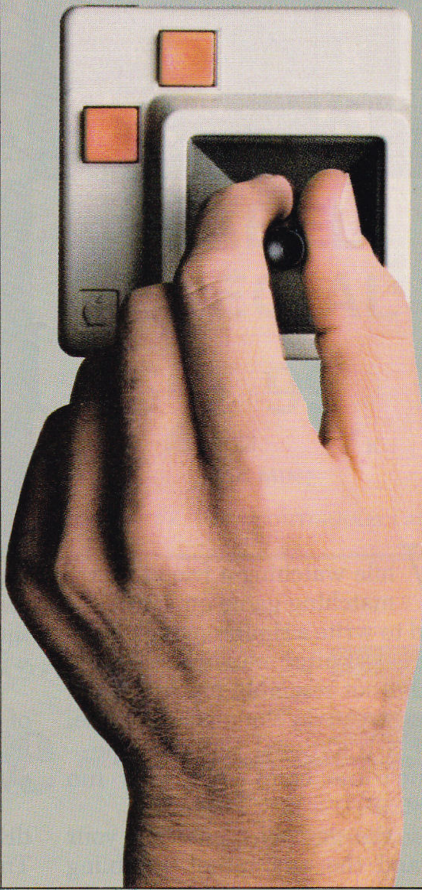
Why is it such a joy to use?

With two firing buttons, it's the first ambidextrous joystick—just as comfortable for lefties as righties.

Of course, it gives you 360° cursor control (not just 8-way like some game-oriented devices) and full X/Y coordinate control.

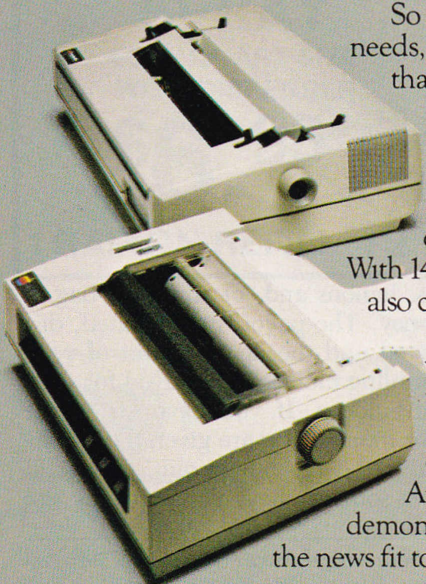
And the Joystick II contains high-quality components and switches tested to over 1,000,000 life cycles.

Which makes it a thing of beauty. And a joystick forever.



Gutenberg would be proud.

Old Faithful Silentype® has now been joined by New Faithfuls, the Apple Dot Matrix Printer and the Apple Letter Quality Printer.



So now, whatever your budget and your needs, you can hook your Apple to a printer that's specifically designed to take advantage of all the features built into your Apple. With no compromises.

The 7x9 Apple Dot Matrix Printer is redefining "correspondence quality" with exceptional legibility. With 144x160 dots per square inch, it can also create high resolution graphics.

The Apple Letter Quality Printer, which gets the words out about 33% faster than other daisywheel printers in its price range, also offers graphics capabilities. See your authorized Apple dealer for more information and demonstrations. Because, unfortunately, all the news fit to print simply doesn't fit.

A storehouse of knowledge.

If you work with so much data or so many programs that you find yourself shuffling diskettes constantly, you should take a look at Apple's ProFile,™ the personal mass storage system for the Apple III Personal Computer.

This Winchester-based 5-megabyte hard disk can handle as much data as 35 floppies. Even more important for some, it can access that data about 10-times faster than a standard floppy drive.

So now your Apple III can handle jobs once reserved for computers costing thousands more.

As for quality and reliability, you need only store one word of wisdom:

Apple.

Up the creek without a paddle?

Or lost in space? Or down in the dungeons?

Whatever your games, you'll be happy to know that someone has finally come out with game paddles built to hold up under blistering fire. Without giving you blisters.

Apple Hand Controller II game paddles were designed with one recent discovery in mind:

People playing games get excited and can squeeze very, very hard.

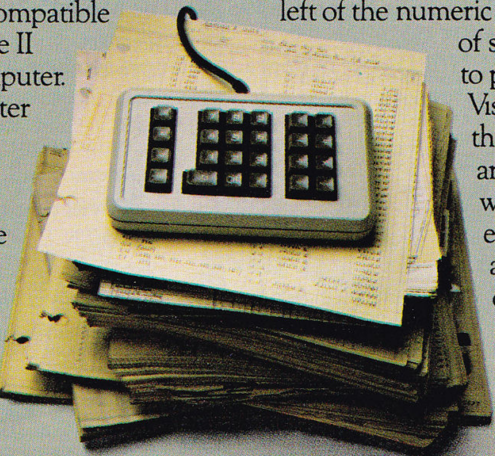
So we made the cases extra rugged. We used switches tested to 3,000,000 life cycles. We shaped them for holding hands and placed the firing button on the right rear side for maximum comfort.

So you'll never miss a shot.

Launching pad for numeric data.

Good tidings for crunchers of numerous numbers:

Apple now offers a numeric keypad that's electronically and aesthetically compatible with the Apple II Personal Computer. So you can enter numeric data faster than ever before.



The Apple Numeric Keypad II has a standard calculator-style layout. Appropriate, because unlike some other keypads, it can actually function as a calculator.

The four function keys to the left of the numeric pad should be of special interest to people who use VisiCalc.® Because they let you zip around your work sheet more easily than ever, adding and deleting entries.

With one hand tied behind your back.

machine again.

Program lines 20 through 110 create a title block. Lines 120 through 300 accept your bet. Lines 310 through 350 set up the slot machine in low-resolution graphics. Lines 360 to 710 select the shapes for the slot machine in three positions by random numbers. Lines 720 through 880 tally the score.

The payoff amount can be changed with lines 800, 810 and 840. Lines 890 to 970 let you decide to play again or exit the program. Lines 980 through 1340 create the four shapes—a bag of money, a cherry, a lemon and an orange.

The ten variables in the program are:

A Starting Money.
B Gambling Money.
C Amount Bet.
D Picture in First Position.
E Picture in Second Position.
F Picture in Third Position.
G Yes or No to Play Again.
H Amount Won or Lost.
X Used to Position Pictures.
Z Your Name.

Good luck! ■

Program listing. Slot Machine.

```
20   HOME : VTAB (7)
30   PRINT  TAB( 10)"$ $ $ $ $ $ $
       $ $ $ $"
40   PRINT : PRINT
50   PRINT  TAB( 14)"SLOT MACHINE"

60   PRINT : PRINT
70   PRINT  TAB( 15)"CREATED BY"
80   PRINT  TAB( 13)"LARRY R. BOHN
      "
90   PRINT : PRINT  TAB( 13)"DECEM
      BER, 1982"
100   PRINT
110   PRINT  TAB( 10)"$ $ $ $ $ $
      $ $ $ $ $"
120   FOR PAUSE = 1 TO 1000: NEXT
      PAUSE
130   HOME : FOR PAUSE = 1 TO 200:
       NEXT PAUSE: VTAB (4)
140   PRINT "TELL ME YOUR NAME.": INPUT
      Z$
150   PRINT : PRINT Z$;: PRINT ",
      HOW MUCH"
160   PRINT "GAMBLING MONEY DO YOU
       HAVE?";: INPUT A: PRINT
170 B = A: GOTO 230
180   IF B > 0 THEN 220
190   TEXT : HOME : FOR PAUSE = 1 TO
      200: NEXT PAUSE: VTAB (4)
200   PRINT Z$;: PRINT ", YOU'RE B
      ROKE, LET SOMEBODY WITH"
210   PRINT "SOME MONEY PLAY OR FI
      ND SOME MORE MONEY.": FOR PA
      USE = 1 TO 1000: NEXT PAUSE:
       GOTO 860
220   PRINT : PRINT Z$;: PRINT ",
      YOU HAVE $";: PRINT B: PRINT

230   PRINT "HOW MUCH DO YOU WANT
      TO BET?";: INPUT C: PRINT
240   IF C = 0 THEN 850
250   IF C = B THEN  GOTO 300
260   IF C < B THEN  GOTO 300
270   IF C > B THEN  PRINT "YOU DO
      N'T HAVE THAT MUCH MONEY,"
280   PRINT "PLACE YOUR BET AGAIN.
      "
290   FOR PAUSE = 1 TO 1250: NEXT
      PAUSE: GOTO 180
300   FOR PAUSE = 1 TO 250: NEXT P
      AUSE
310   GR
320   COLOR= 15
330   FOR N = 13 TO 25
```

Listing continued.

Listing continued.

```
340  HLIN 0,39 AT N
350  NEXT N
360  REM  PIC1=D
370 D =  INT ( RND (1) * 9) + 1
380 X = 0
390  IF D = 1 THEN 980
400  IF D = 2 THEN 1090
410  IF D = 3 THEN 1180
420  IF D = 4 THEN 1270
430  IF D = 5 THEN D = 2: GOTO 10
     90
440  IF D = 6 THEN D = 3: GOTO 11
     80
450  IF D = 7 THEN D = 4: GOTO 12
     70
460  IF D = 8 THEN D = 2: GOTO 10
     90
470  IF D = 9 THEN D = 4: GOTO 12
     70
480  REM  PIC 2=E
490 E =  INT ( RND (1) * 9) + 1
500 X = 13
510  IF E = 1 THEN 980
520  IF E = 2 THEN 1090
530  IF E = 3 THEN 1180
540  IF E = 4 THEN 1270
550  IF E = 5 THEN E = 2: GOTO 10
     90
560  IF E = 6 THEN E = 3: GOTO 11
     80
570  IF E = 7 THEN E = 4: GOTO 12
     70
580  IF E = 8 THEN E = 2: GOTO 10
     90
590  IF E = 9 THEN E = 4: GOTO 12
     70
600  REM  PIC 3=F
610 F =  INT ( RND (1) * 9) + 1
620 X = 26
630  IF F = 1 THEN 980
640  IF F = 2 THEN 1090
650  IF F = 3 THEN 1180
660  IF F = 4 THEN 1270
670  IF F = 5 THEN F = 2: GOTO 10
     90
680  IF F = 6 THEN F = 3: GOTO 11
     80
690  IF F = 7 THEN F = 4: GOTO 12
     70
700  IF F = 8 THEN F = 2: GOTO 10
     90
710  IF F = 9 THEN F = 4: GOTO 12
     70
720  IF F = E THEN 740
730  IF D <  > E THEN 820
740  IF D = F THEN 760
750  IF D <  > F THEN 820
760  IF D = 1 THEN  GOTO 840
770  IF D = 2 THEN  GOTO 800
780  IF D = 3 THEN  GOTO 810
790  IF D = 4 THEN  GOTO 800
800  PRINT "YOU WON $";:C = C * 5
     : PRINT C: FOR PAUSE = 1 TO
     1500: NEXT PAUSE:B = B + C: GOTO
     180
810  PRINT "YOU WON $";:C = C * 1
     5: PRINT C: FOR PAUSE = 1 TO
     1500: NEXT PAUSE:B = B + C: GOTO
     180
820  PRINT "YOU LOST $";: PRINT C
     : FOR PAUSE = 1 TO 1000: NEXT
     PAUSE
830 B = B - C: GOTO 180
840  PRINT "YOU WON $";:C = C * 1
     00: PRINT C: FOR PAUSE = 1 TO
     1500: NEXT PAUSE:B = B + C: GOTO
     180
850  TEXT : HOME
860  IF A > B THEN  PRINT "SORRY,
      BUT YOU LOST $";:H = A - B:
      PRINT H
870  IF A < B THEN  PRINT "YOU WO
     N $";:H = B - A: PRINT H
880  FOR PAUSE = 1 TO 1500: NEXT
     PAUSE: PRINT
890  PRINT "DO YOU WANT TO PLAY A
     GAIN? (YES OR NO)": INPUT G$

900  IF G$ = "YES" THEN  GOTO 940

910  IF G$ = "Y" THEN  GOTO 940
920  IF G$ = "NO" THEN  GOTO 950
930  IF G$ = "N" THEN  GOTO 950
940  FOR PAUSE = 1 TO 150: NEXT P
     AUSE: HOME : VTAB (6): GOTO
     150
950  PRINT : PRINT "THANK YOU FOR
      PLAYING 'SLOT MACHINE'."
960  FOR PAUSE = 1 TO 750: NEXT P
     AUSE
970  GOTO 1360
980  REM  JACKPOT-1
990  COLOR= 8: PLOT X + 6,15: HLIN
     X + 6,X + 7 AT 15: HLIN X +
     6,X + 8 AT 16: HLIN X + 5,X +
     9 AT 17
1000  HLIN X + 4,X + 5 AT 18: HLIN
     X + 7,X + 10 AT 18: HLIN X +
     3,X + 4 AT 19
1010  HLIN X + 2,X + 4 AT 20: HLIN
     X + 6,X + 8 AT 20: HLIN X +
     2,X + 4 AT 21: HLIN X + 8,X +
     10 AT 21
1020  HLIN X + 2,X + 6 AT 22: HLIN
     X + 8,X + 10 AT 22: HLIN X +
     2,X + 4 AT 23: HLIN X + 8,X +
     10 AT 23
1030  HLIN X + 2,X + 5 AT 24: HLIN
     X + 7,X + 10 AT 24: COLOR= 1
     2: PLOT X + 6,18
1040  HLIN X + 5,X + 7 AT 19: PLOT
     X + 5,20: HLIN X + 5,X + 7 AT
     21: PLOT X + 7,22: HLIN X +
     5,X + 7 AT 23
1050  PLOT X + 6,24
1060  IF X = 0 THEN  GOTO 480
1070  IF X = 13 THEN  GOTO 600
1080  IF X = 26 THEN  GOTO 720
1090  REM  LEMON-2 OR 5 OR 8
1100  COLOR= 13
1110  HLIN X + 5,X + 7 AT 15: HLIN
     X + 3,X + 9 AT 16
1120  HLIN X + 2,X + 10 AT 17: HLIN
     X,X + 12 AT 18
1130  HLIN X,X + 12 AT 19: HLIN X
      + 2,X + 10 AT 20
1140  HLIN X + 3,X + 9 AT 21: HLIN

     X + 5,X + 7 AT 22
1150  IF X = 0 GOTO 480
1160  IF X = 13 GOTO 600
1170  IF X = 26 GOTO 720
1180  REM  CHERRY-3 OR 6
1190  COLOR= 1
1200  HLIN X + 4,X + 6 AT 16: HLIN
     X + 3,X + 7 AT 17
1210  HLIN X + 2,X + 7 AT 18: HLIN
     X + 1,X + 9 AT 19: HLIN X +
     1,X + 10 AT 20
1220  HLIN X + 1,X + 10 AT 21: HLIN
     X + 2,X + 10 AT 22: HLIN X +
     3,X + 9 AT 23: HLIN X + 4,X +
     8 AT 24
1230  COLOR= 4: HLIN X + 11,X + 1
     2 AT 15: PLOT X + 10,16: PLOT
     X + 9,17: PLOT X + 8,18
1240  IF X = 0 GOTO 480
1250  IF X = 13 GOTO 600
1260  IF X = 26 GOTO 720
1270  REM  ORANGE-4 OR 7 OR 9
1280  COLOR= 8: HLIN X + 6,X + 7 AT
     14
1290  PLOT X + 6,15: PLOT X + 6,1
     6: COLOR= 9
1300  PLOT X + 5,16: PLOT X + 7,1
     6: HLIN X + 4,X + 8 AT 17
1310  HLIN X + 3,X + 9 AT 18: HLIN
     X + 2,X + 10 AT 19: HLIN X +
     2,X + 10 AT 20: HLIN X + 2,X
      + 10 AT 21
1320  HLIN X + 3,X + 9 AT 22: HLIN
     X + 4,X + 8 AT 23: HLIN X +
     5,X + 7 AT 24
1330  IF X = 0 THEN  GOTO 480
1340  IF X = 13 THEN  GOTO 600
1350  IF X = 26 THEN  GOTO 720
1360  END
```

Word Processing The Easy Way

Warmed up Apple Pie... Software, like food, can be better the second time around.

by Sandra M. Abernathy and Timothy J. Pettibone

Pie Writer is an updated version of Apple Pie, a word processing program developed by Thomas Crosley and originally sold by Programma International. Hayden Book Company (50 Essex St., Rochelle Park, NJ 07662) recently acquired rights to the program, changed the name, totally rewrote the manual and added some modifications to the program to increase its versatility. All files written under Apple Pie are compatible with Pie Writer. A synopsis of the program changes has been included at the beginning of the manual for those who are familiar with Apple Pie. Pie Writer is unlocked and owners should make backups of the original disk before using the program.

With the new manual and the tutorial on disk, a novice to the world of desktop computers can learn to produce reports, charts, tables and letters quickly and easily. In addition, the program is powerful enough to satisfy most users' needs. The ability to produce personalized form letters, mailing labels and to communicate with other computers may be of particular interest to many readers.

Pie Writer requires an Apple II microcomputer with Applesoft and 48K memory as well as a disk drive and printer. It can support multiple disk drives, special types of printer interface cards and printers, lowercase and shift key adaptors, 80-column boards, and a doubletime spooler. Underlining and boldface printing are possible if the user's printer has these capabilities.

The Manual

The quality of documentation and its relative ease-of-use is an important part of any complicated piece of software. Hayden Book Company has arranged Pie Writer's manual so that needed information can be found easily by using the chapter tab feature, or by examining the table of contents, the index or the reference card. The latter is a list of all command processor, text editor and text format commands. In addition, a Help program has been included on the disk.

Chapters are arranged in order of user expertise. Ideas are explained clearly throughout, but more detailed explanations are found in earlier chapters. The first three chapters, Introduction (turning on the Apple and accessing the System Menu), Pie (editing text, loading and saving files) and Format (formatting text and printing), along with information on the disk, are an extensive tutorial to introduce the new user to the word processing system. Upon completion of these chapters, the user should be able to perform routine word processing tasks.

Information within these chapters is organized for quick reference. A single idea is presented and discussed; the user is directed to practice the idea on the computer, and then the command and its definition are summarized. All commands are printed in boldface type throughout the manual to help the user spot them easily.

The fourth, fifth and sixth chapters, Configure, Reference and Advance Topics, are the three that a knowledgeable user would turn to first. Configure contains information to customize Pie Writer to fit the user's particular hardware and printing needs. The two configure programs on the disk are completely menu-driven and each option is clearly explained in the manual. The user can return to the configure programs and modify information as needed. Pie Configure allows the user to enter information

Timothy J. Pettibone, Ph.D., Professor of Educational Management and Development, has taught courses in computer applications and management technology for the past several years as well as giving workshops for teachers and administrators. In addition, he is the publisher/editor of Software Critic.

Sandra M. Abernathy, Ph.D., teaches courses in educational applications of computers and gives workshops for teachers and administrators in the public schools and at the post-secondary level. In addition, she uses her personal Apple II Plus for developing materials and for consulting projects.

"The program has not been changed radically. However, there is one new feature that may save the user hours of grief."

about the interface card used with the printer, lowercase display, shift key modification, doubletime spooler, bell column, tab placements and other Pie features.

Format Configure permits the user to take advantage of the special features of particular printers and to set overall page specifications. If the user has a Diablo 1620, Qume Sprint 5, NEC Spin 5510, Epson MX-80 or Centronics 737, one keystroke will automatically enter the necessary information for the particular printer. If the user has a different printer or wishes to override the standard commands, information about printer initialization, high bit set on output, line feed after return, underline and boldface capabilities, and page stop may be entered manually.

Page formatting is also part of the Format Configure program. Top, bottom and side margins as well as heading and footing margins, right margin justification, page length, line length, line spacing, paragraph indentation and spacing and paragraph need value can be set here. All text format settings can be overridden within a specific text by entering imbedded commands within that text. The latter ability enables the user to set general page formats and not have to return to the Configure program if a particular text needs different settings.

Chapter Five, Reference, is an extensive summary of all Command Processor, Text Editor and Text Format commands used during text production and printing. Specific information in the chapter can be located quickly by looking in the Table of Contents or the Index. Information contained in this chapter will be discussed in the following section.

The final chapter, Advanced Topics, discusses the production of personalized form letters and mailing labels; the programmed interruption of the printing process to change print wheels, etc. or to enter information from the keyboard; and the use of shell files for frequently used functions and the transmission of files between computers. Information about a shift key modification and address tables are also found in this chapter. The latter is included to aid those who wish to write their own programs to use in conjunction with Pie Writer.

The Program

Pie Writer consists of the two configure programs discussed earlier, the Pie Text Editor program, where all text entry, editing and formatting as well as loading and saving of text material takes place and the Format Text Processor program that sets up printing options and initiates printing. The latter program, like the Configure programs, is menu-driven and allows the user to specify the number of copies to be printed, whether the whole or a portion of text is to printed and whether printing is to be continuous or a page-by-page process. Files can be called from the disk or from the computer's memory and can be "printed" to the monitor, the printer or the disk. A printed copy can be any length since a series of binary files can be printed sequentially if an imbedded command is added at the end of each to indicate the next file.

"Hayden rewrote the manual and added some modifications to the program to increase its versatility."

The text editor program has not been changed radically. However, one feature has been added to the current program that may save the user hours of grief. The command processor now displays the name and size of the file currently in memory as well as the amount of memory still remaining. Armed with this information the user no longer has to guess the name of the current file when it is time to save. This feature should also keep the user from saving the current file under the name of another, existing file on disk, thereby erasing the other file. In addition, if the user wishes to save the modified text under the original name, he need only type "S?" and the file will be saved under the displayed name. This feature eliminates the problem of typing the name incorrectly.

Pie Writer has several other additions that make the program more useful. One, you can now write comments during text entry that will not be printed. This feature can be used to specify the previous and succeeding files in a multifile product. Two, the literal text, complete with imbedded commands, can be printed out. This is helpful as a reference if a complicated or little-used sequence of commands is used to produce a particular format. Three, the formatted text can be saved to disk. This may be useful if you wish to send a formatted copy of the text to another computer at a later time. Four, when editing text, you can now move from the end of one line to the beginning of the next and vice versa much more easily.

Pie Writer has a versatile and powerful text editor. The program is not menu-driven as are the others in the system. This means that text entry as well as text editing and formatting can be accomplished without moving to different parts of the program. The lack of good documentation had made the earlier version, Apple Pie, difficult for the beginner to use. The present documentation has corrected the problem and now information about commands is easy to find and understand.

After entering Pie Text Editor, the command processor portion of the program will display the current file name, the length of the file, the amount of memory left for further text entry and the word, Command, followed by a question mark. From this point the user can load and save text material in either text files or binary files, enter a file that has been loaded into the computer's memory, begin a new file, call the Help program, the catalog or a machine language routine, enter the Apple system monitor, transfer to the Text Format program, or quit.

Text is displayed within a window during text processing. The window consists of a rectangle 21 lines long by 38 columns wide, or 78 columns for computers with an 80-column board. A status line is displayed beneath the window and shows information about the line number, type of entry mode and other information. As the user types in text, lines scroll up and off the screen. About thirteen to fourteen double spaced pages of straight text can be stored in a single binary file.

There are three text entry modes, Manual, Wrap (PPWRAP) and Indent, which can be used as needed. Manual mode is used to enter tables and other information that must be printed as entered. Current column numbers are displayed on the status line to aid the user. A total of 64, or 128 columns for 80-column boards, are available to you by moving between two horizontal, overlapping windows.

Wrap mode is used for straight text entry. The user does not move between windows when using this mode because words are automatically wrapped around to the next display line when there is not enough room at the end of a line. The return key is pressed at the end of each sentence to produce a double space between sentences in the printout. When printed, the lines will be the length prescribed by format commands.

Indent mode is used to set automatic hanging indents for special types of text entry such as outlines and some types of programming. Hanging indents can also be set up with imbedded commands when in Wrap mode. Text entry modes can be exchanged easily by pressing appropriate keys.

Commands are accomplished by pressing appropriate keys to move the cursor, insert and delete characters or lines, search and find or search and replace characters, move or copy blocks of text and automatically insert preprogrammed character strings during text entry as well as split and rejoin lines. These commands are fairly logical keystrokes. For instance, control-I inserts a line, control-P inserts or deletes characters within a line, a diamond pattern of four keys, E (up), S (left), C (down), F (right), along with the control key allow the user to move the cursor horizontally or vertically over the text to arrive at a location. Other control commands allow the user to move the cursor from one end of the display line to the other or from one screen to another with a single keystroke sequence. An advantage of Pie Writer is that the user can perform the same function in a variety of ways. The beginner, with a limited Pie Writer vocabulary, is capable of accomplishing the same task as an advanced user, although possible in a less efficient fashion.

Imbedded commands are used to signal special formatting within the text. Commands are placed on a line by themselves, begin with a period, and signal a change in format for one or more succeeding display lines. Examples of frequently used imbedded commands are .IN (indent), .BP (begin new page), .PP (begin paragraph), and .CE (center), .LL (line length) and LS (line space). Some imbedded commands use numbers to indicate the number of lines that should be processed using this command; .CE is one of these. For instance, .CE 2 would command the next two display lines to be centered. Unless a number is typed after the command, the number is assumed to be one. Other commands set up a temporary format until a further command is written. For instance, .IN 5 moves the left margin five columns to the right until another .IN is encountered; .IN +3 moves the margin an additional three spaces to the right; .IN – moves it to the left three spaces and .IN restores the original margin.

Summary

Pie Writer has very good documentation, an excellent software program, the ability to produce personalized form letters and mailing labels and to send and receive text files, as well as an attractive price ($149.95). The owner has the ability to back up copies of the disk for personal use. Two features Pie Writer lacks, which may be important to some readers, are ghost hyphenation and the ability to produce double column text. Very few word processors have all the features a user desires. For those familiar with Apple Pie, the new Pie Writer will prove to be a more convenient and versatile tool particularly because of the printer support and the text name display during the save procedure. The manual is certainly a tremendous improvement over the old one. For those looking for a first word processing system, Pie writer, with its excellent tutorial and powerful text processing system, should be given serious consideration. ■

MICROTEK
COMPUTER PRODUCTS

IF YOU HAVE A GRAPHICS PRINTER YOU MUST HAVE A GRAPHICS INTERFACE.

DUMPLING-GX
GRAPHICS PRINTER INTERFACE

The Dumpling-GX is a DIP Switch Selectable Dual Hi-Resolution Graphics Screen Dump parallel interface card for Apple computers and most popular printers.

At the flick of the DIP switch, The **Dumpling** will interface with: **APPLE ▪ EPSON ▪ NEC ▪ IDS* ▪ ANADEX ▪ C-ITOH ▪ PMC ▪ CENTRONICS ▪ OKIDATA ▪ MANNESMANN TALLEY**

Selectable Strobe and Acknowledge polarities allow use with **any** 8-bit parallel printer in text and block graphics mode.

Microtek's proprietary on-board firmware enables the **Dumpling-GX** to establish intelligent communication between your Apple computer and your printer. Simple commands allow:

- Selection of your printer by DIP switch.
- Selective Dump Page 1, Page 2, or both in either text or graphics mode.
- Chart Recorder Simulation.
- Left & Right Margin Control.
- Line Length/Page Length Selection.
- Block graphics via 8th bit Control.
- Printer bell Control.
- Skip over Perf.
- 90 degree Rotation.
- Double Size Graphics.
- Emphasized Graphics Print.

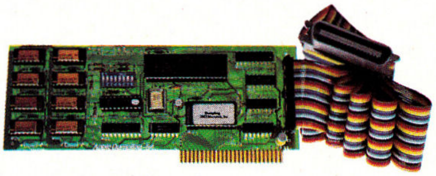
*With special PROM and cable.

DUMPLING-64
GRAPHICS PRINTER SPOOLER

The **Dumpling-64** is the next logical extension to the industry standard Dumpling-GX parallel interface card, allowing the computer to DUMP vast quantities of data into the **Dumpling-64** for later printing, thus freeing up the computer for additional tasks.

The **Dumpling-64** allows full use of all **Dumpling-GX** features. In addition to the standard graphics features, the **Dumpling-64** offers:

- Buffer sizes from 0 K to 64K. User upgradable.
- Graphics Dumps to Buffer. Page 1 and/or 2.
- Multiple Consecutive Screen Dumps to Buffer.
- Software reset to clear Buffer.
- "Space Compression" saves valuable memory taken up by 'spaces' in text or spread sheets.
- Automatic Buffer Size Recognition.
- Pause while printing-immediate.
- Pause while printing-delayed.
- Resume printing.
- REMOTE pause-immediate: hooks up to telephone, switches—etc.
- Buffer ON/OFF control.
- INSERT text editing capability with Pause and Buffer ON/OFF control.



MICROTEK inc.

TECHNOLOGY UPDATE

Circle 329 on Reader Service card.

Compute While You Print With the Buffer that

MICROFAZER PUTS YOU BACK TO WORK

Your computer helps you work fast. Unless the printer is running. Then it doesn't help you work at all. It won't let you enter data or process information. It simply won't compute.

That's where Microfazer by Quadram comes in. It's the print buffer that frees your computer. And lets you keep right on working.

EXPANDABLE TO 512K

THE BUFFER THAT REMEMBERS IT ALL

Microfazer stores data from your computer in its own memory, then sends it to the printer at the proper rate.

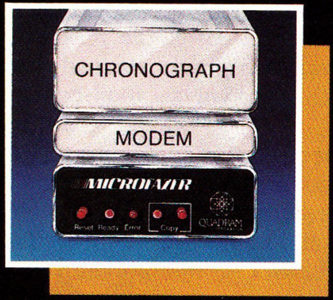
You don't have to worry about losing vital information because of limited buffer space. Because Microfazer starts with 8K of memory and is expandable to 512K—a full half-megabyte. So it can keep pace with your needs. Now, and in the future.

THE ANY COMPUTER, ANY PRINTER BUFFER

Microfazer is perfect for any buffer task. Word Processing. Accounting. Graphics.

Print with Microfazer,™ Remembers It All.

You name it. And it's perfect with any enhancement. Printers. Plotters. Even modems.



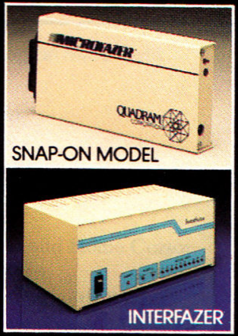
You'll find Microfazer in a variety of models and sizes. Some stand alone while others are stackable. There's one that snaps onto the back of the popular MX Series Epson printers. And another that plugs inside an Epson MX or FX.

There's a Microfazer to interface incompatible devices. And for any data transmission need. Serial or parallel.

QUADRAM REMEMBERS TOO
Whatever your system, Quadram has a buffer to handle it. Including Microfazer's counterpart: Interfazer, for buffering and controlling up to eight computers with one or two printers.

Quadram didn't forget the hardware features either. The Ready LED, manual Reset and Pause/Copy buttons are all part of the Quadram Quality tradition.

A PRICE YOU'LL GO FOR
You'll be glad to know that you can get Microfazer backed by Quadram Quality at a price that won't stop you from owning one. Parallel to parallel versions start at $159 (8K). Serial to parallel, parallel to serial and serial to serial versions start at $199.



MICROFAZER, THE PRINT BUFFER THAT REMEMBERS IT ALL.

QUADRAM CORPORATION
An Intelligent Systems Company
4357 Park Drive / Norcross, Ga. 30093 / (404) 923-6666
TWX 810-766-4915 (QUADRAM NCRS)

Circle 234 on Reader Service card.

ORCA/M:
A Powerful New Tool For Software Development

Shift into high gear with this new assembler package that makes machine coding as easy as Pascal.

by Randel B. Reiss

A few months ago I decided to do what any healthy red-blooded American teenager wants to do. I decided to start my own computer game software company. I was careful to look into all the available assemblers. I didn't want to start work developing the next Pac-Man unless I knew I was using the best assembler there was.

Parker Brothers had just started developing for the Atari VCS using a Digital VAX unit with an incredible 6502 assembler and a six-digit price tag. For a while I had used various macro assemblers but still envied the power my little Apple system lacked. Hayden Software heard my cry. "The Greatest Assembler in the World" is how the ads read. Doubtful? Don't be. The new Hayden ORCA/M macro assembler package finally does justice to the hidden power of the Apple computer and provides the full power of a mainframe system to your home computer.

I brought home my demo copy of ORCA/M and, being a true computer hacker (not reading the instructions), began to examine the files on the three accompanying disks. I was impressed to find that full source code on the system was at my disposal, including

Address correspondence to Randel B Reiss, 15 Temple St., W. Roxbury MA 02132.

Table 1. Macro library.

FLOATING POINT	
Absolute Value	Arc Tangent
Character Input Integer	Convert to 4 Byte Integer
Cosine	Exponent
Add	Divide
Change Sign	Convert to Floating Point
Convert from 4 Byte	Multiply
Input	Output
Subtract	Integer Function
Natural Logarithm	Raise to a Power
Random Number	Sign Function
Sine	Square Root
Tangent	Convert to Integer
Set Floating Point Error	
DOUBLE PRECISION FLOATING POINT	
Convert Double to Single Precision	Convert Single to Double Precision
Absolute Value	Addition
Tangent	Change Sign
Cosine	Division
Exponent	Convert to Integer
Convert to a 4 Byte Integer	Convert from a 4 Byte Integer
Convert from Integer	Integer Function
Natural Logarithm	Multiply
Input	Output
Raise to a Power	Sign Function
Sine Function	Square Root
Subtract	Tangent
INTEGER MATHEMATICS	
2 Byte Add	4 Byte Add
Convert 2 Byte to 4 Byte	Convert 4 Byte to 2 Byte
2 Byte Divide	4 Byte Divide
2 Byte Integer Absolute Value	4 Byte Integer Absolute Value
2 Byte Input	2 Byte Integer Output
2 Byte Sign Function	2 Byte Square Root

Table continued.

Table continued.

4 Byte Input	4 Byte Output
4 Byte Sign Function	4 Byte Square Root
2 Byte Modula Function	4 Byte Modula Function
2 Byte Multiply	4 Byte Multiply
2 Byte Subtract	4 Byte Subtract
LOGIC AND BRANCHING	
2 Byte Arithmetic Shift Left	Branch If Less Than or Equal
Branch If Greater Than	2 Byte Unsigned Compare
Decrement and Branch Not Equal	2 Byte Decrement and Branch Not Equal
Decrement and Branch If Plus	2 Byte Logical Shift Right
2 Byte Decrement	2 Byte Increment
Jump If Carry Clear	Jump If Carry Set
Jump If Equal	Jump If Greater Than
Jump If Greater Than or Equal	Jump If Less Than or Equal
Jump If Less Than	Jump If Minus
Jump If Not Equal	Jump If Plus
Multiple Arithmetic Shift Left	Multiple Logical Shift Right
INPUT AND OUTPUT	
Write Character	Issue Return
Set Flashing Character	Get a Line
Horizontal Tab	Input with Prompt
Set Inverse Character	Set Normal Character
Print A,X as Hex	Print Blanks
Print Accumulator as Hex	Clear Screen and Home Cursor
Print 'ERR'	Print Hex Digit
Set Prompt Character	Print Character with Return
Print Centered Character	Print Character without Return
Print Number String	Read a Character
Vertical Tab	Set Window Bottom
Set Window Left Edge	Set Window Top
Set Window Width	Write Character with Return
Write Centered Character	Write Character without Return
LOW RESOLUTION GRAPHICS	
Clear Screen	Set Color
Read Graphics Screen	Draw Horizontal Line
Plot a Point	Draw Vertical Line
Set Display to Low-Res Graphics	
HIGH RESOLUTION GRAPHICS	
Clear Graphics Page	Set Color
Draw a Line	Position the Cursor
Display High-Res Graphics	Set the Page to Draw On
Plot High-Res Point	
MISCELLANEOUS MACROS	
Sound Apple Bell	Read a Button
Define a Word	Load Address
Load Memory	Move Memory
Set Display to Page 1	Set Display to Page 2
Read a Paddle	Restore Registers
Save Registers	Display Text
Pause for a While	

macro libraries (see Table 1). Eventually resorting to reading the manual I began to realize that ORCA/M was in fact a true environmental operating system that made machine coding as easy as Pascal. The user has full control of DOS functions such as CATALOG, RENAME, DELETE and COPY, all with a wildcard option. Deleted files can even be RESTORED if the implied disk sectors have not been damaged. Disk sector tracks can be read, changed and written to, all without leaving the ORCA/M monitor (see Table 2).

Finally it came time to type up some source code and see what this assembler could do. I jumped into the system editor. I wish that all text editors had the same function keys so I wouldn't have to get used to a new command set every time I tried to use a new editor.

> "What would you expect from the people who brought the world Pie Writer—of course, powerful source code editing."

ORCA/M's commands are given in Table 3. But it certainly was a pleasure to come from using Merlin's line editor to a full page text editor. What would you expect from the people who brought the world Pie Writer—of course, powerful source code editing.

As I began to use the system environment I became more familiar with it and discovered some of its advantages and faults. One of the drawbacks of the system is the need for two disk drives to easily operate the system from editor to assembler to linker. This brings me to an interesting asset of the system: it lets the user reassemble the operating system to custom fit his system's disk drives, printer, 80-column board, or a calendar/clock card.

Talk about software support—a utilities package will soon be on the shelves which includes a disassembler and other tools. A mini version of the ORCA/M package at half the price without linker and macros will soon be available. An enhancement kit for the mini version will be available, containing the linker and macros for the paid difference. There is also talk of an incredibly efficient Pascal compiler in the works, which will be compatible with the ORCA/M operating system.

ORCA/M has a large inventory of macros in its library, making programming easier for both the experienced programmer and novice user. ORCA's macros collection is mathematically inclined and lacks graphics support. This lack of graphics support is compensated for by the creation of user macro definitions, which are simply created in a macro library and easily called from a program and inserted by ORCA during assembly.

ORCA/M's ability to allow subroutine modules proves to be a helpful asset in assembling programs. It allows assembly of separate subroutines to avoid reassembly of an entire program. Implemented in its unique linking system is the distinction of local labels that are only valid in the subroutine they are declared for, and global labels which may be accessed by any of the program subroutine modules. You can use duplicate label names in separate subroutines with no symbol table conflict.

One of ORCA's most outstanding features is its ability to move the monitor and operating system out of core before assembly and back again on subroutine link completion. This, along with the creation of relocatable program files during assembly, means no more worry about writing over the operating system. ORCA is a disk-based system and immediately saves to disk binary and root files during linking. This easily allows programs of any length with no worry about space configuration and writing over your assembler.

The ORCA/M system was not designed as a learning tool—don't expect to learn assembly language from the system. The user instead should be very familiar with assembler procedures and 6502 source coding. The operator's manual is straightforward and very technical; Hayden tells me the manual is still in the working stage. The manual explains how each function of the ORCA environment works, without using enough examples of how features and functions are used in actual situations. This lack of examples led to a lot of time-consuming trial and error programming

Two important things occur during the assembly error detection. The first is that when the assembler encounters an op-code it does not recognize, it immediately checks the system's macro library including user-defined macros. Second, after an error has been detected, the assembler loads the monitor back into memory and places the program in the editor at the point at which the error occurred. This allows the programmer to correct the error before it happens.

ORCA/M is a serious macro assembler that gives developers large system assembler techniques on the Apple. There's no doubt in my mind that ORCA/M will become the number one product for 6502 source code generation for R & D work at the personal level, and at most major software houses. ■

Append File	Assemble and Link
Assemble, Link and Go	Assemble File in Memory
Execute File	Catalog the Disk
Compile and Link	Compile, Link and Go
Check a Disk	Compile File in Memory
Compress the Disk	Copy Files
Delete File	Boot DOS
Soft Entry to Editor	Expand Load
Display Core Used	Load File
Link Edit	Lock File
Set Left Margin	Hard Entry to Editor
Don't Expand Load	Examine/Change Disk Sectors
Print File	Turn Printer Off
Turn Printer On	Exit the Assembler
Rename File	Reset Modification Count
Restore File	Assemble, Link and Execute
Save File	Switch Files
Set Tab Line	Print Time and Date
Unlock File	Set Volume Number
Call a User Subroutine	

Table 2. Monitor instructions.

Tab Right	Shift Lock
Home to Bottom of Page	Pop Buffer Lines
Search and Replace Up	Quit
Clear to End of Line	Remove Blank Lines
Home to End of Line	Tab Left
Home to Top of File	Home to Top of Page
Page Tab	Search and Replace Down
Toggle Display	To Start of Line
Left Bracket	String Search Down
Home to End of File	Switch Cursor Mode
Switch Return Mode	String Search Up
Insert a Line	Scroll Down a Page
Scroll Screen Down	Scroll Up a Page
Scroll Screen Up	Delete a Line
Display Core Used	Left Curly Bracket
Delete a Character	Umlaut
Insert a Character	Backward Slash
Move Cursor Up	Underline Character
Move Cursor Left	Enter Search String
Move Cursor Right	Enter Replace String
Move Cursor Down	Insert Buffer Character
Delete Lines to Buffer	Delete Character to Buffer
Copy Lines to Buffer	

Table 3. Text Editor commands.

	ORCA/M	MERLIN
EDITOR	Full Page Editing	Line Editing
MACROS	Double Precision Floating Point Integer Miscellaneous (Graphics, etc.) User Defined	Miscellaneous User Defined
BASED ON	IBM370	TED II +
SYSTEM SOURCE CODE	Included	Not Included
UTILITIES (DISASM, ETC.)	Available Soon	Included
USER LEVEL	Expert	Intermediate
RETAIL PRICE	$99.95	$64.95

Table 4. ORCA/M-MERLIN comparison.

Apple Repair and Care

Why truck your Apple down to the repair shop when you can handle some repairs yourself? These troubleshooting tips will get you started.

by Matteo Leona and Earle Hancock

We have all experienced that hollow, sinking feeling when a piece of equipment malfunctions. Did you ever notice how the breakage occurs just when you need the equipment the most? Microcomputers fail to work the way they are supposed to as often as any other piece of machinery. But since the microcomputer is very powerful, the resulting inconvenience is equally powerful.

Over the past few years we have experienced our share of microcomputer failures and have worked out solutions to many of the problems. Here are a few common, unpleasant situations and how to handle them.

● *When I attempt to boot my favorite disk on my Apple II Plus, the disk just spins and spins in the drive. This disk worked last month. What can I do?*

First check to make sure that your disk is not formatted in DOS 3.2. If your format were 13-sector, you would have to use the Boot 13 program on your system master to get the disk running.

If your disk is 16-sector, perhaps it isn't centered correctly in the drive. The read/write head in a disk drive only reads disks turning in perfect circles. If your disk isn't centered correctly, it will turn in an ellipse and be impossible for the read/write head to decipher. Try gently opening and closing the disk drive door while the disk is spinning to center it.

Perhaps your disk is ruined due to dirt, mishandling or just old age. Sometimes mishandling can result in tracks 0, 1 and 2 being wiped clean or scrambled. Try to get around these bad tracks by booting your system master disk and then running your program from the damaged disk. Tracks 0, 1 and 2 contain the Disk Operating System or DOS. Without DOS your computer cannot get the program from the disk. Booting the system master installs DOS in the RAM, ready for use. A true computer enthusiast could examine the contents of the malfunctioning disk to see what is wrong, using a program like Bag-of-Tricks. But most of us will just call for help.

If none of these suggestions works, your disk drive may be out of alignment. The alignment can be adjusted at home, but is better left to an authorized Apple dealer.

Occasionally, the speed of the drives varies enough to cause problems. Programs like Locksmith 4.1, Brain Surgeon and The Filer may help you check and correct the speed, but please note that removing your disk drive cover to make speed adjustments could void your warranty.

When you have a disk/drive failure, do not reach automatically for your backup disk to see if that works. Sometimes a drive fails in a way that removes all information on the disk as it spins. You don't want your only backup to be in that drive until the drive is adjusted.

● *My Apple won't turn on. What shall I do?*

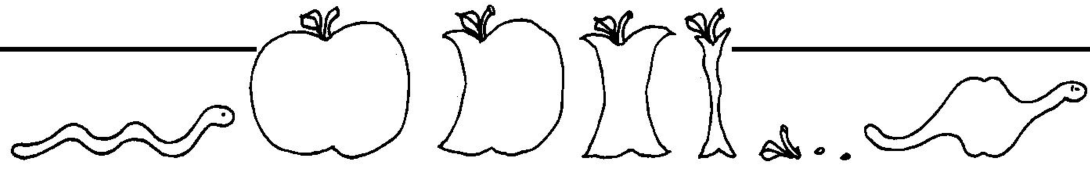
First make sure the computer is plugged into a functioning wall socket. Try clicking the on/off switch a few times. If it still does not turn on, there may be a problem with the power supply.

Do *not* attempt to fix the power supply yourself; the high voltages are *dangerous*. Also, Apple will penalize you for tinkering with the power supply by charging more to replace the total component. Bring your Apple to an authorized dealer.

● *When I boot my disk, strange nonsense characters appear on the screen. What shall I do?*

Make sure that the vertical hold on your CRT is adjusted properly. If it appears correct, try shutting off the computer and booting the disk again; the Apple gnome may have played a

Matteo Leona and Earle Hancock are both involved in vocational education. Address correspondence concerning their article to Matteo Leona, Winthrop 15, Swanson Road, Boxboro MA 01719.



practical joke on you. If the video still comes out in gibberish, try another program disk. Should this new disk work, then you know that the problem is a destroyed disk. However, if this new disk also yields nonsense, your computer may have a defect in ROM or RAM. Take it to the repair shop.

● *The letter "t" on my Apple keyboard will not register on my monitor screen. Why does this happen? What can I do?*

The switch connected to the letter "t" has broken. An authorized Apple dealer can easily fix the problem. Should you tamper with the keyboard on your own but ultimately have to take it to the shop, Apple may charge you more.

If many keys on your keyboard do not work, open your microcomputer and look under the keyboard to see if the ribbon cable is plugged into the motherboard properly. Just press the cable gently with your finger to make sure it is seated tightly.

● *Everywhere I look I see ads for fans and filters for my Apple. Are these things really necessary?*

If you use your microcomputer for any length of time, the integrated circuits or chips heat up and expand. This is especially true when many peripheral cards are plugged into the back of the Apple, inhibiting air flow and causing more heat output from the power supply. When you turn the computer off and it cools, these same parts contract. This expansion and contraction will eventually cause the components to malfunction. Fans ensure that your computer keeps cool and works smoothly.

Powerline conditioner filters provide uniform voltage to the microcomputer. Electricity from your wall socket varies with power surges. Filters assure the user of an even energy supply. Additionally, filters remove powerline electrical noise that may cause parts of your microcomputer to misbehave.

"Treat your microcomputer and software disks as you would an expensive stereo record player and fine records."

● *The printer for my Apple is filthy. Little pieces of paper appear to be jammed in the paper feed. What can I do to clean it up?*

Tiny paper fibers can clog your printer and cause it to malfunction. You can take it to an authorized repair shop for periodic cleaning, or attempt to clean it yourself.

You'll need filtered high-pressure air (from 50 to 75 pounds). Holding the air gun 8 to 12 inches from the printer, very carefully blow away all dust and dirt. Pay specific attention to the flexible electrical connections so you don't damage them. After cleaning, spray all moving parts with a low flashpoint silicone lubricant spray.

● *Can you please provide a summary of general care and maintenance guidelines for my Apple microcomputer?*

You should always treat your microcomputer and software disks as you would an expensive stereo record player and fine records. Remember that dirt and dust are dire enemies of computers. Static charges in the computer itself and the CRT attract dirt. They should both always be covered when not in use. Attempt to maintain a dust-free environment for your system. Keeping the windows in the room closed will help. Before the advent of microcomputers, mainframes and minis required special air-conditioned, dust-free rooms. While today's microcomputers are not as delicate and can function in most normal living conditions, reasonable care should still be taken to keep dust and dirt away.

Avoid eating, drinking and smoking at the computer. Clean the monitor screen and all external computer parts periodically with a nonabrasive window cleaner and a soft cloth or paper towel.

Clean the read/write heads in your disk drives by using one of the several head cleaning kits available on the market. These kits consist of a special disk that you coat with cleaning fluid and run through the drive.

Dirt can also destroy disks. Do not put your fingers on the exposed read/write slot. When not in use, always store disks in their sleeves. Don't leave them on a CRT or on the dashboard of your car. They are susceptible to extreme temperatures.

Disks can also be scrambled or destroyed by magnetic fields. Keep them away from ringing telephones, stereo speakers and magnetized screwdrivers. Your CRT also emits magnetic fields, so disk drives should always be located at least 6 inches away from it.

One last word of caution—never use lubricants on the inside of a disk drive. An authorized service person might lubricate some of the internal mechanism, but the average microcomputer user should not attempt this rather delicate operation. ■

Small Packages

The secret behind making Pascal work for you is to neatly structure your units of code. Structure makes it happen!

by John Stephenson

Units, or precompiled support modules, are the UCSD Pascal programmer's great efficiency tools. Just as programmers create unique data types within modules to express and solve application problems, so they create and use specialized libraries of units aimed at generic tasks. Using units reduces the amount of code that needs to be compiled during individual program development, limits the amount of code that needs recompilation during maintenance, and improves communication between many programmers working on a single project. Time invested isolating common subtasks of different application problems and coding them in unit form yields a high return.

What Is DataOps?

DataOps is an all purpose data processing module. Perhaps the most common element of application programming is management of data entry and formatting. Numeric lists need to be justified. Phone numbers, dates and the like need to be shown in a consistent manner. Text needs to be precisely placed on the screen. The data entry process must be controlled and robust. Validity must be assured. Unanticipated user input should never cause a system error message except, possibly, from use of the break key. Unit DataOps attempts to address these problems in a general manner useful to many programs.

Like all units, DataOps consists of an *interface* section and an *implementation* section. The interface section defines what services DataOps can provide; the implementation section defines how DataOps will provide them. Later optimizations of the "how" part require no modification to programs. The only effect will be improved operation.

> **"The only effect will be improved operation."**

What Kind of Data?

DataOps processes fundamental data types and special data types. The specials are money, dates, telephone numbers and Social Security numbers. The fundamentals are single ASCII characters, integers, reals and strings. Strictly speaking, a string is not a fundamental (nonreducible) data type, but an array of characters prefixed with a length byte. However, since strings are predeclared in UCSD Pascal, it is convenient to think of them as fundamental.

How Does DataOps Accept Data?

DataOps accepts data as an item described by a series of attributes. The attributes streamline processing. They inform the unit and program about the screen position for display, how to justify the item within its field position, what default value may be assigned, what special image the data might conform to, and what specific Pascal data type is being processed. DataOps packs this information into one record structure, which contains variant subrecords to handle different types.

The powerful variant record structure supported by UCSD Pascal is crucial to DataOps. The variant record allows the program to interpret the same chunk of memory as different data, depending on how that memory is referenced by name. Programmers use variant records for a variety of reasons, such as accessing machine-specific memory locations (systems programming) or converting data from one type to another.

Address correspondence to John Stephenson, 9118 Smith Ave., N. Bergen, NJ 07047.

```
                    (*S+,G+)
               UNIT dataops;
           INTRINSIC code 26 data 27;
   {$c copyright 1982 John C. Stephenson }

                    INTERFACE
TYPE
        money          = INTEGER[16];

        daterec        = PACKED RECORD  { corresponds in structure to }
                month  : 00..12;        { to date records used in      }
                day    : 00..31;        { various parts of the UCSD    }
                year   : 00..100;       { p-System.                    }
                END;

        phonerec       = PACKED RECORD
                prefix   : STRING[1];
                area     : STRING[3];
                number   : STRING[7];
                extension:STRING[4];
                END;

        justifytype    = (dontjustify,
                          leftjustify,
                          centerjustify,
                          rightjustify,
                          securejustify);

        resulttype     = (chardata,
                          datedata,
                          integerdata,
                          moneydata,
                          phonedata,
                          realdata,
                          ssndata,
                          stringdata);

        maptype        = (sizedata,
                          picturedata);

        maprec         = PACKED RECORD
                CASE kind:maptype OF
                        sizedata:       (size:INTEGER);
                        picturedata:    (pic:STRING);
                END;

        resultrec      = PACKED RECORD
                CASE kind:resulttype OF
                        chardata:       (c:CHAR);
                        realdata:       (r:REAL);
                        integerdata:    (i:INTEGER);
                        moneydata:      (m:money);
                        datedata:       (d:daterec);
                        stringdata:     (s:STRING);
                        ssndata:        (ssn:STRING);
                        phonedata:      (phone:phonerec)
                END;

        origintype     = (userdata,
                          programdata);

        datarec        = PACKED RECORD
                errcode : INTEGER;                   { -1 pressed ESC key }
                xpos    : 0..79;                     { 80 column screen   }
                ypos    : 0..23;                     { 24 row screen      }
                justify : justifytype;
                origin  : origintype;
                default : STRING;
                map     : maprec;
                result  : resultrec;
                END;
VAR
        securefill,
        fieldmarker    : CHAR;
        outputstring   : STRING;      { holds last output string      }

PROCEDURE input (VAR d:datarec);
PROCEDURE output (d:datarec);
                    IMPLEMENTATION

CONST
        title = 'UNIT dataops [1/8/83] (c) 1983 John C. Stephenson';
TYPE
        charset        = SET OF CHAR;
VAR
        digitset,
        specialset,
        printset       : charset;

(*$I LISTING3.TEXT*)
(*$I LISTING4.TEXT*)
(*$I LISTING5.TEXT*)
(*$I LISTING6.TEXT*)
(*$I LISTING7.TEXT*)
(*$I LISTING8.TEXT*)
(*$I LISTING9.TEXT*)
(*$I LISTING10.TEXT*)
(*$I LISTING11.TEXT*)

                    BEGIN (UNIT dataops)
                    WRITELN(title);
                    outputstring:='';
                    fieldmarker:='_';
                    securefill:='*';
                    END.
```

Listing 1. Unit DataOps intrinsic code.

Listing 2.
Program example using DataOps.

```
                              {$S+,G-}
                          program example;
                           uses dataops;

var
        sampledata      : datarec;
        qty             : integer;
        each,total      : money;

begin
        gotoxy(0,4);
        writeln ('Example of using unit DataOps');
        { set up the screen display }
        gotoxy(15,10); write('qty');
        gotoxy(20,10);write('price each');
        gotoxy(32,10);write(' total amt');
        { pass necessary attributes to enter an integer from -99 to 999 }
        sampledata.xpos:=15;
        sampledata.ypos:=11;
        sampledata.justify:=rightjustify;
        sampledata.origin:=userdata;
        sampledata.default:='';
        sampledata.map.kind:=sizedata;
        sampledata.map.size:=3;
        sampledata.result.kind:=integerdata;
        { Let user keep trying until he gets it right.  Not the friendliest
          way to treat him! }
        repeat input(sampledata) until sampledata.errcode=0;
        qty:=sampledata.result.i;
        { Pass necessary attributes for a money amount up to ten digits long.
          Some attributes are the same as last time so no need to reset them. }
        sampledata.xpos:=20;
        sampledata.map.size:=10;
        sampledata.result.kind:=moneydata;
        { Input until user gets it right. }
        repeat input(sampledata) until sampledata.errcode=0;
        each:=sampledata.result.m;
        { Calculate the total.  Set attributes for its display. }
        total:=qty * each;
        sampledata.xpos:=32;
        sampledata.result.m:=total;
        { Display it. }
        output(sampledata);
        { And, end the demo. }
        gotoxy(0,5);
        writeln ('End of demo program.');
end.
```

In the Apple II, RAM address −16336 is not memory, but a hard-wired connection to the speaker. Attempting to read from that location does not retrieve data (nothing's there), but generates a click. The short program in Example 1 produces a steady tone by employing a variant record to access the Apple II speaker. First the program treats the variant record named Memory as an integer. It is assigned the target address. Then the program treats the variant record Memory as a pointer to the first byte of the target word. The program repeatedly attempts to read from the target location, thereby sustaining a tone.

How Do Programs Use Unit DataOps?

Designating DataOps, or any unit for that matter, in a USES statement will incorporate it into a program. Refer to pp. 75ff in the *Apple Pascal*

Language Reference Manual that came with the Apple Pascal 1.1 system for details.

In particular, Unit DataOps contains two procedures, Input and Output. Both operate according to the attributes contained in the DATAREC passed in their parameter lists. Input accepts a string from the user and converts it to the required data type. Output accepts a data type and displays it as a formatted string.

The source listing of Unit DataOps is substantial. It is broken up into several text files for editing ease. Listing 1 shows the main file. Listing 2 illustrates a trivial example of using DataOps. Listings 3 through 11 provide the necessary Include files.

Listing 1 contains the Interface sec-

```
PROGRAM buzzoff;
CONST
   speaker           =        -16336;
TYPE
   byte              =        0..255;
   word              =        PACKED ARRAY [0..1] of byte;
VAR
   i,j               :        INTEGER;
   memory            :        PACKED RECORD CASE BOOLEAN OF
             TRUE    :     (address:INTEGER);
             FALSE   :     (target:^word);
             END;
   BEGIN
   WITH memory DO
             BEGIN
             address: = speaker;
             FOR i: = 511 DOWNTO 0 DO j: = target^[0];
END
```

Example 1. Program Buzzoff.

Listing 3. Procedure Output.

```
PROCEDURE output;
VAR
      ss          : STRING;        ( tempstring FOR various suborocedures )

      PROCEDURE outstring;
      VAR
            fieldsize         : INTEGER;
            blanks,
            leftfill,
            rightfill         : STRING;

            PROCEDURE fill (VAR filler:STRING);
            VAR
                  space   : INTEGER;
            BEGIN
            space:=fieldsize-length(d.result.s);
            IF(space)0)THEN filler:=copv(blanks,1,space);
            END;

      BEGIN
      IF (d.map.kind=picturedata) THEN fieldsize:=length (d.map.pic)
      ELSE fieldsize:= d.map.size;
      {$r-}
      fillchar(blanks[0],1,fieldsize);
      {$r+}
      IF(d.justify=securejustify)THEN fillchar(blanks[1],fieldsize,securefill)
      ELSE fillchar(blanks[1],fieldsize,' ');
      IF NOT(d.justify IN [leftjustify..rightjustifv]) THEN
            IF (d.origin=userdata) THEN
                  BEGIN
                  GOTOXY(d.xpos,d.ypos);
                  WRITE(blanks)
                  END;
      IF (d.justify IN [leftjustify..rightjustifv]) THEN
        IF (d.origin=userdata) THEN ( display it )
            BEGIN
            leftfill:='';
            rightfill:='';
```

Listing continued.

Listing continued.

```
CASE d.justify OF
     leftjustify:fill(rightfill);
     centerjustify:
             BEGIN
             fill(rightfill);
             leftfill:=
               copy(rightfill,1,length(rightfill)DIV 2);
             rightfill:=leftfill;
             IF((odd(length(d.result.s)))
                AND(length(d.result.s)<(fieldsize)) THEN
                  leftfill:=concat(' ',leftfill);
             END;
     rightjustify:fill(leftfill);
     END (CASE d.justify);
GOTOXY(d.xpos,d.ypos);
WRITE(leftfill);
WRITE (d.result.s);
WRITE (rightfill);

       END;
outputstring:=d.result.s;
END;

PROCEDURE outchar;
BEGIN
ss:=' ';
ss[1]:=d.result.c;
d.result.s:=ss;
d.result.kind:=stringdata;
outstring;
END;

PROCEDURE outreal;
VAR
      negative          : BOOLEAN;
      zerofill          : STRING[36];
      decimaladjust     : INTEGER;

      FUNCTION shift:INTEGER;
      CONST
             targetlow          = 1000.00;
             targethi           =10000.00;
      VAR
             shiftup,
             shiftdown          : INTEGER;
      BEGIN
      shiftup:=0; shiftdown:=0;
      IF(d.result.r<>0.00) THEN
             BEGIN
             IF(d.result.r<targetlow)THEN
                 WHILE (d.result.r<targetlow) DO (shift up)
                     BEGIN
                     d.result.r:=d.result.r*10;
                     shiftup:=shiftup+1;
                     END
             ELSE
                 WHILE (d.result.r)=targethi) DO (shift down)
                     BEGIN
                     d.result.r:=d.result.r/10;
                     shiftdown:=shiftdown+1;
                     END;
             END;
      shift:=shiftup-shiftdown;
      END;

      PROCEDURE pad;
      CONST
             shiftleft          = 3;
      VAR
             i,
             p1,p2              : INTEGER;
             fraction           : REAL;
             s1,s2              : STRING;
      BEGIN
      p1:=trunc(d.result.r);
      fraction:=d.result.r-p1;
      FOR i:=1 TO shiftleft DO fraction:=fraction * 10;
      p2:=round(fraction);IF(p2>999)THEN p2:=999;
      str(p1,s1); str(p2,s2);
         WHILE(length(s1)<4) DO s1:=concat('0',s1);
         WHILE(length(s2)<3) DO s2:=concat('0',s2);
         ss:=concat(zerofill,s1,s2,zerofill);
         END;

         PROCEDURE decimalpoint;
         BEGIN
         insert('.',ss,41-decimaladjust)
         END;

         PROCEDURE shave;
         BEGIN
         WHILE(ss[1]='0') DO delete(ss,1,1);
         WHILE(ss[length(ss)]='0') DO delete(ss,length(ss),1);
         IF(ss[1]='.') THEN ss:=concat('0',ss);
         IF(ss[length(ss)]='.') THEN ss:=concat(ss,'00')
         ELSE IF(ss[length(ss)-1]='.') THEN ss:=concat(ss,'0');
         END;

BEGIN
{$r-}
fillchar(zerofill[0],1,36);
{$r+}
fillchar(zerofill[1],36,'0');
IF(d.result.r)=0.0) THEN negative:=FALSE ELSE negative:=TRUE;
d.result.r:=abs(d.result.r);
```

Listing continued.

Listing continued.

```
        decimaladjust:=shift;
        pad;
        decimalpoint;
        shave;
        IF(negative)THEN ss:=concat('-',ss);
        d.result.kind:=stringdata;
        d.result.s:=ss;
        outstring;
        END;

        PROCEDURE outint;
        BEGIN
        str(d.result.i,ss);
        d.result.s:=ss;
        d.result.kind:=stringdata;
        outstring;
        END;

        PROCEDURE outmoney;
        VAR
                negative:BOOLEAN;
                i,j:INTEGER;
        BEGIN
        IF(d.result.m<0) THEN negative:=TRUE ELSE negative:=FALSE;
        str(d.result.m,ss);
        WHILE (length(ss)<3) DO ss:=concat('0',ss);
        insert('.',ss,length(ss)-1);
        i:=pos('.',ss);
        j:=i-3;
        WHILE(j>1) DO
                BEGIN
                i:=j;
                IF NOT((negative)AND(i=2)) THEN insert(',',ss,i);
                j:=i-3;
                END;
        d.result.s:=ss;
        d.result.kind:=stringdata;
        outstring;
        END;

        PROCEDURE outdate;
        VAR
                mo,da,yr:STRING;
        BEGIN
        str(d.result.d.month,mo);
        str(d.result.d.day,da);
        str(d.result.d.year,yr);
        ss:=concat(mo,'/',da,'/',yr);
        d.result.s:=ss;
        d.result.kind:=stringdata;
        outstring;
        END;

        PROCEDURE outssn;
        CONST
                ssnsize         = 11;
        VAR
                limit           : integer;
        BEGIN
        IF(d.map.kind=picturedata)THEN limit:=LENGTH(d.map.pic)
        ELSE limit:=d.map.size;
        IF NOT((length(d.result.ssn)+2)>limit) THEN
                BEGIN
                insert('-',d.result.ssn,4);
                insert('-',d.result.ssn,7);
                END;
        d.result.kind:=stringdata;
        outstring;
        END;

        PROCEDURE outphone;
        VAR
                number     :STRING[8];
        BEGIN
        ss:='';
        IF (length(d.result.phone.prefix)>0) THEN
                ss:=concat(d.result.phone.prefix,' ');
        IF (length(d.result.phone.area)>0) THEN
                ss:=concat(ss,'(',d.result.phone.area,') ');
        IF (length(d.result.phone.number)=7) THEN
                BEGIN
                number:=d.result.phone.number;
                insert('-',number,4);
                ss:=concat(ss,number)
                END
        ELSE ss:=concat(ss,d.result.phone.number);
        IF (length(d.result.phone.extension)>0) THEN
             ss:=concat(ss,' [',d.result.phone.extension,']');
        d.result.s:=ss;
        d.result.kind:=stringdata;
        outstring;
        END;

BEGIN (output)

CASE d.result.kind OF
        chardata:outchar;
        realdata:outreal;
        integerdata:outint;
        moneydata:outmoney;
        datedata:outdate;
        stringdata:outstring;
        ssndata:outssn;
        phonedata:outphone;
        END (CASE);
END;
```

```
PROCEDURE input;

CONST
        etx = 3;          esc = 27;          cr = 13;          bs = 8;
        bell = 7;
        maxresponse = 80;
VAR
        firstkeyin        : BOOLEAN;
        defaultselected   : BOOLEAN;
        letterset         : charset;
        controlset        : charset;
        processset        : charset;
        maskset           : charset;
        next              : CHAR;
        endinput          : BOOLEAN;
        charindex         : INTEGER;
        blankstring       : STRING[maxresponse];
        response          : PACKED ARRAY [1..maxresponse] OF CHAR;
        datastring        : STRING[maxresponse];
        charcounter       : INTEGER;
        i                 : INTEGER;
    PROCEDURE convert;
    VAR
        tempdate          : daterec;
        tempmoney         : money;
        tempssn           : STRING;
        tempphone         : phonerec;
```

Listing 4. Procedure Input.

Listing 5.

```
FUNCTION inreal (userstring:STRING) :REAL;
{
        interprets STRING as a REAL.  IF garbage characters
        THEN returns zero AND sets d.errcode TO 1
}

CONST
        dp              ='.';
        plus            ='+';
        minus           ='-';

VAR
        symbolset                                   : charset;
        negative,endstring,badchar,decimalfound     : BOOLEAN;
        thenumber                                   : STRING;
        len,index,dindex,vindex                     : INTEGER;
        factor,result                               : REAL;

        PROCEDURE zero;
        BEGIN
        inreal:=0.0; d.errcode:=1; EXIT(inreal)
        END;

        PROCEDURE validate;

                PROCEDURE ignoreblanks;
                LABEL 10,20;
                BEGIN
                10:
                len:=LENGTH(userstring);
                IF NOT(len>0) THEN zero
                ELSE IF (userstring[1]=' ')THEN
                        BEGIN
                        DELETE(userstring,1,1);
                        GOTO 10;
                        END;
                20:
                IF NOT(len>0) THEN zero
                ELSE IF (userstring[len]=' ')THEN
                        BEGIN
                        DELETE(userstring,len,1);
                        len:=LENGTH(userstring);
                        GOTO 20;
                        END;
                END;

        BEGIN
        ignoreblanks;
        IF NOT (userstring[1] IN digitset) THEN
                BEGIN
                IF NOT (userstring[1] IN symbolset) THEN zero;
                IF (len=1) THEN zero;
                IF ((userstring[1]=plus)OR(userstring[1]=minus))
               THEN BEGIN
                    IF (userstring[2]=dp) THEN
                      IF (len<3) THEN zero
              ELSE {it'S VALID}
            ELSE IF NOT (userstring[2] IN digitset) THEN zero;
            IF (userstring[1]=minus) THEN negative:=TRUE
            ELSE negative:=FALSE
            END
         ELSE IF (userstring[1]=dp) THEN
                IF NOT (userstring[2] IN digitset) THEN zero
                ELSE {continue}
         END;
 thenumber:=userstring;
 IF ((thenumber[1]=plus)
    OR(thenumber[1]=minus)) THEN delete (thenumber,1,1)
 END {validate};
 PROCEDURE checkchars;
```

Listing continued.

Listing continued.

```
        BEGIN
        index:=0; badchar:=FALSE; decimalfound:=FALSE;
        endstring:=FALSE;
        REPEAT
                index:=index+1;
                IF (length(thenumber)=index) THEN
                    BEGIN
                    endstring:=TRUE;
                    vindex:=index
                    END;
                IF (thenumber[index]=dp) THEN CASE decimalfound OF
                    TRUE: BEGIN vindex:=index-1; badchar:=TRUE END;
                    FALSE:BEGIN decimalfound:=TRUE; dindex:=index END;
                    END {CASE}
                ELSE IF NOT (thenumber[index] IN digitset) THEN
                    BEGIN
                    vindex:=index-1;
                    badchar:=TRUE
                    END;
        UNTIL ((badchar) OR (endstring));
        IF (vindex<length(thenumber)) THEN
                delete (thenumber,vindex+1,length(thenumber)-vindex)
        END   {checkchars};


        PROCEDURE evaluate;
        CONST
                hilimit          = 1.00000E37;
                lolimit          = 1.00000E-24;
        VAR
                i:INTEGER;
        BEGIN
        result:=0.0;
        CASE decimalfound OF
                TRUE:
                BEGIN
                IF (dindex>1) THEN
                        BEGIN
                        factor:=1;
                        FOR i:=(dindex-1) DOWNTO 1 DO
                                BEGIN
                                result:=result+
                                  (ord(thenumber[i])-ord('0'))*factor;
                                factor:=factor*10;
                                IF (NOT(factor<hilimit)) THEN zero;
                                END
                        END;
                IF (dindex<length(thenumber)) THEN
                        BEGIN
                        factor:=0.1;
                        FOR i:=(dindex+1) TO length(thenumber) DO
                                BEGIN
                                result:=result+
                                    (ord(thenumber[i])-ord('0'))*factor;
                                factor:=factor/10;
                                IF (factor<lolimit) THEN zero;
                                END
                        END
                END {TRUE};
                FALSE:
                BEGIN
                factor:=1.0;
                FOR i:=vindex DOWNTO 1 DO
                        BEGIN
                        result:=
                          result+((ord(thenumber[i])-ord('0'))*factor);
                        factor:=factor*10;
                        IF (NOT(factor<hilimit)) THEN zero;
                        END
                END {FALSE}
         END {CASE decimalfound};
         IF (negative) THEN result:=result*(-1);
         END {evaluate};

BEGIN   {FUNCTION inreal}
IF (length(userstring)<1) THEN zero;
negative:=FALSE;
symbolset:=[plus,minus,dp]+digitset;
validate; checkchars; evaluate; inreal:=result; d.errcode:=0;
END     {FUNCTION inreal};
```

```
FUNCTION inint (s:STRING):INTEGER;
VAR
        r                   : REAL;


          PROCEDURE badint;
          BEGIN inint:=0; EXIT(inint) END;



BEGIN
r:=inreal(s);
IF (d.errcode<>0) THEN BEGIN d.errcode:=1; badint END;
IF (abs(r)>maxint)  THEN BEGIN d.errcode:=2; badint END;
inint:= round(r);
d.errcode:=0;
END;
```

Listing 6.

tion, the beginning of the Implementation section, the Include directives instructing the compiler to fetch appropriate text files in correct order, and the unit initialization code.

The initialization code is executed when the unit is loaded, before the application program begins. It sets up global variables.

"It is unwise to express money amounts as reals because of limited precision."

The Interface section contains the declarations needed to exchange, enter, validate and format multiple data types. Programs interact with the unit through these declaratons.

The first special type is Money. It is unwise to express money amounts as reals because of limited precision. Calculations performed with long integers are slow, but pennies do not get misplaced. Money is a long integer type.

Next is DATEREC. Dates are carried in various parts of the UCSD p-System. This type mirrors that structure.

The PHONEREC is used for telephone numbers. Telephone numbers are stored as a series of short strings, one each for PBX prefix, area code, local number and extension.

The JUSTIFYTYPE enumerates the possible styles of displaying data within a defined field; DONTJUSTIFY is used for temporary data, such as answers to prompts; LEFTJUSTIFY and CENTERJUSTIFY are used for names, labels, and titles; RIGHTJUSTIFY is used for numbers; SECUREJUSTIFY is used for passwords. They are not limited to these uses and may be interchanged.

The RESULTTYPE enumerates the possible kinds of data that may reside in a RESULTREC, such as characters, Social Security numbers, integers, etc.

The MAPTYPE enumerates the possible entry modes that may reside in

Listing 7.

```
PROCEDURE inmoney (VAR mm:money; ascii:STRING);
(
        inmoney interprets strings as INTEGER[16]s money amounts
        the maximum is $ 99,999,999,999,999.99. the result is
        deposited into the VAR parameter mm.

        error codes are returned IN the global variable d.errcode
        as follows..

         0   conversion successful
         1   illegal character IN STRING
         2   null STRING passed
         3   misplaced dollar sign
         4   misplaced comma
         5   STRING overrun
         6   misplaced decimal point
         7   too many cents digits
         8   misplaced minus sign
)

CONST
        maxlen=16;
        zeros14='00000000000000';
        zeros16='0000000000000000';

VAR
        negative,nodigits:BOOLEAN;
        symbols,possibles:charset;
        dollars,cents:STRING;
        minus,blank,comma,dsign,dp:STRING[1];
        comma_pos:PACKED ARRAY[1..4] OF INTEGER;
        i,j,err,minuscount,blankcount,
        commacount,dpcount,dsigncount:INTEGER;

        PROCEDURE error;
        (
                SET d.errcode TO error code. SET mm TO zero
                AND EXIT the PROCEDURE.
        )
        BEGIN d.errcode:=err; mm:=0; EXIT(inmoney) END;

        PROCEDURE initialize;
        (
                SET up variables.
        )
        VAR i:INTEGER;
        BEGIN (initialize)
        IF (length(ascii)<1) THEN BEGIN err:=2; error END;
        err:=0;
        negative:=FALSE;
        FOR i:= 1 TO 4 DO comma_pos[i]:=0;
        mm:=0;
        minus:='-';         blank:=' ';       comma:=',';
        dp:='.';            dsign:='$';
        minuscount:=0;      blankcount:=0;    commacount:=0;
        dpcount:=0;         dsigncount:=0;
        symbols:=['-',',','.','$',' '];

possibles:=digitset+symbols;
END   (initialize);


PROCEDURE minus sign check;
(
        check IF valid negative number.
)
BEGIN
IF (pos(minus,ascii)=0) THEN (no action needed)
ELSE IF (pos(minus,ascii)=1) THEN
        BEGIN negative:=TRUE; delete (ascii,1,1) END
ELSE
        BEGIN
        FOR i:= 1 TO (pos(minus,ascii)-1) DO
                IF ((ascii[i]<>' ') OR (ascii[i]<>'$')) THEN
                        BEGIN err:=8; error END;
        negative:= TRUE; delete (ascii,1,pos(minus,ascii));
        END;
END;

PROCEDURE checkdsign;
(
        check FOR properly placed dollar sign.
)
BEGIN (checkdsign)
IF (pos(dsign,ascii)=0) THEN (no action needed)
ELSE IF (pos(dsign,ascii)=1) THEN delete(ascii,1,1)
ELSE
        BEGIN
        FOR i:=1 TO (pos(dsign,ascii)-1) DO
                IF (ascii[i]<>' ') THEN BEGIN err:=3; error END;
        blankcount:=blankcount-(pos(dsign,ascii)-1);
        delete (ascii,1,pos(dsign,ascii));
        END;
END   (checkdsign);

PROCEDURE checkblanks;
(
        remove leading AND trailing blanks.
)
BEGIN (checkblanks)
i:=1;
WHILE (ascii[i]=' ') DO i:=i+1;
i:=i-1;
IF (i>0) THEN delete(ascii,1,i);
i:=length(ascii);
WHILE (ascii[i]=' ') DO i:=i-1;
i:=i+1;
IF (i<=length(ascii)) THEN delete (ascii,i,length(ascii)-i+1);
END   (checkblanks);

PROCEDURE checkcomma;
(
        check FOR proper placement OF commas
)
VAR i,j,k:INTEGER;
BEGIN (checkcomma)
j:= 1; FOR i:= 1 TO (length(ascii)) DO IF (ascii[i]=',') THEN
        BEGIN

        comma_pos[j]:= i; j:= j+1;
        END;
FOR j:= 1 TO 4 DO IF (comma_pos[j]>0) THEN
        IF (comma_pos[j]>(length(ascii)-3)) THEN
                BEGIN err:=4; error END;
FOR j:= 1 TO 4 DO IF (comma_pos[j]>0) THEN
        FOR k:= (comma_pos[j]+1) TO (comma_pos[j]+3) DO
                IF NOT (ascii[k] IN digitset) THEN
                        BEGIN err:=4; error END;
IF (pos(dp,ascii)>0) THEN k:=pos(dp,ascii)
ELSE k:= length(ascii)+1;
FOR i:= 1 TO 4 DO IF (comma_pos[i]>0) THEN
        IF (((k-comma_pos[i])MOD 4)<>0) THEN
                BEGIN err:=4; error END;
WHILE (pos(comma,ascii)<>0) DO
        delete (ascii,pos(comma,ascii),1);
END   (checkcomma);

PROCEDURE checkcents;
(
        check FOR properly entered cents digits.  supply missing
        zeros IF needed.
)
BEGIN (checkcents)
IF ((dpcount=0)OR(length(ascii)=pos(dp,ascii))) THEN cents:='00'
ELSE
 cents:=copy(ascii,pos(dp,ascii)+1,length(ascii)-pos(dp,ascii));
IF (length(cents)=1) THEN cents:=concat(cents,'0');
IF (length(cents)>2) THEN BEGIN err:=7; error END;
FOR i:=1 TO 2 DO IF NOT (cents[i] IN digitset) THEN
        BEGIN err:=1; error END;
END   (checkcents);

PROCEDURE checkdollars;
(
        check FOR properly entered dollars amount, OR SET
        TO zero IF necessary.
)
BEGIN (checkdollars)
IF (dpcount=0) THEN dollars:=ascii
ELSE IF (pos(dp,ascii)=1) THEN dollars:=zeros14
ELSE dollars:=copy(ascii,1,pos(dp,ascii)-1);
FOR i:=1 TO length(dollars) DO
    IF NOT(dollars[i] IN digitset) THEN
        BEGIN err:=1; error END;
END   (checkdollars);


PROCEDURE makelong;
(
convert the ascii STRING TO long16
)
VAR
        i:INTEGER;
        factor:INTEGER[17];
BEGIN
factor:=1;
mm:=0;
FOR i:=length(ascii) DOWNTO 1 DO
        BEGIN
                mm:=mm +factor * (ord(ascii[i]) - ord('0'));
                factor:=factor * 10;
                END;
        END;

BEGIN (inmoney)
initialize;
nodigits:=TRUE; FOR i:=1 TO length(ascii) DO
    BEGIN
    IF NOT (ascii[i] IN possibles) THEN BEGIN err:=1; error END;
    IF (ascii[i] IN digitset) THEN nodigits:=FALSE
    ELSE IF (ascii[i]=' ') THEN blankcount:=blankcount+1
    ELSE IF (ascii[i]=',') THEN commacount:=commacount+1
    ELSE IF (ascii[i]='.') THEN dpcount:=dpcount+1
    ELSE IF (ascii[i]='$') THEN dsigncount:=dsigncount+1
    ELSE IF (ascii[i]='-') THEN minuscount:=minuscount+1;
    END;
IF ((nodigits=TRUE)OR
    (minuscount>1)OR(commacount>4)OR(dpcount>1)OR(dsigncount>1))
    THEN BEGIN err:=1;  error END;
IF (minuscount>0) THEN minussigncheck;
IF (dsigncount>0) THEN checkdsign;
IF (blankcount>0) THEN checkblanks;
IF (commacount>0) THEN checkcomma;
checkcents;
checkdollars;
ascii:=concat(dollars,cents);
IF (length(ascii)>maxlen) THEN BEGIN err:=5; error END;
IF (length(ascii)<maxlen) THEN
```

Listing continued.

Listing continued.

```
              BEGIN (pad left part OF STRING WITH zeros)
              ascii:=concat(zeros16,ascii);
              delete(ascii,1,length(ascii)-maxlen)
              END;
      makelong;
      IF (negative) THEN mm:=mm * (-1);
      d.errcode:=err;
      END   (inmoney);
```

Listing 8.

```
PROCEDURE indate (date:STRING; VAR rec:daterec);

CONST   delimeter = '/';

VAR     mo,da,yr,i          : INTEGER;

        PROCEDURE bad date;
        BEGIN
        rec.month:=0;        rec.day:=0;          rec.year:=0;
        EXIT(indate);
        END;

        PROCEDURE assign (VAR part: INTEGER; index: INTEGER);
        VAR temp: STRING;
        BEGIN
        temp:= copy (date,1,index-1);
        part:= inint(temp);
        delete (date,1,index);
        END;

        PROCEDURE verify;
        TYPE
        month_range=1..12; day_range=1..31; year_range=0..100;
        VAR
        leapyear: BOOLEAN;
        short_months,
        feb_month,
        valid_months    :SET OF month_range;
        valid_days      :SET OF day_range;
        valid_years     :SET OF year_range;

                PROCEDURE initialize;
                BEGIN
                valid_months:=[1..12];
                valid_days:=[1..31];
                valid_years:=[0..100];
                short_months:=[4,6,9,11];
                feb_month:=[2];
                END;

        BEGIN
        initialize;
        IF (mo IN valid_months) THEN rec.month:=mo
        ELSE
                BEGIN
                d.errcode:=1; bad_date;
                END;
        IF (da IN valid_days) THEN rec.day:=da
        ELSE
                BEGIN
                d.errcode:=2; bad date;
                END;
        IF (yr IN valid_years) THEN rec.year:=yr
        ELSE
                BEGIN
                d.errcode:=3; bad date;
                END;
        leapyear:= (rec.year MOD 4 = 0);
        IF ((NOT(leapyear))
           AND(rec.month IN feb_month)
           AND(rec.day>28))THEN
                BEGIN
                d.errcode:=2; bad_date
                END;
        IF ((leapyear)
           AND(rec.month IN feb_month)
           AND(rec.day>29))THEN
                BEGIN
                d.errcode:=2; bad_date
                END;
        IF ((rec.month IN shortmonth)
           AND(rec.day>30))THEN
                BEGIN
                d.errcode:=2; bad date
                END;
        END;

BEGIN (PROCEDURE indate)
i:= pos(delimeter,date);
IF (i=0) THEN
        BEGIN
        d.errcode:=6; bad date
        END;
IF (i<2) THEN
        BEGIN
        d.errcode:=1; bad date
        END
ELSE assign (mo,i);
```

Listing continued.

Listing continued.

```
        i:= pos(delimeter,date);
        IF (i=0) THEN
                BEGIN
                d.errcode:=6; bad_date
                END;
        IF (i<2) THEN
                BEGIN
                d.errcode:=2; bad date
                END
        ELSE assign (da,i);
        IF (length(date)<1) THEN
                BEGIN
                d.errcode:=6; bad date
                END
        ELSE yr:=inint(date);
        verify;
        d.errcode:=0;
        END   (PROCEDURE indate);
```

```
PROCEDURE inssn (s:STRING; VAR ssn:STRING);
(
  checks FOR a STRING consisting OF exactly nine digits.
  d.errcode value are
        0       normal completion
        127     length OF numeric STRING <> 9
        128     non digit chars encountered IN STRING
)
CONST exactlength=9;
VAR i:INTEGER;

    PROCEDURE abort(err:INTEGER);
    BEGIN
    ssn:='';
    d.errcode:=err;
    EXIT(inssn);
    END;

BEGIN
IF(length(s)<>exactlength) THEN abort(127);
FOR i:=1 TO exactlength DO
        IF NOT(s[i] IN digitset) THEN abort(128);
ssn:=s;
d.errcode:=0;
END;
```

Listing 9.

```
PROCEDURE inphone (s:STRING; VAR f:phonerec);
( s should be a STRING IN the form OF a (bbb) ccc - dddd [eeee] where
a is an optional single digit prefix
bbb is an optional area code enclosed IN parenthesis
ccc AND dddd is the seven digit phone number
eeee is an optional extension up TO 4 digits enclose IN brackets

the following are examples OF valid STRINGs

9 (212) 676 - 2888 [127]
6762888
(212) 676-2888

note that a pbx prefix cannot occur without an area code.
values returned IN d.errcode are

0        completion valid
127      garbage STRING passed
128      aborted on prefix WITH bad data
129      aborted on area code WITH bad data
130      aborted on number WITH bad data
131      aborted on extension WITH bad data
255      null string passed
)
CONST
         leftparen        = '(';
         leftbrac         = '[';
         extrachar        = '?';
VAR
         inphoneset        : charset;

     PROCEDURE abort (err:INTEGER);
     BEGIN
     d.errcode:=err;
     fillchar(f,sizeof(phonerec),0);
     EXIT(inphone)
     END;
PROCEDURE shrinkit;
VAR
         i:INTEGER;
BEGIN
FOR i:=1 TO length(s) DO
         IF NOT(s[i] IN inphoneset) THEN s[i]:=extrachar;
WHILE (pos(extrachar,s)<>0) DO delete(s,pos(extrachar,s),1);
IF (length(s)<1) THEN abort(127);
END;

PROCEDURE prefix;
BEGIN
f.prefix:='';
IF pos(leftparen,s)=2 THEN
     BEGIN
     f.prefix:=copy(s,1,1);
     IF NOT(f.prefix[1] IN digitset) THEN abort(128);
     delete(s,1,1);
     END
ELSE IF pos(leftparen,s)>2 THEN abort(128);
END;

PROCEDURE area;
BEGIN
f.area:='';
IF(pos(leftparen,s)>1) THEN abort(129)
ELSE IF (pos(leftparen,s)=1) THEN
     BEGIN
     delete(s,1,1);
     f.area:=copy(s,1,3);
     IF(length(f.area)<>3) THEN abort(129);
     FOR i:=1 TO 3 DO IF NOT(f.area[i] IN digitset) THEN abort(129);
     delete(s,1,3);
     END;
END;

PROCEDURE number;
BEGIN
f.number:='';
IF (length(s)<7) THEN abort(130);
f.number:=copy(s,1,7);
delete(s,1,7);
FOR i:=1 TO 7 DO IF NOT(f.number[i] IN digitset) THEN abort(130);
END;

PROCEDURE extension;
BEGIN
f.extension:='';
IF(length(s)>0) THEN
     IF pos(leftbrac,s)=1 THEN
          BEGIN
          delete(s,1,1);
          f.extension:=s;
          IF length(f.extension)>4 THEN abort(131);
          IF length(f.extension)>0 THEN
               FOR i:=1 TO length(f.extension) DO
                    IF NOT(f.extension[i] IN digitset) THEN abort(131);
          END
     ELSE abort(131);
END;
BEGIN
fillchar(f,sizeof(phonerec),0);
IF (length(s)<1) THEN abort(255);
inphoneset:=digitset+[leftparen,leftbrac];
shrinkit;
prefix;
area;
number;
extension;
d.errcode:=0;
END;
```

Listing 10.

MAPREC. The two modes of entering data are free form and picture form. Free form is ASCII characters within a maximum area (sizedata). Controlled form is restricted ASCII characters according to a specified mask (PICTUREDATA).

Following is MAPREC and RESULTREC, both variant records whose actual data is defined by the tag field, KIND.

The ORIGINTYPE enumerates the possible modes of conversion: convert from one type to another only (programdata), or convert and display on the console (userdata).

Finally, there is DATAREC. This large record is used as the staging area for all data communication between unit DataOps and application programs. All of the types described above are combined to form the fields of DATAREC.

"The two modes of entering data are free form and picture form."

DATAREC.ERRCODE contains an integer value indicating status after a data entry operation. Zero indicates proper completion, -1 indicates the escape key was pressed, and a positive number indicates an exception, which is defined differently depending on the actual data type desired.

DATAREC.XPOS and DATAREC.YPOS contain screen coordinates for the entering and the displaying of the item. These are assigned by the application.

DATAREC.JUSTIFY governs how data will be displayed within its field area on the screen. This is assigned by the program.

DATAREC.ORIGIN governs if the data will be both converted and displayed, or simply converted. This will usually be set to USERDATA by the program.

DATREC.DEFAULT contains a default value to be displayed on the screen

Listing 11.

```
BEGIN (convert)
CASE d.result.kind OF
     integerdata:d.result.i:=inint(datastring);
     realdata:d.result.r:=inreal(datastring);
     chardata:d.result.c:=datastring[1];
     moneydata:BEGIN
               inmoney(tempmoney,datastring);
               d.result.m:=tempmoney;
               END;
     datedata:BEGIN
               indate(datastring,tempdate);
               d.result.d:=tempdate;
               END;
     stringdata:d.result.s:=datastring;
     ssndata:BEGIN
               inssn(datastring,tempssn);
               d.result.ssn:=tempssn;
               END;
     phonedata:BEGIN
               inphone(datastring,tempphone);
               d.result.phone:=tempphone;
               END;
     END;
END;

PROCEDURE echo (c:CHAR);
BEGIN
GOTOXY(d.xpos+charindex-1,d.ypos);
IF(d.justify=securejustify) THEN WRITE(securefill) ELSE WRITE(c)
END;

PROCEDURE beep;
BEGIN
WRITE(chr(bell))
END;

PROCEDURE backup;
BEGIN
CASE d.map.kind OF
        sizedata:
                BEGIN
                IF NOT(charindex>d.map.size) THEN echo(fieldmarker);
                IF (charindex>1) THEN
                        BEGIN
                        charindex:=charindex-1;
                        echo(fieldmarker);
                        END
                ELSE beep
                END;
        picturedata:
                BEGIN
                IF NOT(charindex>length(d.map.pic)) THEN
                        IF NOT(d.map.pic[charindex] IN maskset) THEN
                                echo(d.map.pic[charindex])
                        ELSE echo(fieldmarker);
                IF (charindex>1) THEN
                        BEGIN
                        charindex:=charindex-1;
                        IF NOT(d.map.pic[charindex]IN maskset) THEN
                                BEGIN
                                echo(d.map.pic[charindex]);
                                backup;
                                END
                        ELSE echo(fieldmarker);
                        END
                ELSE beep
                END;
        END(CASE);
END;

PROCEDURE keyin (VAR c:CHAR);
            ( SET d.errcode TO -1 IF specialset is pressed
                    AND EXIT ELSE return a CHAR )
TYPE
        ccarray = PACKED ARRAY [0..1] OF CHAR;
VAR
        cc      : ccarray;
BEGIN
        REPEAT
                unitread(2,cc[0],1);
                IF (ord(cc[0])>127) THEN
                    cc[0]:=chr(ord(cc[0])-128);
        UNTIL (cc[0] IN processset);
        IF (cc[0] IN specialset) THEN
                BEGIN
                d.justify:=dontjustify;
                output(d);
                d.errcode:=-1;
                EXIT(input);
                END
        ELSE c:=cc[0];
END;

PROCEDURE endit;
BEGIN
charindex:=charindex-1; endinput:=TRUE
END;

PROCEDURE addit;
BEGIN
echo(next);
response[charindex]:=next;
charindex:=charindex+1;
END;
```

Listing continued.

Listing continued.

```
        PROCEDURE stuffit;
        BEGIN
        ($r-)
        datastring[0]:=chr(charindex);
        moveleft(response[1],datastring[1],charindex);
        ($r+)
        END;

        PROCEDURE inputpicture;
        (
            d.map.picture STRING is interpreted as follows:
        D OR d  means accept chars '0' thru '9' only
        N OR n  means accept chars '0' thru '9' and space
        L OR l  means accept alphas AND space
        A OR a  means accepts alphas only
        S OR s  means accept digit chars, alphas, OR space
        X OR x  means accept any printable CHAR

        any character appearing IN d.map.picture but NOT IN the
        above list will be inserted verbatim into the input STRING.

    )
    BEGIN
    WHILE NOT(endinput) DO
            BEGIN
            IF(charindex)length(d.map.pic)) THEN
                    REPEAT
                    keyin(next);
                    IF NOT(next IN controlset) THEN beep;
                    UNTIL (next IN controlset)
            ELSE IF NOT(d.map.pic[charindex] IN maskset) THEN
                    next:=d.map.pic[charindex]
            ELSE
                BEGIN
                    GOTOXY(d.xpos+charindex-1,d.ypos);
                    CASE d.map.pic[charindex] OF
                     'a','A':
                        REPEAT
                            keyin(next)
                        UNTIL (next IN letterset)OR(next IN controlset);
                     'd','D':
                        REPEAT
                            keyin(next)
                        UNTIL (next IN digitset)OR(next IN controlset);
                     'l','L':
                        REPEAT
                            keyin(next)
                        UNTIL (next IN letterset)OR(next IN controlset)
                              OR(next=' ');
                     'n','N':
                        REPEAT
                            keyin(next)
                        UNTIL (next IN digitset)OR(next IN controlset)
                              OR (next=' ');
                     'x','X':
                        REPEAT
                            keyin(next)
                        UNTIL (next IN processset);
                     's','S':
                        REPEAT
```

Listing continued.

prior to data entry. The default string may be null. In that case no default is accepted.

DATREC.MAP contains either the maximun number of characters allowed during data entry or a precise mask that will be enforced during data entry. If the program assigns SIZEDATA to DATAREC.MAP.KIND, then it should also assign DATAREC.MAP.SIZE an integer value. If the program assigns PICTUREDATA to DATAREC.MAP.KIND, then it should also assign DATAREC.MAP.PIC a string image.

DATAREC.RESULT.? contains the converted user's input string. The program assigns a value to the tag variant field DATAREC.RESULT.KIND, which governs the conversion procedure subsequently called by the unit. For instance, if DATAREC.RESULT.KIND:= moneydata, then the unit will convert the user's input into a money value, returned as DATAREC.RESULT.M.

Unit DataOps provides three global variables: SECUREFILL is the character printed on the screen when data is entered in SECUREJUSTIFY mode. It is preset to an *(*) but may be changed by the program. FIELDMARKER is the character used to outline the area on the console where data will be entered. It is preset to an underscore (_) but may be changed by the program. OUTPUTSTRING contains the string interpretation of the last item of data sent to PROCEDURE INPUT or PROCEDURE OUTPUT.

"I would be delighted to hear about improvements."

Example

Listing 2 shows a meager program that uses DataOps. Two items of data are obtained, a third item is calculated, and all are formatted on the console. Using DataOps in a trivial program is overkill; it is intended for screen-intensive and data-intensive programs such as accounting packages and database managers.

Before compiling or executing the example program or any program that uses DataOps, the unit's code file must be installed into the library file named SYSTEM.LIBRARY located on the boot disk. Proceed as follows:

Step 0—Start with an initialized disk. Name it DA:. Type in all the text files (Listing 1, and Listings 3–11) and store them on DA. Enter the filer and set the prefix to DA. Enter the compiler and compile LISTING1.TEXT to DATAOPS.CODE. Transfer DATAOPS.CODE to a work disk. Save DA in a safe place.

Step 1—Execute the program called LIBRARY CODE (usually found on the Apple Pascal 1.1 System disk named APPLE3:) Follow to the instructions for this program on pp. 188ff in the *Apple Pascal Operating System Reference Manual* that came with the system.

Step 2—Replace *SYSTEM LIBRARY with the file NEW LIBRARY. Of course, always work with *copies* of the system disks. In case of castastrophe, recover from the master disks.

Conclusion

I hope unit DataOps saves you time and effort in programming projects and inspires you to code and share your favorite units. I would be delighted to hear about improvements in the DataOps implementation, especially about any bugs that have escaped my notice. Strictly as a convenience for those readers who would like to try DataOps but abhor typing in long files, I will send a disk copy of the source for $10. Enjoy. ■

Listing continued.

```
                        keyin(next)
                    UNTIL (next IN digitset)OR(next IN controlset)
                          OR(next IN letterset)OR(next=' ');
                END;
            END;
        IF((next=chr(cr))AND(firstkeyin=TRUE)AND(length(d.default))0))THEN
                BEGIN
                defaultselected:=TRUE;
                endit
                END
                ELSE IF (next=chr(bs)) THEN backup
                ELSE IF((next=chr(cr))AND (charindex)length(d.map.pic))) THEN
                        endit
                ELSE IF ((next=chr(cr)) OR NOT(next IN processset)) THEN beep
                ELSE addit;
                firstkeyin:=FALSE;
                END;
        IF(defaultselected) THEN datastring:=d.default
        ELSE stuffit;
        END;

        PROCEDURE inputfreeform;
        BEGIN
        keyin(next);
        IF (length(d.default))0)AND(next=(chr(cr))) THEN
                BEGIN
                defaultselected:=TRUE;
                FOR charcounter:=1 TO length(d.default) DO
                        BEGIN
                        next:=d.default[charcounter];
                        addit;
                        END;
                endit;
                END
        ELSE defaultselected:=FALSE;
        WHILE NOT(endinput) DO
                BEGIN
                IF (next=chr(bs)) THEN backup
                ELSE IF (next=chr(cr)) THEN endit
                ELSE IF NOT(next IN processset) THEN beep
                ELSE IF (charindex)d.map.size) THEN beep
                ELSE addit;
                IF NOT(endinput) THEN
                        BEGIN
                        GOTOXY(d.xpos+charindex-1,d.ypos);
                        keyin(next)
                        END
                END;
        stuffit;
        END;

        PROCEDURE initialize;
        BEGIN
        charindex:=1;
        endinput:=FALSE;
        defaultselected:=FALSE;
        firstkeyin:=TRUE;
        letterset:=['A'..'Z','a'..'z'];
        printset:=[' '..'}'];
        digitset:=['0'..'9'];
        specialset:=[chr(esc)];
        maskset:=['A','a','D','d','L','l','N','n','S','s','X','x'];
        controlset:=[chr(bs),chr(cr)],
        processset:=printset+specialset+controlset;
        {$r-}
        blankstring[0]:=chr(maxresponse),
        {$r+}
        fillchar(blankstring[1],maxresponse,' ');
        fillchar(response[1],maxresponse,'?');
        d.errcode:=0,
    GOTOXY(d.xpos,d.ypos);
    CASE d.map.kind OF
          sizedata:
                 BEGIN
                 IF length(d.default))0 THEN
                         BEGIN
                         IF (length(d.default))d.map.size) THEN
                                 d.default:=copy(d.default,1,d.map.size);
                         FOR i:=1 TO d.map.size DO WRITE(' ');
                         GOTOXY(d.xpos,d.ypos);
                         WRITE(d.default);
                         END
                 ELSE FOR i:=1 TO d.map.size DO WRITE(fieldmarker);
                 END;
          picturedata:
                 BEGIN
                 IF length(d.default))0 THEN
                         BEGIN
                         IF (length(d.default))length(d.map.pic)) THEN
                             d.default:=copy(d.default,1,length(d.map.pic));
                         FOR i:=1 TO length(d.map.pic) DO WRITE(' ');
                         GOTOXY(d.xpos,d.ypos);
                         WRITE(d.default);
                         END
                 ELSE FOR i:=1 TO length(d.map.pic) DO
                     IF NOT(d.map.pic[i] IN maskset) THEN WRITE(d.map.pic[i])
                     ELSE WRITE(fieldmarker);
                 END;
          END (CASE);
    GOTOXY(d.xpos,d.ypos);
    END;

BEGIN (input)
initialize;
IF (d.map.kind=picturedata) THEN inputpicture ELSE inputfreeform;
convert;
output(d);
END;
```

Confessions of a Franklin Owner

How one man turned his back on the Apple for... the Other Machine.

by Gerry Souter

Life is not easy for a Franklin owner, standing alone in the orchard. I thought of submitting this under a *nom de word processor*, but at the last moment I decided to let the chips fall where they may. You have to take a stand somewhere and what better place than in a publication devoted to the Apple.

Let me risk instant rejection by stating that this is not an apologia. The Franklin computer is an excellent machine and a real workhorse. Many, many hours have been spent hunched over its hot keyboard since it was uncrated in my home some months ago. This is no sleezoid compilation of bits and pieces imported from the Third World and cobbled into a tatty imitation. The Franklin is a stand-alone computer designed to take advantage of the widest selection of software and some of the better peripherals available. At the time I made my purchase (the waning months of 1982), the features it offered were very competitive.

I had shopped for two months before narrowing my purchase options down to the most computer I could get for the money I had to spend. Since I am a writer, word processing was paramount, and since my wife runs a graphic design business from our home, business management and eventual graphics packages were needed.

A friend of mine owned an Apple and touted it to the skies. He even let me fondle the keys and make the G beep. He ran mailing lists, word processing, spreadsheets, games, the whole gamut of software, churning it out through a huge Perkin Elmer letter quality printer he got in a horse trade. The amount of Apple software seemed limitless and the II Plus chugged away day and night without complaint. There were a few quirks to be noted, however.

> **"I noticed an unfamiliar shape crouching on the counter top. The lump was brown and beige and sort of squarish."**

He described how he had wandered around inside his Apple with a soldering iron and some bell wire to effect the legendary "shift key option." I blanched at the idea of drooling hot solder around those delicate-looking innards and poking wires into tiny holes. Sobered by the possibility of turning my investment into a ten pound paperweight, I decided I could get along using the escape key if I had to.

Another problem was entering columns of numbers into his spreadsheets. The typewriter-style keyboard was inconvenient at best. Still, I persisted. I had spent years pecking away at the standard QWERTY keyboard. In the end, the vast amount of available software and peripherals swayed me. I sought out the best deal on an Apple.

Having found an amiable Apple dealer who would take my money a little at a time, I went to his shop with my car trunk empty. As he showed me his best package deal, I noticed an unfamiliar shape crouching on the counter top. The lump was brown and beige and sort of squarish. On top of its huge, aircraft-carrier-size lid, an NEC monitor continuously scrolled the words, "I am a Franklin Ace computer... Won't you please buy me for 'n' dollars...?"

I was moved. "What's that?" I asked.

"We used to be a Franklin dealer," the salesman answered. "That's the last model we have in stock."

A memory stirred, recalled from the tons of magazines and literature I had pored over. The light seemed to leave the salesman's eyes as I moved closer to the Franklin. In my mind's microfiche, the specs came together: 64K RAM, full keypad with a set of math operators, built-in fan, upper/lowercase letters—and Apple hardware and

Address correspondence to Gerry Souter, 905 E. Frederick St., Arlington Heights, IL 60004.

software compatible! It was huge, half again as large as the Apple. It was brown with an orange stripe. It was last year's model—the Ace 100. It was the last puppy in the window, the runt of the litter.

"N dollars?" I asked. The salesman nodded numbly. "Then I could afford a second disk drive—that Apple Disk II—and an Apple dot matrix printer?"

"Yes. . . yes," he answered, the light returning beneath his shaggy brows.

"Done," I said, stroking the NEC monitor which accompanied the deal and was still scrolling its plaintive message. I switched off the screen.

That was months ago. Today, words fill my disk files, numbers virtually thunder across my VisiCalc. I have become a connoisseur of word processors, having tried AppleWriter II, Screenwriter II and finding simple pleasure in the one I am using to write this manuscript: Acewriter, of course. It needs no shift key modification to give full keyboard function and I have an 80-column screen with full-size letters—thanks to horizontal scrolling. All keys repeat their duties effortlessly, and the tab key is a joy, sitting as it does above both a break and a pause key. I use the latter convenience to page through my homework from the Applesoft basic programming course I'm taking at the local community college.

I heard that sneer! I've encountered it before—mostly in Apple showrooms when I buy disks and other supplies. I've seen Apple salesmen go pale at the mere mention of "Franklin." Some can barely control their rage. "No color!. . ." they stammer. "No serial interface!" "Screwed up ROMs!" "Big and stupid-looking!" ". . . programs won't run!" Sometimes, just a puff of dust comes out of their mouths and they faint away behind the counter.

Sure, there's no color, but I don't have a color monitor yet. When I'm ready for that big jump, Franklin has a hard-wired board that will do the trick. As far as screwed-up ROMs and programs that won't run? Well, I haven't found an Apple program that won't run on my old Ace 100 system. The Ace 1000 and the new 1200 must be at least as good. Yes, my beast could almost store an Apple inside it—for all intents and purposes, it does. But that's okay. I like a sense of solidity. Besides, there's that orange racing stripe.

Being a Franklin owner, I've grown accustomed to that blank stare I receive at cocktail parties when someone asks me what kind of computer I own. I reassure them by saying, "It's sort of a great, big Apple." They nod with approval and I am admitted back into their circle.

> **"I haven't found an Apple program that won't run on my old Ace 100 system. . . . Besides, there's that orange racing stripe."**

The same thing happened with the computer club at work. Most members own either an Apple, or the TRS Color Computer. They are electronic engineers, so forgive them. By the time they finish marching around inside the Radio Shack products, wierd hybrid machines are born. I joined and heads swiveled around when I announced my brand—but, surprisingly, I found I was not alone and those in the club who shared my eccentricity were normal-appearing human beings without scales or horns. I almost wept. In the meantime, all our software literature and hardware purchase lists have been amended to read "Apple/Franklin compatible."

And so, you Apple folk, while you ponder the Lisa, the Macintosh and the IIe, think about us, your brothers, playing with our keypads, buying tons of Apple software, buying crates of Apple peripherals, programming the same Applesoft FOR/NEXT loop. We just couldn't help making Aces out of ourselves. ■

Ampersand Number Converter

This may be the world's handiest number converter.

by Richard L. Emerson

Working with Apple utility programs usually requires some conversion of numbers from hexadecimal to decimal and the reverse. The conversion is complicated by the frequent use of negative numbers for memory locations above hexadecimal $7FFF (decimal 32767). The number conversion can be accomplished using printed conversion tables, a hand calculator, pencil, paper and a little time. This, however, violates a basic rule of the computer user's unwritten code—let your computer do the calculating.

Hex-dec conversion programs show up frequently in computer magazine articles. Some use Basic and others use assembly language. They convert one or many digits, in one or both directions, and are proclaimed in some cases to be the "world's shortest converter" or the "ultimate converter." But there is a problem; they are not always easy and convenient to use.

This number conversion program operates without the bother of entering the monitor and then returning to Basic. It makes conversions without dumping the program and enters hexadecimal numbers, positive decimal numbers and negative decimal numbers without elaborate selection routines. All appropriate conversions are displayed together, and the program also reminds the user how to enter the numbers to avoid having to memorize or look up the required procedures.

The assembly language and Applesoft Basic programs in the following two listings allow the instructions to be displayed and the program to be put in the computer memory by one simple keyboard entry. Once the program is in memory, number conversion is as easy and as complete as the following entry and display examples indicate:

"This, however, violates a basic rule of the computer user's unwritten code—let your computer do the calculating."

```
]&53248
$D000    53248    -12288
]&-151
$FF69    65385    -151
]&182
$00B6    182
]&$FDED
$FDED    65005    -531
]&$EF
$00EF    239
```

Negative conversions are displayed only when the entry is greater than $7FFF. Numbers greater than 65535 cause an ? ILLEGAL QUANTITY ERROR message. Hexadecimal digits other than 0 to F and decimal digits other than 0 to 9 result in ? SYNTAX ERROR. If more than four hex digits are entered, the extras are ignored.

While the applications are not too evident, decimal formulas can be entered also, as indicated by the following entry and display examples. The results are displayed as integers not greater than 65535.

```
]&4*4096
$4000    16384
]&450.95-327.15
$007B    123
```

This handy program has one resident bug. The entry of any hexadecimal number containing the sequence DEF, such as &$FDEF, causes a ? SYNTAX ERROR message. DEF is a reserved word used in the define function command as indicated on page 122 of the *Applesoft Basic Programming Reference Manual*. This limitation is not much of an inconvenience since the 32

Address correspondence to Richard L. Emerson, 7921 Borson St., Downey, CA 90242.

memory locations involved are not significant locations.

Assembler Program Listing

The Apple Assembler/Editor was used for the Ampersand Number Converter assembly language listing. See Listing 1. The program is located on page 3 of memory from \$0345 to \$03CA, at the end of the \$0300 to \$03CF space available for machine language programs. This leaves room for other short routines that start at \$0300. Lines 19 to 29 list the many monitor and Applesoft Basic locations and subroutines used to minimize the program length.

When the program is BRUN, the instructions in lines 35 to 41 set the ampersand (&) vector in locations \$03F5 to \$03F7 to make the computer jump to the start of the Convert Numbers (CONV) routine at \$0355 when the & command is entered. Refer to page 65 of the *Apple II Reference Manual* for a reference to these locations.

The CONV routine starts by checking the accumulator (A) to see if the first character after the & in the entry is a hex sign (\$). If it is, the computer falls through to the Hex Input routine in lines 50 to 77. This routine converts the first four ASCII hex digits in the entry into a two-byte number. It is based on the monitor GETNUM subroutine located from \$FF8A to \$FFBD with the entry point at \$FFA7. The modifications include use of the LINNUM locations to assemble the two-byte number, conversion of ASCII character codes with the high bit reset, and stopping after the first four digits. The use of the Applesoft subroutine Data in line 76 is necessary to avoid a ? SYNTAX ERROR message even for valid entries. This subroutine moves the text pointer (TXTPTR at \$B8 to \$B9) to the end of the entry.

If the first character after the & in the entry is not a hex sign (\$), the computer branches to the Decimal Input (DEC) routine in lines 81 and 82. The two powerful Applesoft subroutines FRMEVL and GETADR used there evaluate the decimal number or formula in the entry, leave the result in the floating point accumulator (FAC at \$9D to \$A3), and then convert the re-

***Listing 1**. Ampersand Number Converter.*

```
SOURCE FILE: AMP NUMBER CONV
                 1 ********************************************
0000:            2 *
0000:            3 *                    AMPERSAND
0000:            4 *
0000:            5 *               NUMBER CONVERTER
0000:            6 *
0000:            7 *               BY R. L. EMERSON
0000:            8 *
0000:            9 * 1  BRUN PROGRAM
0000:           10 * 2  ENTER HEX OR DECIMAL NUMBERS AS
0000:           11 *    &$ED24, &60708, OR &-4828
0000:           12 * 3  HEX AND POSITIVE AND NEGATIVE
0000:           13 *    DECIMAL NUMBERS WILL BE DISPLAYED
0000:           14 * 4  SOLVES DECIMAL FORMULAS WITH
0000:           15 *    INTEGER ANSWERS TO 65535, ENTER &2*3
0000:           16 *
                17 ********************************************


0050:           19 LINNUM  EQU  $50       ;2 BYTE NUMBER LOCATION
0200:           20 BUF     EQU  $0200     ;INPUT BUFFER
03F5:           21 AMPERV  EQU  $03F5     ;AMPERSAND VECTOR LOCATION
D995:           22 DATA    EQU  $D995     ;MOVE TXTPTR TO END OF INPUT
DD7B:           23 FRMEVL  EQU  $DD7B     ;FORMULA IN BUF TO FAC
E752:           24 GETADR  EQU  $E752     ;FAC TO 2 BYTE INTEGER IN LINNUM
ED24:           25 LINPRT  EQU  $ED24     ;PRINT DECIMAL NUMBER IN X,A
F941:           26 PRNTAX  EQU  $F941     ;PRINT HEX NUMBER IN X,A
F948:           27 PRBLNK  EQU  $F948     ;PRINT 3 BLANKS
FD8E:           28 CROUT   EQU  $FD8E     ;PRINT CARRIAGE RETURN
FDED:           29 COUT    EQU  $FDED     ;PRINT BYTE IN A


----- NEXT OBJECT FILE NAME IS AMP NUMBER CONV.OBJ0
0345:           31         ORG  $0345     ;AT END OF $0300-$03CF SPACE


0345:           33 *** SET AMPERSAND VECTOR ***

0345:A9 4C      35         LDA  #$4C      ;SET AMPERSAND VECTOR TO JUMP
0347:8D F5 03   36         STA  AMPERV    ; TO CONV
034A:A9 55      37         LDA  #>CONV
034C:8D F6 03   38         STA  AMPERV+1
034F:A9 03      39         LDA  #<CONV
0351:8D F7 03   40         STA  AMPERV+2
0354:60         41         RTS


0355:           43 *** CONVERT NUMBERS ***

0355:C9 24      45 CONV    CMP  #$24      ;CHECK FOR HEX $ IN A
0357:D0 35      46         BNE  DEC       ;BRANCH ON DECIMAL INPUT

0359:           48 * HEX INPUT *

0359:A9 00      50         LDA  #$00      ;FIRST 4 ASCII HEX DIGITS IN BUF
035B:85 50      51         STA  LINNUM    ; TO 2 BYTE NUMBER IN LINNUM
035D:85 51      52         STA  LINNUM+1  ; BASED ON MONITOR GETNUM
035F:A0 02      53         LDY  #$02      ; ROUTINE FROM $FF8A TO $FFBD
0361:B9 00 02   54 NXTCHR  LDA  BUF,Y     ;START AFTER & AND $
0364:F0 22      55         BEQ  EOL       ;BRANCH ON END OF LINE $00,
0366:C8         56         INY            ; LESS THAN 4 DIGITS
0367:49 30      57         EOR  #$30      ;CONVERT ASCII 0-9, $30-$39,
0369:C9 0A      58         CMP  #$0A      ; TO $00-$09
036B:90 09      59         BCC  DIG       ;BRANCH ON 0-9
036D:69 88      60         ADC  #$88      ;CONVERT ASCII A-F, $41-$46,
036F:C9 FA      61         CMP  #$FA      ; TO $FA-$FF (CARRY SET)
0371:B0 03      62         BCS  DIG       ;BRANCH ON A-F
0373:4C 94 03   63         JMP  PRINT     ;NOT HEX DIGIT
0376:A2 03      64 DIG     LDX  #$03
0378:0A         65         ASL  A         ;DUMP USELESS HIGH ORDER
0379:0A         66         ASL  A         ; NIBBLES $0 AND $F
037A:0A         67         ASL  A
037B:0A         68         ASL  A
037C:0A         69 NXTBIT  ASL  A         ;LOW ORDER BIT INTO CARRY
037D:26 50      70         ROL  LINNUM    ;CARRY BIT INTO LINNUM
037F:26 51      71         ROL  LINNUM+1
0381:CA         72         DEX
0382:10 F8      73         BPL  NXTBIT    ;STOP AT 4 BITS
0384:C0 06      74         CPY  #$06
0386:90 D9      75         BCC  NXTCHR    ;STOP AT 4 DIGITS
0388:20 95 D9   76 EOL     JSR  DATA      ;TO AVOID SYNTAX ERROR MESSAGE
038B:4C 94 03   77         JMP  PRINT

038E:           79 * DECIMAL INPUT *
```

Listing continued.

Listing continued.

```
038E:20 7B DD    81 DEC     JSR  FRMEVL    ;FORMULA IN BUF TO FAC AND THEN
0391:20 52 E7    82         JSR  GETADR    ; TO 2 BYTE INTEGER IN LINNUM

0394:            84 * PRINT NUMBERS *

0394:A9 A4       86 PRINT   LDA  #$A4      ;PRINT HEX $
0396:20 ED FD    87         JSR  COUT
0399:A6 50       88         LDX  LINNUM
039B:A5 51       89         LDA  LINNUM+1
039D:20 41 F9    90         JSR  PRNTAX    ;PRINT HEX NUMBER
03A0:20 48 F9    91         JSR  PRBLNK    ;PRINT 3 BLANKS
03A3:A6 50       92         LDX  LINNUM
03A5:A5 51       93         LDA  LINNUM+1
03A7:20 24 ED    94         JSR  LINPRT    ;PRINT POSITIVE DECIMAL NUMBER
03AA:20 48 F9    95         JSR  PRBLNK    ;PRINT 3 BLANKS
03AD:A5 51       96         LDA  LINNUM+1  ;NO NEGATIVE PRINT UNLESS >$7FFF
03AF:10 16       97         BPL  END
03B1:A9 AD       98         LDA  #$AD      ;PRINT MINUS -
03B3:20 ED FD    99         JSR  COUT
03B6:18         100         CLC            ;CONVERT POSITIVE DECIMAL
03B7:A5 50      101         LDA  LINNUM    ; TO NEGATIVE DECIMAL
03B9:49 FF      102         EOR  #$FF
03BB:69 01      103         ADC  #$01
03BD:AA         104         TAX
03BE:A5 51      105         LDA  LINNUM+1
03C0:49 FF      106         EOR  #$FF
03C2:69 00      107         ADC  #$00
03C4:20 24 ED   108         JSR  LINPRT    ;PRINT NEGATIVE DECIMAL NUMBER
03C7:20 8E FD   109 END     JSR  CROUT     ;PRINT CARRIAGE RETURN
03CA:60         110         RTS

*** SUCCESSFUL ASSEMBLY: NO ERRORS
```

```
100      REM   NUMBER CONVERTER MAKER -- MAKES AN EXEC FILE NUMBER
               CONVERTER TO DISPLAY INSTRUCTIONS AND RUN THE
               AMPERSAND NUMBER CONVERTER BINARY PROGRAM
110      REM   TO USE, ENTER 'EXEC NUMBER CONVERTER'
120  PRINT CHR$(4);"OPEN NUMBER CONVERTER"
130  PRINT CHR$(4);"WRITE NUMBER CONVERTER"
140  PRINT "PRINT"CHR$(34)"TO CONVERT NUMBERS, ENTER HEX OR_
     DECIMALNUMBERS AS &$ED24, &60708, OR &-4828"CHR$(34)
150  PRINT "BRUN AMP NUMBER CONV" :
         REM   LOADS FROM $0345 TO $03CA
160  PRINT CHR$(4);"CLOSE NUMBER CONVERTER"
170  END
```

Listing 2. Number Converter Maker.

sult into a two-byte positive integer in LINNUM. If the resulting number is negative, the positive integer in LINNUM is 65536 minus the negative number.

Both the Hex Input and Decimal Input routines display the hex and positive and negative decimal numbers by means of the Print Numbers (PRINT) routine in lines 86 to 110. The hex sign ($) is printed followed by the contents of LINNUM in the X and A registers as a hex number using the monitor subroutine PRNTAX. Three blanks are printed by PRBLNK and then the X and A registers are loaded again from LINNUM to print the positive decimal number with Applesoft subroutine LINPRT.

After more spaces, the high order byte of LINNUM is checked to see if bit 7 is set indicating a number greater than $7FFF. If not, the computer branches to END and no negative decimal number is displayed. If bit 7 is set, however, a minus sign (−) is printed and the equivalent negative decimal number is obtained by complementing the number in LINNUM with EOR #$FF instructions and then adding $01. This has the same effect as subtracting the number from 65536. The negative decimal number is then displayed using LINPRT. A carriage return is output through monitor subroutine CROUT to space the number of displays apart for easier reading on the screen. A return from subroutine instruction RTS returns the computer to an Applesoft prompt (]) for the next entry.

The assembly language program can be assembled and saved on a disk by using your own assembler program. Or the 134 bytes of machine code can be entered directly into memory as indicated on page 44 of the *Apple II Reference Manual.* The machine code should be saved on a disk as a binary file by entering BSAVE AMP NUMBER CONV, A$0345, L134.

Applesoft Program Listing

While the Ampersand Number Converter can be put into operation by a simple BRUN AMP NUMBER CONV keyboard entry, the instructions for using the program can be displayed and the binary file BRUN all at the same time by use of a short EXEC file on the same disk.

The Applesoft Basic listing Number Converter Maker should be entered as shown in lines 120–170. (See Listing 2.) The REM statements are not required.

"The Applesoft program should be saved on a disk, at least until correct operation of the text file is confirmed."

When this program is run, it creates a text file on a disk with the file name Number Converter. The Applesoft program should be saved on a disk, at least until correct operation of the text file is confirmed. When EXEC NUMBER CONVERTER is entered, the text file displays the instructions in line 140 and then the AMP NUMBER CONV binary file is BRUN ready for use.

For a description of the creation and operation of EXEC files, see chapter 7 of the *Apple II DOS Manual.* CHR$(4) is the control-D necessary for use of DOS commands within a program. CHR$(34) provides quotation marks (") for DOS that are necessary to allow the instructions within conventional quotation marks to be displayed on the screen.

The Ampersand Number Converter will remain in memory for use until Applesoft is booted again, unless the page 3 memory space is used for some other assembly language program. ■

Circle 306 on Reader Service card.

NEW

INTRODUCING:
MICRO
TUTOR™

ONLY $99.95

JUST FOR YOU:
- TEACH STUDENTS, CHILDREN, OR TRAINEES ANY SUBJECT MATTER
- OBSERVE MICRO TUTOR'S USER FRIENDLINESS
- PRINT HARD COPIES OF SESSIONS (TEACHERS CAN PRINT TESTS)
- STORE SESSIONS FOR LATER USE
- DEFINE NUMBER OF GUESSES ALLOWED FOR EACH QUESTION
- CREATE MULTIPLE CHOICE, SIMPLE ANSWER, FILL IN THE BLANK, AND MATCHING QUESTIONS
- UTILIZE SIMPLE GRAPHICS FOR CHARTS, DIAGRAMS, OR GRAPHS
- CHAIN SESSIONS TOGETHER

AND MORE:
- RECOMMENDED BY PARENTS AND TEACHERS
- BUILT IN EDITING COMMANDS (POWERFUL YET EASY TO USE)
- HELPS INCREASE LEARNING AND RETENTION
- DISPLAYS RESPONSE TO CORRECT/INCORRECT ANSWERS
- OFFERS EITHER SOUND/NO SOUND MODE
- PROVIDES EASY TO FOLLOW INSTRUCTIONS AND READY TO RUN EXAMPLES

LOCUS SYSTEMS

P.O. BOX 248 — North Wilkesboro, N.C. 28659
CALL OR WRITE TODAY ... (919) 838-4166
VISA • MASTERCARD ACCEPTED
PLEASE ADD $2.00 FOR SHIPPING AND HANDLING
NC RESIDENTS ADD 4% SALES TAX

• APPLE II • DOS 3.3 • 48 K MEMORY • 1 DISK DRIVE

• APPLE IS A TRADEMARK OF APPLE COMPUTER, INCORPORATED

NEW NEW NEW

Circle 73 on Reader Service card.

PROTECT YOUR APPLE* KEYBOARD
WITH
PLEXA-LOK

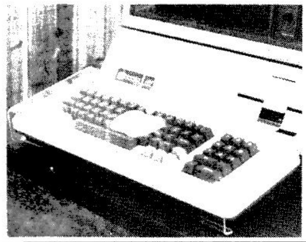
PROTECT YOUR EXPENSIVE INVESTMENT

OFFERED FOR THE FIRST TIME *PLEXA-LOK* COMES WITH A 30-DAY MONEY BACK GUARANTEE IF NOT SATISFIED!

PLEXA-LOK slips up and over the keyboard – then gently snaps into position.

- Your valuable computer is protected from objects and spills directly on top of keyboard which could cost hundreds of dollars to repair!
- *PLEXA-LOK* allows your secretary to go on break without having to worry about visitors accidentally destroying their hours (and your $) of work.

- **ENHANCES** looks of your system
- **PROTECTS** keyboard from dust
- **ALLOWS** computer to remain on while unattended

* TM APPLE COMPUTER Inc.

MON.–FRI. 9:00-5:00

MasterCard and Visa Accepted

Allow 4-6 Weeks Delivery

Dealer Inquiries Welcome

LE
LAST ELECTRONICS
P.O. BOX 1300S
SAN ANDREAS, CA 95249
(209) 754-1800

INTRODUCTORY SPECIAL

APPLE II	$19.95
APPLE III	$24.95
FROSTY APPLE	1.50 extra

Prepaid UPS Continental USA

CA Residents Add 6% Tax

Circle 387 on Reader Service card.

WHEN THE CHIPS ARE DOWN . . .

"master diagnostics +plus"

It's reasonable to suspect that every Apple owner has wondered at some time or another, if all those chips are working up to par, or if it's time for Disk Speed Calibration, Head Cleaning or General Maintenance . . .

Master Diagnostics Plus is a comprehensive diagnostics package for the Apple computers. As a diagnostic set, it can save unhooking all the peripherals and carrying an Apple to the dealer every time a minor hardware problem is suspected or periodic maintenance procedures are required, such as DISK SPEED CALIBRATION, MONITOR ALIGNMENT, DRIVE HEAD CLEANING, etc., etc. Necessary maintenance is made easy by the routines and documentation provided.

For those without a service center nearby, the diagnostic routines can save travel or shipping and considerable downtime for minor problems. Anyone fighting glitches in a program would rest a little easier knowing the computer, at least, is working properly.

As quoted in NIBBLE magazine:
"**Master Diagnostics Plus** is an impressive collection of diagnostic routines for the Apple II and Apple II Plus. It is capable for supplying sufficient information so you know whether or not your computer is performing normally. Regular use of the maintenance routines and supplies will help to insure top-notch operation. The peace of mind afforded by being able to regularly monitor computer performance is well worth the price of the program. This package should be in the library of every Apple user . . ." Also received "AAA" reviews in every U.S. Micro Mag.

Specify Version II or //e
REQUIRES: 48K, FP IN ROM
1 DISK DRIVE, DOS 3.3
unlimited warranty

master diagnostics $55.00
master diagnostics +plus $75.00

APPLE is Registered Trademark of Apple Computer Co.

THE TESTS INCLUDE

MOTHERBOARD ROM TEST	DISK DRIVE SPEED CALIBRATION	MONITOR SKEWING TESTS
APPLESOFT CARD TEST	DRIVE HEAD READ/WRITE TEST	MONITOR & MODULATOR CALIBRATION
INTEGER CARD TEST	WRITE PROTECT SWITCH TEST	MONITOR TEXT PAGE TEST
MOTHERBOARD RAM TEST	DRIVE HEAD CLEANING ROUTINES	MONITOR TEST PATTERN
16K RAM CARD TEST*	DISK DRIVE MAINTENANCE	MONITOR & TV YOKE ALIGNMENT
AUX RAM TEST *	DC HAYES MICROMODEM II TEST	LO-RES COLOR TESTS
80 COLUMN CARD TEST *	PADDLE & SPEAKER TEST	HI-RES COLOR TESTS
PARALLEL CARD TEST	PADDLE & BUTTON TEST	LISSAJOUS PATTERNS
SPEAKER FUNCTION TEST	PADDLE DRIFT TESTS	RND HI-RES COLOR GENERATOR
SQUARE WAVE MODULATION	INTERNAL MAINTENANCE	GENERAL MAINTENANCE
ON BOARD HELP	FORTY PAGE MANUAL	* APPLE //e

THE +plus
master diagnostics +plus provides everything needed to maintain your computer. The entire package is housed in our own molded case to protect against static electricity, x-ray and other contaminants.

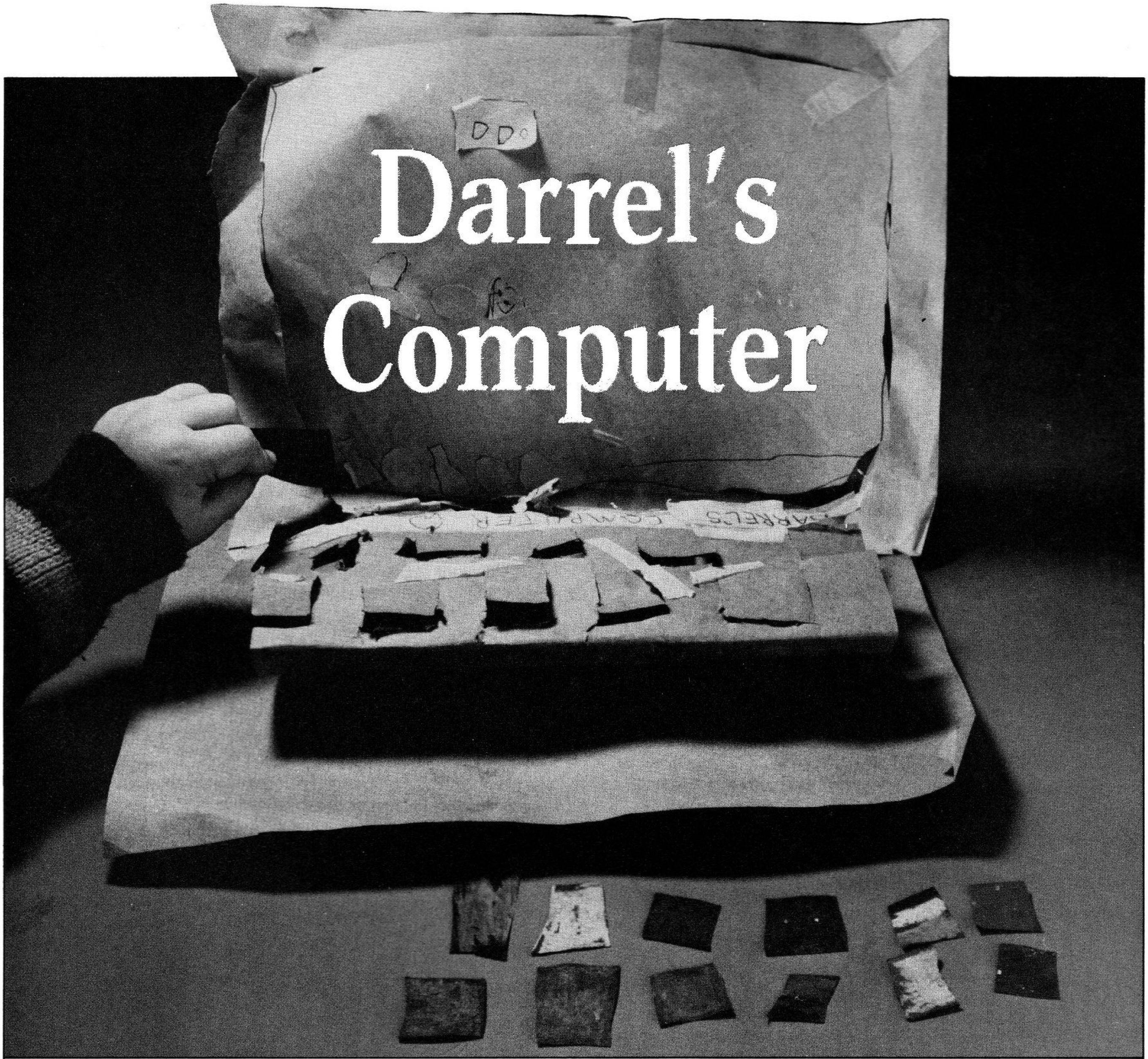
Included in the kit is:
- THE DIAGNOSTICS DISKETTE
- FORTY PAGE PROCEDURE MANUAL
- HEAD CLEANING KIT + 2 CLEANING DISKS
- CRT SCREEN CLEANER
- COMPUTER/DRIVE HOUSING CLEANER
- REUSEABLE CHAMOIS TIPPED WANDS

ORDER TOLL FREE ANYTIME or SEE YOUR NEAREST DEALER
MasterCard, Visa & C.O.D. users call: 1-800-835-2246
Kansas Residents call: 1800-362-2421
Add: $5.00 shipping outside U.S. & Canada

MasterCard VISA

NIKROM®
Technical Products, Inc.
25 PROSPECT STREET • LEOMINSTER, MA 01453

Darrel's Computer



by Tobi Hoffman

When we first acquired our Apple in the spring of 1980, I wondered what its influence might be on the children. Computers were certainly not a part of my home environment; I never expected that I would own one.

Our five-year-old has shown us that the computer is an integral part of the house. Darrel loves to create houses, cars, people, etc., out of paper, and his latest homemade toy is a computer! It began with a flattened tube of orange paper on a blue paper base. The keys are a series of cuts, with the paper folded so that they stick up; other pieces of paper are pasted on as switches, one for the computer, the other for the disk drive, which is a slot on the body of the machine. Small squares of black paper are the disks, complete with holes. The drive latch is a lollipop stick that hooks into a taped-on strip of paper. Disk storage is inside the computer itself, where disks tend to end up when inserted. The monitor is a frame of pink paper, with a rectangle of light green paper taped inside.

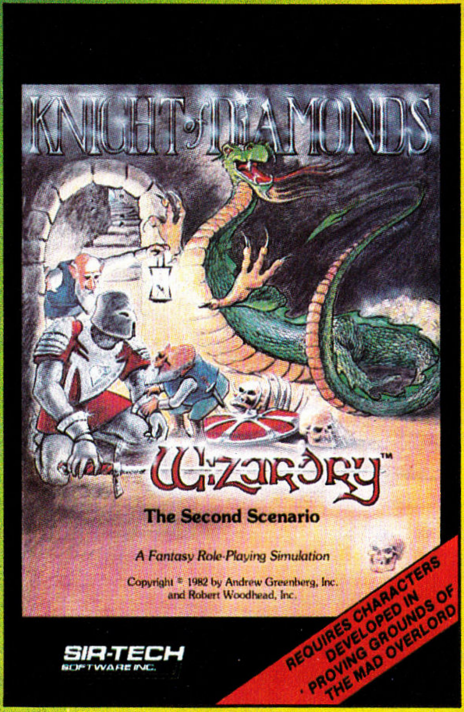
Although the computer console in the picture might seem to be backwards, judging by the words "Darrel's Computer" written across it, Darrel assures me that that is how it is supposed to be; the disk drive slot has to be in the back, because that is where the disk drives are!

Darrel has just begun to program games into his computer. The picture shows the Pac-Man characters on the screen. Another screen, for Alien Invaders, is being prepared. He does have a collection of disks, which I assume are general utility programs. One of them, he tells me, is Super-Text.

He has not told me of any plans for marketing his computer, but with time, he could go far! ■

Address correspondence to Tobi Hoffman, 58 Hilldale Road, Ashland, MA 01721.

Wizardry



KNIGHT OF DIAMONDS . . . THE SECOND SCENARIO

The characters with whom you fought through to the 10th level in "Proving Grounds" must now call upon every ounce of resourcefulness to save the City of Llylgamyn. In fact, they must have reached the 13th level of power merely to survive!

The same high resolution, 3-dimensional visual effects that thrilled you in the first scenario are here—and more. "Knight of Diamonds," so eagerly anticipated by Wizardry players everywhere, surpasses highest expectations for story, action and complexity.

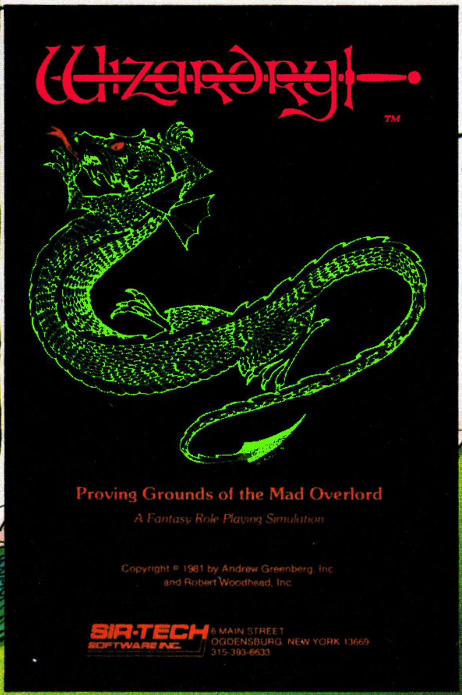
If successful, you will return the enchanted staff of Gnilda to Llylgamyn and *become* the Knight of Diamonds, but formidable adversaries block you at every turn. To begin your quest, simply **place yourself under the spell of Wizardry®**.

THE LEGENDARY FANTASY ROLE-PLAYING SIMULATION

. . . In "Proving Grounds of the Mad Overlord" you begin a challenging and totally absorbing journey through a 10-level, 3-dimensional maze. Create, then command, a hearty band of up to six adventurers who all must cooperate to explore a deep and mysterious maze in search of loot and glory.

The mages cast spells; thieves plot to steal treasure; and warriers battle the monstrous fiends of the underworld. In Wizardry®, no one remains unchanged; each member of the party grows in age, experience and, you hope, wisdom.

The puzzles, passageways and personalities in "Proving Grounds" have already fascinated and astounded the computer world and created a dedicated following of players. Find out why—**place yourself under the spell of Wizardry®.**



ALL SOFTWARE AVAILABLE FOR THE APPLE AND IBM PC AT YOUR FAVORITE RETAILER

SOFTWARE THAT COMPLIMENTS YOUR INTELLIGENCE

SIR-TECH SOFTWARE INC.
6 MAIN STREET
OGDENSBURG, N.Y. 13669
(315) 393-6633

Circle 218 on Reader Service card.

Undeleter

by Malcolm S. Cox

The purpose of Undeleter is to do just as it sounds—undelete files. It works on Apple II (with 48K) 16-sector DOS files.

But what sorcery can this be, you may ask? How can something that is lost and gone forever be brought back to life? The answer is that when DOS *deletes* a program or file on disk, it never *really* deletes anything at all. DOS merely modifies the entry in the disk directory to signify that its associated disk space is available territory and can be overwritten during the next file or program save.

> **"How can something that is lost and gone forever be brought back to life?"**

More often than not, if you accidentally delete a file, you realize the mistake fairly promptly, making Undeleter's chances of saving the day quite good. If, however, you have already saved another file or two, the chances of a reprieve are pretty slim.

The task at hand is quite simple. You must locate the directory on disk, find the entry you are interested in, then change a couple of bytes back to their former state so that DOS regards the file as an intact or *undeleted* file.

The DOS READ, WRITE, TRACK, and SECTOR (RWTS) subroutine is very powerful and so thoroughly documented in the *Apple DOS Manual* (pages 94–98 and 124–127) that it is very easy to do all sorts of interesting things with it, even from Basic.

I chose to write Undeleter in Basic instead of assembler because a Basic program is much easier for me to write and debug than an assembly language program. Also, seldom is blinding speed an overriding consideration, so Undeleter's speed is totally adequate for its function.

Undeleter is heavily remarked so you can follow the thread of the logic without difficulty.

- Lines 120–130 set up a subroutine so that the X and Y registers are scrubbed and tidy enough to call on the DOS READ, WRITE, TRACK, AND SECTOR subroutine (RWTS).
- Lines 170–260 POKE around to set up the DEVICE CHARACTERISTICS TABLE (DCT).

Program listing. Undeleter.

```
100  REM UNDELETER
110  HOME : VTAB 3: HTAB 5: PRINT
     "APPLE DOS 3.3 UNDELETER - 4
     8K": POKE 34,5: VTAB 6
120  FOR I = 768 TO 776: READ A: POKE
     I,A: NEXT : REM MAKE RWTS DR
     IVER
130  DATA 169,183,160,232,32,217,
     3,96,00
140  REM ************************
150  REM SET UP I/O CNTROL BLOCK
160  REM ************************
170  POKE 47080,1: REM MUST BE 1
180  POKE 47081,96: REM SLOT#*16
190  POKE 47082,1: REM DRIVE #1
200  POKE 47083,254: REM VOL NO
210  POKE 47086,251: REM DCT LO
220  POKE 47087,183: REM DCT HI
230 ADDRESS = 38400: REM BUFFER A
     DDRESS
240  POKE 47088,00: REM INSERT LO
      BYTE OF BUFFER ADDRESS INTO
      IOB
250  POKE 47089,150: REM INSERT H
     I BYTE OF BUFFER ADDRESS INT
     OIOB
260  POKE 47092,1: REM READ MODE
270  REM ************
280  REM EXAMINE VTOC
290  REM ************
300  POKE 47084,17: REM TRACK NUM
     BER OF VTOC
310  POKE 47085,0: REM SECTOR NUM
     BER OF VTOC
320  CALL 768: REM LOAD VTOC INTO
      BUFFER
330 DTRACK =  PEEK (ADDRESS + 1):
      REM DIRECTORY TRACK IS 2ND
     BYTE OF VTOC
340 DSECTR =  PEEK (ADDRESS + 2):
      REM DIRECTORY SECTOR IS 3RD
      BYTE OF VTOC
350  REM *********************
360  REM LOAD DIRECTORY BUFFER
370  REM *********************
380  POKE 47084,DTRACK: REM TRACK
      NUMBER OF DIRECTORY
390  POKE 47085,DSECTR: REM SECTO
     R NUMBER OF DIRECTORY
400  CALL 768: REM GO LOAD UP BUF
     FER
410  REM ************************
     *******
420  REM DETERMINE NEXT TRACK AND
      SECTOR
430  REM ************************
     *******
440 DTRACK =  PEEK (ADDRESS + 1):
      REM DTRACK IS NEXT TRACK
450 DSECTR =  PEEK (ADDRESS + 2):
      REM DSECTR IS NEXT SECTOR
460  REM *********************
470  REM EXAMINE THE DIRECTORY
480  REM *********************
490  FOR ENTRY = 0 TO 6
500 OFFSET = ENTRY * 35 + 11
510 TYPE =  PEEK (ADDRESS + OFFSE
     T + 2)
520 CLASS$ = " INTACT ": IF  PEEK
     (ADDRESS + OFFSET) = 255 THEN
     CLASS$ = "DELETED "
530 NAME$ = ""
540  FOR A = ADDRESS + OFFSET + 3
      TO ADDRESS + OFFSET + 31
550 NAME$ = NAME$ +  CHR$ ( PEEK
     (A)): NEXT
560 FTRACK =  PEEK (A): REM IF DE
     LETED, FILE TRACK IS LAST CH
     AR OF FILE NAME
570 LOCKED$ = " ": IF TYPE > 127 THEN
     LOCKED$ = "*":TYPE = TYPE -
     128
580  IF TYPE = 0 THEN TYPE$ = "T"
```

Listing continued.

Address correspondence to Malcolm S. Cox, 787 Gantry Way, Mountain View, CA 94040.

Cider Vinegar

Bent on Business, May

There is an error in my "Bent on Business" column entitled "Ah, Payday!" in the May issue of *inCider*. I transposed the row and column designations for the example array labeled Figure 3. In a double-dimension array, correct array notation has the subscript for the row first, followed by the subscript for the column. The example used, QT(3,4), specifies row 3, column 4, not column 3, row 4 as I indicated. A special note of thanks to Mrs. M. Kratky, a teacher from Hickory Hills, IL, who found my mistake.

Greg Glau
PO Box 1627
Prescott, AZ 86302

Applesoft Adviser, May

There is a mistake in Listing 4 on page 36 of "The Applesoft Adviser" column in *inCider*'s May issue. In line 260 FNND should read FIND. Thus, the line will read:

260 Y = YR:M = F:ND = FIND(Z)

Without this correction the program yields a syntax error.

—the editors

Con Job, April

I enjoyed "Con Job" by Paul Raymer, but I think I've found a slight problem in the Sector Count program. The problem is that it does not correctly reveal the sectors used or the sectors still free. Since there are 560 sectors on a 3.3 disk with standard DOS, and DOS and the directory use 64 sectors, there are 496 sectors left for the user, as explained on page 135 of the *DOS Manual*. The program can be corrected by inserting GT = GT + 64 as line 255, and then changing 493 to 560 in line 260. The GT = GT + 64 adds on the 64 sectors already taken and the other change makes the sectors free correct.

Sandy Smith
Clearwater, FL

via Paul Raymer
3464 Townhouse Drive
Las Vegas, NV 89121

Listing continued.

```
590  IF TYPE = 1 THEN TYPE$ = "I"

600  IF TYPE = 2 THEN TYPE$ = "A"

610  IF TYPE = 4 THEN TYPE$ = "B"

620  IF TYPE = 8 THEN TYPE$ = "S"

630  IF TYPE = 16 THEN TYPE$ = "R
     "
640 SIZE =  PEEK (ADDRESS + OFFSE
     T + 33): IF SIZE = 0 THEN  GOTO
     850
650 SIZE$ =  STR$ (SIZE)
660  IF SIZE < 10 THEN SIZE$ = "0
     " + SIZE$
670  IF SIZE < 100 THEN SIZE$ = "
     0" + SIZE$
680  IF SIZE > 99 THEN SIZE$ =  STR$
     (SIZE)
690  IF CLASS$ = "DELETED " THEN
      INVERSE
700  PRINT LOCKED$;TYPE$;" "SIZE$
     ;" "NAME$;: HTAB 33: PRINT C
     LASS$;: NORMAL
710  IF CLASS$ = " INTACT " THEN
      GOTO 830
720  PRINT "UNDELETE THIS ENTRY ?
      ";: FLASH : PRINT " Y";: NORMAL

730  PRINT "ES OR ";: FLASH : PRINT
      "N";: NORMAL : PRINT "O";
740  HTAB 31: GET Q$: HTAB 1: CALL
      (64578): REM CLEAR LAST LINE

750  IF Q$ <  > "Y" THEN  GOTO 82
     0
760  POKE ADDRESS + OFFSET,FTRACK
     : REM RESET FILETRACK
770  POKE A,160: REM RESET LAST C
     HAR OF DIRECTORY NAME
780  POKE 47092,2: REM SET IOB TO
      WRITE MODE
790  CALL 768: REM REWRITE BUFFER
      BACK TO DISC
800  POKE 47092,1: REM RESET IOB
     ACK TO READ MODE
810  GOTO 500
820  IF Q$ <  > "N" THEN  GOTO 72
     0
830  NEXT ENTRY
840  GOTO 380
850  POKE 34,0: END
```

- Lines 270–340 create a VOLUME TABLE OF CONTENTS akin to the card catalog in a library to help in locating files.
- Lines 350–400 are meaty stuff; once you have set up, they make an easy task of moving a sector into RAM.
- Lines 410–450 enable you to find out where the next sector of interest is.
- Lines 460–700 build a catalog look-alike. All this is fully described in the *Apple DOS Manual*, pages 127–133. However, the secret is in line 560; DOS takes the track number of the deleted files' track and sector list and hides it as the last character of the directory name. To cover up the theft, the number 255 is left in its place.
- Lines 710–850 do the actual undeleting.

I and numerous relatives and friends have been using Undeleter for over a year now and it has prevented at least 14 nervous breakdowns to my knowledge. I hope it will prevent one for you! ■

The 80-Column List

by Mark J. Yannone

If you are unfamiliar or only somewhat familiar with the command POKE 33,33, then you may be in for a treat.

According to Apple's *Basic Programming Reference Manual*, POKE 33,W sets the width of the monitor display to the width W, in the range of 1 to 40, inclusive. So, everything that is printed on the screen appears in a window W characters wide. The advantage of this is that it creates a faster-scrolling, split screen effect: part of the display remains undisturbed as the data in the new window changes.

Another useful characteristic of this command appears in editing program lines. This is commonly used before listing a program or part of a program in order to eliminate all unnecessary spaces in mulitple-line program statements. Then, when using the → to copy, you'll no longer need to employ the free-cursor moves to skip over the troublesome spaces that

Address correspondence to Mark J. Yannone, 2202 N. Laurel Ave., Phoenix, AZ 85007.

atlantic Software

9 Spinning Wheel Rd. Monroe, CT 06468

UP TO 50% DISCOUNT

Apple—Atari—IBM

SPECIAL DISCOUNTS

Nibbles Away II	54.95
Replay II	130.00
Wildcard	110.00
PFS: File	82.50
Franklin Computer	Call
Apple IIe Computer	Call

GAMES

Program	List Price	Our Price
Starcross	39.95	**27.45**
Mask of the Sun	39.95	**29.95**
Blade of Blackpool	39.95	**27.45**
Tubeway	34.95	**24.00**
Aztec	39.95	**27.45**
The Prisoner II	32.95	**22.65**
Pinball Construction Set	39.95	**27.45**
Chop Lifter	34.95	**24.00**
The Arcade Machine	54.95	**42.00**
Castle Wolfenstein	29.95	**20.55**
Cannonball Blitz	34.95	**24.00**
Swashbuckler	34.95	**24.00**
Serpentine	34.95	**24.00**
Night Mission Pinball	29.95	**23.20**
Zork I, II, or III	39.95	**27.45**
Flight Simulator	33.50	**27.20**
Wizard and the Princess	32.95	**22.65**
Deadline	49.95	**34.25**
Time Zone	99.95	**65.00**
Sherwood Forest	34.95	**26.20**
Wizardry	49.95	**36.20**
Knight of Diamonds	34.95	**25.30**
Frogger	34.95	**24.00**
Sargon II	34.95	**25.30**
Cross Fire	29.95	**20.55**
David's Midnight Magic	34.95	**24.00**
Galactic Adventures	59.95	**41.25**
Miner 2049er	39.95	**31.00**
Bomb Alley	59.95	**41.00**
Lunar Leepers	29.95	**20.50**
Night Flight	29.95	**20.50**
Critical Mass	39.95	**27.50**
U-Boat Command	29.95	**20.50**
Zaxxon	39.95	**28.00**
Millionaire	79.95	**54.95**
The Dark Crystal	39.95	**27.50**
The Serpent's Star	39.95	**27.50**
Odesta Chess 7.0	69.95	**52.00**
Caverns of Freitag	29.95	**20.50**
The Vein Game	34.95	**24.00**
Wizardry III	39.95	**27.50**
Germany 1985	59.95	**41.25**
Ultima II	59.95	**41.25**
Rendezvous 1.1	39.95	**27.50**
Missing Ring	29.95	**20.50**
Wavy Navy	34.95	**24.00**
Wiz Plus	39.95	**27.50**
Microscopic Journey	39.95	**24.00**
Microbe	44.95	**30.95**
Police Artist	34.95	**24.00**
Beneath Apple Manor Spec. Ed.	29.95	**20.50**
Solitaire & Cribbage	34.95	**24.00**
Spectre	29.95	**20.50**
Go	34.95	**24.00**
Jawbreaker II	29.95	**20.50**
The Desecration	39.95	**27.50**

BUSINESS

Program	List Price	Our Price
DB Master	229.00	**159.00**
DB Master Utility Pak 1	99.00	**68.00**
DB Master Utility Pak 2	99.00	**68.00**
D Base II	700.00	**460.00**
PFS: Report	125.00	**78.00**
PFS: Graph	125.00	**90.00**
Visitrend/Visiplot	300.00	**224.00**
Visicalc 3.3	250.00	**184.00**
The General Manager II	229.95	**174.00**
Magic Window II	149.95	**105.00**
Versaform	389.00	**268.00**
Screenwriter II	129.95	**93.50**
Word Handler	199.00	**137.50**
Super Text 40/80	175.00	**129.00**
Wordstar	495.00	**335.00**
Mailmerge	250.00	**172.50**
The Dictionary	99.95	**71.95**
Supercalc (Req. Softcard)	295.00	**203.50**
Letter Perfect	149.95	**107.95**
Sensible Speller	125.00	**82.00**
1st Class Mail	74.95	**60.00**
Real Estate Analyzer II	195.00	**134.50**
Pie Writer	149.95	**105.00**
Spell Star	250.00	**172.50**
Magic Mailer	99.95	**50.00**
Pascal Programmer	150.00	**115.00**
Data Fax	199.00	**139.00**

EDUCATIONAL

Program	Our Price
In Search of the Most Amazing Thing	**27.45**
Hey Diddle Diddle	**20.50**
Facemaker	**23.00**
Snooper Troops 1 or 2	**31.00**
Story Machine	**24.00**
Kindercomp	**20.75**
Delta Drawing	**41.25**
Rhymes and Riddles	**20.75**
Sticky Bear Numbers	**29.95**
Sticky Bear ABC	**29.95**
Sticky Bear Bop	**29.95**
Old Ironsides	**29.95**
Rocky's Boots	**37.43**
Gertrude's Puzzles	**33.50**
Gertrude's Secrets	**33.50**
Bumble Plot	**29.95**
Bumble Games	**29.95**
Juggle's Rainbow	**22.50**
Compu-Read	**20.75**
Compu-Math	**34.50**
Spelling Bee w/Primer	**27.50**
Algebra 1 2 3 or 4	**27.50**
SAT Wd. Attack Skills	**34.00**
PSAT Wd. Att'k Skills	**34.00**
Whole Brain Spelling	**24.00**
The New Step by Step	**59.00**
Speed Read Plus	**41.25**
Speed Reading	**51.45**
Decimals—Eduware	**34.00**
Fractions—Eduware	**34.00**
Perception	**17.00**
The Linguist	**28.95**
E-Z Learner	**29.95**
Counting Bee	**20.75**
Know Your Apple	**24.00**
Momptown Hotel	**29.95**
Momptown Parade	**29.95**
Magic Spell	**27.25**

HARDWARE

Printers	Our Price
Okidata	
Microline 82A FT	**439**
Microline 83A FT	**689**
Microline 92	**515**
Microline 93	**880**
NEC	
8023A w/Graph. Par.	**485**
Smith Corona TP-1	**629**
Epson	
FX 80	**550**
ProWriter	
8510AP Parallel	**415**
1510 Parallel	**675**

Monitors	Our Price
Amdek Color I	**339**
Amdek Color II	**725**
Amdek Color III	**435**
Amdek RGB Interface	**149**
Amdek 300G Hi-Res G	**149**
Sanyo 9'' Bl. & Wh.	**149**
Sanyo 9'' Green	**155**
Sanyo 12'' Green	**210**
Sanyo 13'' Color L/R	**375**
NEC 12'' Color	**325**
USI 12'' Amber	**169**
USI 12'' Green	**149**

Disk Drives	Our Price
Micro SCI A2	**299**
Micro SCI A2 w/contr	**400**
Rana Elite 1 (add on)	**299**
Rana Elite 1 w/contr	**400**

HOBBY

Program	List	Our Price
Dos Boss	24.00	**16.50**
The Graphics Magician	59.95	**44.95**
The Complete Graphics Sys.	69.95	**52.45**
Special Effects	39.95	**29.95**
Utility City	29.50	**20.25**
Lisa 2.5	79.95	**54.95**
Lisa Educational System	119.95	**82.45**
Apple Mechanic	29.50	**20.25**
TASC Compiler	175.00	**126.95**
Alpha Plot	39.50	**27.25**
Bug Byter	47.50	**40.00**
Zoom Graphix	39.95	**29.95**
Bag of Tricks	39.95	**29.95**
Merlin Assembler	64.95	**47.00**
Merlin Combo Pack	119.95	**80.45**
Hi-Res Secrets	124.95	**82.50**
Back It Up II Plus	59.95	**41.00**

HOME

Program	List	Our Price
Home Accountant	74.95	**48.00**
ASC II Express:	129.95	**85.00**
Data Capture 4.0	65.00	**44.65**
Typing Tutor II	24.95	**18.00**
Transend I	89.00	**67.50**
SAT English I	30.00	**21.65**
Master Type	39.95	**27.45**
ACE	39.95	**28.95**
Visiterm	100.00	**78.25**
Facemaker	34.95	**25.30**

ACCESSORIES

Item	Our Price
System Saver	**74.00**
Hayes Micromodem II	**265.00**
Hayes Micromodem II w/terminal program	**295.00**
The Joyport	**53.55**
TG Paddles	**28.55**
TG Joysticks	**39.95**
TG Select-A-Port	**42.85**
Microsoft Softcard	**238.95**
Videx Vidioterm Card	**248.00**
The Grappler	**130.00**
Microsoft RAM Card	**75.00**
Kraft Joystick	**48.50**
Videx Enhancer II	**121.00**
Mockingboard	**Call**
Corvus Hard Disks	**Call**
S.A.M. Automatic Mouth	**85.95**
Saturn 128K RAM	**375.00**

MEDIA

	Our Price
Elephant 5¼ SSSD/10	**22.00**
Verbatim 5¼ SSSD/10	**30.00**
Maxell 5¼ SSDD/10	**30.00**

VISA® MasterCard

Foreign Orders: Canada-$5.00, Europe-$15.00, APO-$5.00, P.O.-$5.00, Alaska-$5.00. For fast delivery send certified check, cashiers check or money order. We also accept C.O.D's, Mastercard and Visa. (Include card # and expiration date.) Conn. residents add 7.5% sales tax. Include $2.50 for shipping. We ship same day for most orders. Prices subject to change without notice. School purchase orders accepted.

ORDERS ONLY

TOLL FREE · 7 DAYS/WK

1-800-243-4536

For Inquiries and Connecticut Orders Call (203) 268-4596

If something you would like is not listed, just call. If we don't have it, we can get it.

Circle 310 on Reader Service card.

appear at the beginning and end of each row of print. A common command sequence in the immediate mode would be:

```
30  PRINT "**********************
     ****************************
     ****************************
     **"
40  PRINT "THIS IS HOW YOUR PRINT
     ER NORMALLY PRINTS A (LIST)I
     NG."
50  PRINT "**********************
     ****************************
     ****************************
     **"
```

Listing 1. Normal printing of a list.

HOME (to avoid confusion, this is important)
POKE 33,33 (sets the optimum window width for editing)
LIST
(edit your program)
LIST (to check your editing)
POKE 33,40 (returns the normal 40-character-wide window)

But there is yet another hidden reward for using this command. A POKE 33,W with W in the range of 1 to 33, inclusive, comes in very handy when you want a hard copy listing of your program. Normally, a printout of a listed program squeezes the statements into very hard to read 30-character lines that leave most of the page blank. Not only does this make a long program twice as long (or longer) and twice as hard (or harder) to read and follow, but it wastes a lot of paper.

In order to use the full 80-column width of your printer and paper, merely issue a POKE 33,33 command before listing. (For an additional 25 percent reduction in length and paper usage, change the printer line spacing from the normal six lines per inch to eight lines per inch.)

This simple command performs these functions in either immediate or deferred execution mode. ■

```
90  PRINT "*********************************************************************
***********"
100  PRINT "THIS IS HOW YOUR PRINTER WILL PRINT YOUR (LIST)INGS WHEN YOU USE THE
 COMMAND (POKE 33,33)."
110  PRINT "********************************************************************
************"
```

Listing 2. Printing using POKE 33,33.

Move and Restore

by Dave Schroeder

Move and Restore is a dual-package relocatable machine-language program. It consists of a high-res graphics mover ($300–34E) and a data pointer restorer ($34F–368).

High-Res Graphics Mover

Occasionally you want to display a color picture on some portion of the graphics screen and then put text around it. Normally for very small shapes a shape table will do just fine—except it is tedious to put together. You could also create the picture on the graphics tablet, BSAVE it, and then BLOAD it. This takes up a lot of space on the disk and is choppy when displayed.

To overcome this, I wrote a short program (see Listing 1) that will transfer a section of a graphics screen from one place to another. Thus, using the mover I can put several pictures on one graphics page, load it into memory at some other location, pull the pictures off when I need them and put them on the display screen like a photo in a scrap book. This sounds hopelessly complicated,

Address correspondence to David Schroeder, 224 Clark St., Brockport, NY 14420.

Location	Function
$19(25)	X-coordinate of where to put it
1A(26)	Y-coordinate of where to put it
$1B(27)	X-coordinate of where it is from
$1D(29)	Y-coordinate of where it is from
$1E(30)	Number of bytes across (7 columns per byte)
$1F(31)	Number of rows down
$E6(230)	High byte of display page ($20 or $40)
$FB(251)	High byte of where table is loaded

Table 1. List of locations and functions.

&	Gives an error no place to go.
& 1000	Restores to line 1000.
& N	Restores to line number in N.
& I% + 1	Restores to value of function.
& "HI"	Errors because of no line number.

Table 2. Ways to reset the data pointer.

but it works well and can be very handy.

Let's step through the program:

1. BLOAD a picture into HGR2. (To do this just type BLOAD YOUR PICTURE, A$4000.)
2. Type TEXT:HGR.
3. Then type CALL-151 (the Apple should have the machine language * showing).
4. Then type the following:

*19:00 00 64 00 64 05 14

*FB:40

*300G

*control-C and return (to get back to Basic).

Listing 1. Move and Restore, a dual machine-language utility.

```
SOURCE FILE: MOVE AND RESTORE
0000:                1 ;
0000:                2 ; MOVE AND RESTORE ARE BOTH
0000:                3 ; WRITTEN BY
0000:                4 ;
0000:                5 ;
0000:                6 ;          DAVID J. SCHROEDER
0000:                7 ;          224 CLARK STREET
0000:                8 ;          BROCKPORT, NY 14420
0000:                9 ;          716-637-4838
0000:               10 ;
0000:               11 ;
0000:               12 ; MOVER WILL MOVE A BLOCK OF
0000:               13 ; HIRES GRAPGHICS TO AND FROM
0000:               14 ; ANY WHERE.
0000:               15 ;
----- NEXT OBJECT FILE NAME IS MOVE AND RESTORE.OBJO
0300:               16          ORG   $300
F411:               17 HPOSN    EQU   $F411
0300:A6 19          18 START    LDX   $19
0302:A0 00          19          LDY   #$00
0304:A5 1A          20          LDA   $1A
0306:20 11 F4       21          JSR   HPOSN
0309:18             22          CLC
030A:A5 E5          23          LDA   $E5
030C:65 26          24          ADC   $26
030E:90 02          25          BCC   STORE
0310:E6 27          26          INC   $27
0312:85 26          27 STORE    STA   $26
0314:A5 26          28          LDA   $26
0316:85 F9          29          STA   $F9
0318:A5 27          30          LDA   $27
031A:85 FA          31          STA   $FA
031C:A6 1B          32          LDX   $1B
031E:A0 00          33          LDY   #$00
0320:A5 E6          34          LDA   $E6
0322:48             35          PHA
0323:A5 FB          36          LDA   $FB
0325:85 E6          37          STA   $E6
0327:A5 1D          38          LDA   $1D
0329:20 11 F4       39          JSR   HPOSN
032C:18             40          CLC
032D:A5 E5          41          LDA   $E5
032F:65 26          42          ADC   $26
0331:90 02          43          BCC   STORE1
0333:E6 27          44          INC   $27
0335:85 26          45 STORE1   STA   $26
0337:68             46          PLA
0338:85 E6          47          STA   $E6
033A:A4 1E          48          LDY   $1E
033C:B1 26          49 LOOP     LDA   ($26),Y
```

Listing continued.

Listing continued.

```
033E:91 F9                STA  ($F9),Y
0340:88                   DEY
0341:10 F9                BPL  LOOP
0343:C6 1F                DEC  $1F
0345:D0 01                BNE  MORE
0347:60                   RTS
0348:E6 1A       MORE     INC  $1A
034A:E6 1D                INC  $1D
034C:18                   CLC
034D:90 B1                BCC  START
034F:            ;
034F:            ; HERE BEGINS THE & RESTORE
034F:            ; FUNCTION. IT ALLOWS YOU
034F:            ; TO RESTORE TO A DATA LINE
034F:            ; IN AN APPLESOFT PROGRAM.
034F:            ;
034F:20 67 DD             JSR  $DD67
0352:20 52 E7             JSR  $E752
0355:20 95 D9             JSR  $D995
0358:20 1A D6             JSR  $D61A
035B:A5 9B                LDA  $9B
035D:85 7D                STA  $7D
035F:A5 9C                LDA  $9C
0361:85 7E                STA  $7E
0363:C6 7D                DEC  $7D
0365:C9 FF                CMP  #$FF
0367:D0 02                BNE  EXIT
0369:C6 7E                DEC  $7E
036B:60          EXIT     RTS
*** SUCCESSFUL ASSEMBLY: NO ERRORS
```

```
10 PRINT CHR$(4);"BLOAD MOVE AND RESTORE"
20 POKE 1013,79 : POKE 1014,48 : REM LOCATION $3F5-6
30 INPUT LN : RESTORE : REM SET TO THE START EACH TIME
40 &LN : READ N$
50 PRINT N$ : GOTO 30
60 DATA "THIS IS LINE NUMBER 60—THE FIRST DATA LINE"
70 DATA "THIS IS THE SECOND DATA LINE"
80 DATA "THIS ISN'T THE BEGINNING EITHER"
1000 DATA "THIS IS OUT IN THE COUNTRY"
9999 END
```

Listing 2. Demonstration of & restore function.

This takes a chunk of the page 2 screen starting at 100,100 (35 by 20 dots) and moves it to the upper left corner of the screen (0,0). You can POKE in the needed values, and then CALL them. Table 1 shows what each location does.

The picture to be cut up should be loaded in at a starting address ending in 00, such as $4000 (HGR2). Being off on the placement of the picture with respect to the starting point will cause a change in color every seven columns. Thus, if the color is wrong in the segment, move the starting point seven columns in either direction.

Data Pointer Restorer

The restorer, employing the & command, is a routine that covers what I consider to be a major flaw in Applesoft. Restorer functions simply as a data pointer reset to the beginning of the program. The routine is quite short and makes use of several existing routines to restore the data pointer to any line number in your program. To use it you simply set the & jump vector to the starting address of this routine. Ways to reset the data pointer are shown in Table 2.

The function restores to its line number or to the next lowest line number. All standard error trapping routines are employed. Listing 2 shows how to use the & RESTORE function. ■

Control Character Maze

by Paul M. Danzer

I have several hobbies, including an Apple II Plus which I use both under DOS 3.3 and CP/M. When I'm wearing my photography "hat" and press the shutter on my camera, I know that I may or may not get the photographic effect I want. When I turn my ham radio set on and press the microphone button, I may or may not make contact with a fellow in England.

But, using my Apple computer is different. When I press a key today it should do the same thing it did yesterday. And it does—as long as it's in the same mode as yesterday.

The control characters needed for shifting modes are the bane of my existence. My Apple is equipped with a Microsoft Z-80 processor and Videx Videoterm, as well as an extra 16K to bring it up to the full 64K limit. This rather common configuration provides a great deal of flexibility and capability, but also adds confusion.

Address correspondence to Paul M. Danzer, 2 Dawn Road, Norwalk, CT 06851.

Control Character

Control Character	ASCII CHR $ ( )	Apple Monitor Code	Applesoft and DOS 3.3	SSC (Serial Interface to Printer)	Printer
A	1				
B	2	Go to Language in ROM Lose Variables			
C	3	Go to Language in ROM Save Variables	Breaks Running Program Aborts Listing		
D	4	Display A, X, Y, P, S Registers	Print "Ctrl-D"; as Prefix for DOS Commands		
E	5				
F	6				
G	7	Bell 100 Hz for 1/10 Sec	Bell		
H	8	Backspace	Null Character in Get Command, Also Backspace		Backspace
I	9			Used as Prefix to Set SSC and Printer via SSC	Tab
J	10	Line Feed			Line Feed
K	11	Acts as IN# 0 Ctrl-K Input from Keyboard			
L	12				Form Feed
M	13		Return		Carriage Return Line Feed
N	14				
O	15				
P	16	Acts as PR#1 Ctrl-P Outputs via Slot 1 to Printer			
Q	17				Left Margin Set
R	18				Tab Set
S	19	Stop Listing	Stop Listing/ Restart Listing		Right Margin Set
T	20				Tab Clear
U	21	Forward Space	Forward Space		Print ¼
V	22				Print ½
W	23				Print ¢
X	24	Delete Line Being Typed	Delete/Ignore Line Being Typed		Margin Release
Y	25	Jump to $3F8			Automatic Underscore
Z	26				Note: CHR $ 26–31, 94, 96, 126–128 No Response. All Others Print a Character

Chart

Videoterm	Softcard	MBasic	Wordstar (Menus Only)
Shift Key Enable (with Hardware Mod.)	Shift Key Enable (with Hardware Mod.)	Enter Edit Mode on Current Line	
	Backslash	Backslash	
	Warm Boot	Break and Return to MBasic Command Level	
	End of Line and Start New Line		
Bell	Bell	Bell	
Backspace Cursor	Backspace, Also In Edit	Backspace, Delete Last Char Typed	
	Tab in Edit	Tab (Every 8 Columns)	
Move Cursor Down	Ret/Line Feed	Line Feed	Help
Clear to End of Screen	[	[	Block
Clear Screen	Used in Edit, Also CR/LF		
Carriage Return, Line Feed	Carriage Return		
Select Second Character Set		Halts Output—Program Continues to Run—Restart Output with Ctrl-O	On Screen
	Printer On/Off		Print
		Resume Program Execution after Ctrl-S	Quick
	Retype Line	Retypes Current Line	
Start/Stop Scrolling	Start/Stop Scrolling	Halts Program Execution	
Forward Space	Ctrl-I via Config. #10 ID		
?	Delete Line Being Typed	Delete Current Line	
Cursor to 0,0	Homes Cursor	Recovery from Reset	
Major Controls See Videx Manual	Used in PIP, EDIT Also See Videx Manual		

Typically I hit a control-C when trying to return to the Basic command level, then after several tries remember that this works only under MBasic with CP/M, not for Applesoft.

In the past when I converted an old program written in Integer Basic, I used control-L to clear the 80-column screen in several places. Then when I attempted to list the program on the CRT to check my changes, the system reacted to the control-L characters and cleared the screen each time one appeared. This made it impossible to read the commands just before each listed control-L.

Chalking the effect up to the Videoterm card, I tried a sneaky end run by listing the program on my printer. The reward was a surplus of ejected blank paper, since the control-L was now interpreted as a form feed and each time (and there were over a dozen times) a control-L appeared in the listing, the printer responded accordingly.

To help keep things straight I compiled a control character chart from the five or so manuals that accompanied my Apple, and the manuals for the Microsoft Z-80 card, Videx Videoterm and my printer. In doing so I made the remarkable discovery that even the best edited manual seems to lack accuracy when it comes to the index in the back. Perhaps both the authors and editors are a little tired by then.

After considerable experimentation I was able to verify the entries in the chart. The combination of equipment shown as additions to the Apple II Plus is sold commonly as the "Microsoft Premium Pac." In my case the printer is an SCM TP-1 fed by an Apple SSC card, but the printer/printer interface control characters appear to be standard for most combinations. The only known omission from the list is a full definition of control-Z, which would require condensing several pages of the Videoterm handbook into the small space allocated in the chart.

I hope this list is as helpful to you as it has been to me, and we can all save a little frustration. ■

by Bill O'Brien

Of Types and Varieties

Last month I discussed an Apple III printer device driver, the notorious .PRINTER driver supplied by Apple to power the Qume printer. Among other things, I described how to enter the System Configuration Program and modify it to meet specific needs.

The .PRINTER driver uses hardware handshaking. Usually, if your printer has some sort of a buffer or holding area, when it is almost full one of the signals sent to the Apple along the RS-232 lines drops to what the Apple recognizes as zero. (These signals are, more often than not, monitored on pins 20 and 4.) When this drop occurs the III stops sending data until the line goes high again.

There are, however, other kinds of printers out there in the world and other ways in which they work. Take the NEC 7715, or the Diablo 630 for instance. When the buffer in them is almost full, pin 19 goes low. In interfacing one of these printers, it would certainly help to move the signal from that pin over to pins 20 and 4 on the Apple. But there are ways to start and stop the flow of data by having the printer send the III a signal saying, "Slow down there Apple. I've got to digest this first before I'll be ready for any more."

This type of printer uses a signaling *protocol* (a set way of doing things) called either ETX/ACK or XON/XOFF between the printer and the computer. This protocol establishes agreement over when and when not to send data.

Generally speaking, under the ETX/ACK protocol the sending device puts out data in blocks, with the size of the block just big enough to almost fill the buffer of the receiving device. At the end of the block of data an ETX signal is sent. The printer, or other receiving device, sends back an ACK when the buffer is sufficiently empty to take the next block.

XON/XOFF, on the other hand, lets the sending device stream data out continuously. When the buffer of the receiving device is nearly full, it sends an XOFF character and the transmitter stops. When the buffer empties to a prespecified level, the receiver sends an XON character and the transmitter starts up again.

Unfortunately, the device type for our .PRINTER driver is CHAR, WRITE. It does *not* read from the device that's connected to it; and if it doesn't listen to the printer, it will never hear the desperate cry to, "STOP!"

To get around this, without having to shuffle wiring, we need a way to both send and receive data. For this purpose the wizards at Apple have provided the .RS232 driver. It is a bidirectional driver that uses the RS-232C signal levels, meaning the voltage on any of the lines can vary between a low of -12 volts to a high of $+12$ volts. The RS-232C is a specification, not a piece of equipment.

Take a look at Table 1. Except for the name, it looks almost the same as the listing for our poor, underprivileged Qume driver. However, there is an obvious difference. Under the heading Device Type we see CHAR, READ, WRITE!

I'll be tied! This driver can hear as well as talk! But in order to do that, and other wonderful things, it must be a longer program and therefore take up more space in SOS.DRIVER and the III's memory than .PRINTER does. Keep that in mind.

I'm going to presume that you remember how to edit a driver routine through the System Configuration Program (SCP). Let's go down to Configuration Block Data and edit it.

Item	Field	Value	
1	Device Name		.PRINTER
2	Device Type	$04	Char,Read,Write (range: 00..FF)
3	Device Subtype	$01	(range: 00..FF)
4	Driver Status		ACTIVE
5	Comment		
6	Configuration in Block Data		
	Slot Number		n/a
	Unit Number	$00	
	Manufacturer ID	$0001	Apple
	Block Count		n/a
	Version ID	1.10	

Press: Escape to exit to top Edit menu

Table 1. SCP editing choices for .PRINTER driver.

Address correspondence to Bill O'Brien at WABASA Consulting & Management, 111 Brook St., Scarsdale, NY 10583.

Parameter	Description	Value	
0	Baud Rate	03	110
		04	134.5
		06	300
		07	600
		08	1200
		09	1800
		0A	2400
		0C	4800
		OE	9600
1	Word length	22	7 bits, odd parity
		26	7 bits, even parity
		2A	7 bits, mark parity
		2E	7 bits, space parity
		00	8 bits, no parity
		42	6 bits, odd parity
		46	6 bits, even parity
		4A	6 bits, mark parity
		4E	6 bits, space parity
2	Waiting time	xx	Number of character times after a carriage return
3	Waiting time	xx	Number of character times after a linefeed
4	Waiting time	xx	Number of character times after a form feed
5	Protocol	00	XON/OFF
		40	ENQ(ETX)/ACK
6	Char. value	13	XOFF
		05	ENQ
		03	ETX
7	Char. value	11	XON
		06	ACK
8	Max. buffer	xx	Size of receiving buffer
9	Min. buffer	xx	Size to resume receiving
10	Block size	xx	With ETX(ENQ)/ACK, the length of the transmitted block
11	Mode	80	Hardware handshake
		00	Software protocol

Table 2. Description of possible device configuration block values.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
Ox –	08	22	00	00	00	00	13	11	DF	84	50	00				

Figure. Device configuration block data for .PRINTER.

Looking at the Figure, we find 12 information "bins" to be filled. Table 2 will give you an idea of what the possible values are. (Remember, they're all in hexadecimal notation.) The first five should be familiar from the Qume driver. The rest are indicators for the type of protocol we'll need and the data related to making it work correctly. Since I've previously discussed the reasons behind the values for the first five bins, I'll only mention the remaining seven here.

Byte 05 specifies the protocol to be used. With a 00 there, the III will assume we've got hardware handshaking and not worry about anything else. 80 tells the III we're using XON/XOFF, while 40 is the flag for ETX/ACK (or a variant known as ENQ/ACK).

Byte 06 holds the character we'll use as either XOFF, ETX or ENQ. It can be either 13. 03, or 05, respectively.

Byte 07 holds 11 for XON and 06 for ACK. (The character is the same for either the ENQ or ETX protocol.)

Byte 08 sets the maximum buffer level when using XON/XOFF. This ap-

Printer	Diagram	Notes
ANADEX DP-9500	5	Check parity, baud, stop bits, etc
ANADEX DP-9501	5	Check parity, baud, stop bits, etc
ANDERSON JACOBSON 832	A	
ANDERSON JACOBSON 833	A	
ANDERSON JACOBSON 841	A	Install jumper at location F
AXIOM EX-801	8	
AXIOM EX-820	8	
CENTRONICS 737	3	
COMPRINT 912	8	Check printer option jumpers
DIABLO 630	3	
EPSON MX-70	2	Use at 9600 baud
EPSON MX-80	2	Use at 9600 baud
HEATHKIT H-14	1	
IDS 125	7	
IDS 225	7	
IDS 440	7	
IDS 445	7	
IDS 460	7	
IDS 560	7	
NEC 5510	6	
NEC 5515	6	
NEC 5520	6	
NEC 5525	6	
QUME SPRINT 5	A	
TELETYPE 43	8	
TI 810	3	

1)

Apple III	Printer
1	1
2	3
6	4
7	7

2)

Apple III	Printer
1	1
2	3
6	11
7	7

3)

Apple III	Printer
1	1
2	3
4	6 & 8
6	11
7	7

4)

Apple III	Printer
1	1
2	3
6	14
7	7

5)

Apple III	Printer
1	1
2	3
6	19
7	7

6)

Apple III	Printer
1	1
2	3
4	5
6	19
7	7
20	6 & 8

7)

Apple III	Printer
2	3
6	20
7	7

8)

Apple III	Printer
1	1
2	3
6	20
7	7

A) Apple Supplied Cable

Apple III	Printer
1	1
7	7
2	3
3	2
4 & 5	8
8	4 & 5
6	20
20	6

Table 3. Printer connections for the Apple III serial port.

plies to the Apple III only and is necessary because the III can receive. (Your printer, if it uses this protocol, will already have its buffer set.) Since the III buffer can hold 255 characters maximum, set byte 08 at DF 233 characters, which will theoretically give the sending device time to hear the XOFF and stop before the buffer overflows. If you have a device that's a little hard of hearing, or slow to respond, set this value lower.

Byte 09 is the complement of XON/OFF, the minimum buffer level. As with the maximum, your printer will already be set (unless you've got it connected to another III). For the Apple set byte 09 at 84 (132 characters) before it sends an XON.

Byte 0A designates the block size for the data transmitted via the ETX/ACK protocol. Apple defaults to 50 (80 characters), but I'd try 80 (128 characters). If that proves too large, you can work it down.

The last byte, 0B, tells the III whether you're using a hardware handshake or not. 80 indicates you are using it (which means setting byte 05 to 00). A value of 00 in byte 0B means you are using one of the protocols.

When you first look at the .RS232 driver you'll find these values preset for modem use. With the exception of the baud rate, you might want to leave them like that. They do work.

To find out what values to specify for your printer, you'll have to take a trip through its manual; most manuals are very informative. I've included in Table 3 some of the Apple recommended cable arrangements for the III and for a variety of printers (purloined from an Apple III technical bulletin). In case you aren't the best detective in the world, and you can't figure out all the settings for a good driver routine, check Table 4 for a list of DCB values for various devices; it may also be of help.

PUNKY'S DILEMMA

Before I go any further, I've got to let you in on a letter I received from a man in Florida. He's had his III for seven months and is just getting around to really using it. Aside from explaining that my request for a

Device and Characteristics (baud,bits,parity,protocol)	Device Configuration Block Byte 00	01	02	03	04	05	06	07	08	09	0A	0B
Default values (300,7,odd,no protocol)	06	22	00	00	00	00	13	11	DF	84	50	00
Another Apple III (9600,7,odd,hdwr handshake)	0E	22	00	00	00	00	xx	xx	xx	xx	xx	80
Apple II (300,7,SPACE,no protocol)	06	2E	00	00	00	00	xx	xx	xx	xx	xx	00
DEC LA120 terminal (1200,7,SPACE,XON/XOFF)	08	2E	00	00	00	80	13	11	DF	84	xx	00
DEC VT100 terminal (9600,7,SPACE,XON/XOFF)	0E	2E	00	00	00	80	13	11	DF	84	xx	00
SOROC IQ120 terminal (9600,7,MARK,no protocol)	0E	2A	00	00	00	00	xx	xx	xx	xx	xx	00
Qume Sprint 5 printer (1200,7,odd,hdwr handshake)	08	22	00	00	00	00	xx	xx	DF	84	xx	80
Qume Sprint 5 printer (1200,7,odd,ETX/ACK)	08	22	00	00	00	40	03	06	xx	xx	6E	00
HP 7225 plotter (2400,7,SPACE,hdwr handshake)	0A	2E	00	00	00	00	xx	xx	DF	84	xx	80

All setups shown are typical local connections, and use a modem eliminator. Device configuration block parameters that have no effect under a given protocol are shown as "xx": these parameters can be left to set to any value.

Table 4. DCB values for commonly used devices.

SASE was difficult to fulfill since he had never found an envelope capable of addressing and stamping itself, he also described a problem with his dealer, and as an example related this little anecdote.

It seems that somewhere along the way my correspondent wanted to print something on his printer. When he asked the dealer how to go about it, he was told to type in PR#1.

That's not even laughable. Unfortunately, I get the feeling this response is typical of the support the III gets from a lot of the "superb" dealers in the network Apple maintains. And these are the same folk Apple is gambling its future on with Lisa. In fact, I'm going to risk placing not only my foot, but my entire leg in my mouth. After initial success because of its novelty, unless Lisa receives adequate dealer support, it, like the III, is not going to do half as well as it might. When you've got to pump iron out the door to meet payroll and the rest of the overhead, there's rarely time to learn driver routines, system configuration programs and the many other things that constitute real support for equipment. *Help* is needed.

I can see the hackles rising, but I stand by what I've said—even though I hope I'm wrong. Being three or four months from this writing, perhaps you know whether I'm beaming like a Cheshire cat or munching away on my tibia.

Enough of the soapbox at present. For any of you who have recently gotten your III and would like to know how to print with it, don't type PR#1. That's for the Apple II, or the III in *Emulation mode*.

Since all of the drivers can be "opened" (much the same as a file), printing is a simple chore. If you want to print just a line or two, all you have to do is this:

OPEN#1, "<whatever you call the printer driver>"
PRINT#1;"<whatever you want to print>"
CLOSE#1

It's most probable that your printer driver is called .PRINTER, but it doesn't have to be. If you don't know what it is, find out by using the SOS Utilities disk to load in the SOS.DRIVER file from the disk you're running. It might be called .PARALLEL or it might be named after the printer you're using, to set it apart.

The other way to use the printer would be to list out something like a program you're working on. If that is the case, type the following:

OPEN#1,"<whatever. . ., etc.>"
OUTPUT#1
LIST
CLOSE#1

The OUTPUT command redirects all output to the device you've opened, which, in this case, is the printer. (A filename and a drive would capture the program on disk instead of paper.)

The good part about doing things this way is the ability to manipulate output with greater simplicity than on other machines. You could slip any driver you wanted into the slot after OPEN#1 and route the output anywhere.

Now that you know how to modify a printer driver and how to use the printer (or any device connected to the Apple III's RS-232 port), I should mention problems that are inherent in using them. There aren't many, but those few that do crop up deserve attention.

First, the III's output on any of the RS-232 lines is 0 when the port isn't addressed. That means that if you use a modem (like the Hayes Smartmodem) that monitors DTR (pin 20), and you should happen to use a program (like Access III) that can open and close the driver depending on the options menu you select, the modem will disconnect you as soon as DTR is low. In the case of the Smartmodem, you can go inside and set a switch to ignore DTR. For any other modem you'll have to play with the cable a little to find a line that stays high. (My Smartmodem has the DTR-ignore switch activated; other than that I use a straight 25-connector cable.)

Also, some of the lines on the III, when pulled low, halt all output. In effect, they freeze the machine until they go high again. If, for instance, your printer ignored a few of the electrical signals on the RS-232C line (by keeping them low), and the line happened to be one that was being monitored, you'd probably end up with both the computer and the printer doing nothing, and you scratching your head while mumbling unintelligible praise of the system.

If you're going to do cabling yourself, you've got to take into account both the direction the signal travels (either to the III from the peripheral device, or from the III to the device) and the voltage levels that will actually make things happen. (See Table 5.)

Considering the amount of information I've crowded in here, you

Pin	Name	Description
1	GND	Protective frame ground
2	TxD	Transmit data (from the Apple)
3	RcD	Received data (to the Apple)
4	RTS	Request to send (from the Apple)
5	CTS	Clear to send (to the Apple)
6	DSR	Data set ready (to the Apple)
7	GND	Signal ground
8	DCD	Data carrier detect (to the Apple)
9–19		No connection
20	DTR	Data terminal ready (from the Apple)
21–25		No connection

Table 5. Apple III serial port signal descriptions.

may have a bit of rereading to do before you get everything down exactly right, but don't worry about it. With a moderate run-through you should have enough savvy to get over at least small hurdles.

If you run into large obstacles, there are other routes to take. One way is easy; go to your dealer and ask for help. After all, you did spend your money at *their* store, and Apple's policy calls for total Apple equipment support by the dealers Apple has selected. If the dealer tells you something right, thank them kindly. If they tell you something wrong, don't get upset and angry and hostile and all worked up. Above all, do not place foreign objects, such as your fist, through the monitor. Instead, write or call Apple Computer.

If time is of the essence and you *must* call, their hotline number for non-California folk is (800) 538-9696. Those in California can dial (800) 662-9238. As fas as phoneable solutions go, the results of these contacts aren't too bad.

All in all, it's better to document the problem you're having and the name of your dealer and mail it to Apple (of course, keeping a copy of the letter in your files for future reference).

But do be fair. If your dealer has been good to you, document that as well and send it off. Too often no one mentions the good dealers.

The other approach to problems you can't solve is to write to your local Apple magazine. More likely than not, someone else has experienced the same problem and may well have suggestions for you. Even if this fails, you will have at least alerted other owners to difficulties they might face given the same set of circumstances.

LOOSE ENDS

The other day I got a letter from a reader concerning files saved from Access III. When they're read with AppleWriter III, each line of text begins with an inverse J. This makes formatting difficult, if not impossible.

Here's how it happens. Access III stores all text with a leading linefeed character, CHR$(10); AppleWriter III displays control characters (those with an ASCII value of less than 32) as inverse characters. If we count up, starting with control-A as 1, we'll find that control-J is actually ASCII character 10. By the smallest of coincidences, that's the character that represents a linefeed.

Using the [F]IND option of AppleWriter III, enter a slash (/), then press down the control and J keys at the same time. You'll see an inverse J on the screen. Type in two more slashes and the letter A. That tells the AppleWriter to find all occurrences of control-J, remove them and replace them with nothing—and to do it (A)utomatically without asking you each time. After that, you can format to your heart's content.

I'd like to thank Dr. R., ST0823, of the Source Apple III User's Group for extending me a complimentary membership. I would reply in person, but there has been a lot of commotion around the ranch lately, and I can't find the magazine I stuck his letter in. (No—it was a car magazine, my second love.)

That's it then for this month. Live long and program.

Ciao bene, AppleAmerica! ■

-Friendly Service-

Business & Utilities

Accent Software
- Soft-Step $ 34
- Accu-Shapes $ 34
- Loop-Hole $ 17
- The Graphic Solution $ 95

Applied Software
- Versaform $239

Artsci
- Magic Window II $ 95

Ashton-Tate
- dBase II (Req. Z-80) $419

Avant-Garde
- Hi-Res Secrets $ 79
- Ultra Plot/Data Graph $ 62
- Graphics Application System $ 49

Beagle Brothers
- Apple Mechanic $ 20
- Tip Disk #1 $ 15
- Flex Text $ 20
- Frame Up $ 20
- Typefaces $ 15

Broderbund
- Payroll $249
- Accounts Receivable $249
- The Bank Street Writer $ 45

Continental Software
- The Home Accountant $ 48
- 1st Class Mail $ 48
- CPA Module #1 — G/L $159
- CPA Module #2 — A/R $159
- CPA Module #3 — A/P $159
- CPA Module #4 — Payroll $159
- CPA Module #5 — Property Mgt $159
- The Form Letter $ 20

Crane Software
- Menu Generator $ 27

Cybertronics
- Karel the Robot $159

Dakin 5
- The Depreciation Planner $249
- The Budget Planner $ 95
- The Business Bookkeeping System $249

Datamost
- R.E.I.P. $ 82
- Write-On $ 82
- Personal Ledger $ 82
- The Market Technician $ 82
- The Property Manager $189

Datasoft
- Micropainter $ 23
- Datasm 65 2.0 $ 59
- Lisp Interpreter $ 79
- The Basic Compiler $ 65

Data Transforms
- Graphtrix 1.3 $ 49

Delta Software
- The Bookkeeper Master Program $ 59
- The Bookkeeper Check Writer $ 27

Denver Software
- Pascal Tutor $ 82
- Pascal Programmer $189
- Financial Partner $159

DJR Associates
- FMS-80 (Req. Z-80) $545
- FMS-81 (Req. Z-80) $235
- FMS-82 (Req. Z-80) $235

Don't Ask Software
- S.A.M. $ 79

Eagle Software
- Money Decisions $129

Fox & Geller
- Quick Code (Req. Z-80) $189
- Dutil (Req. Z-80) $ 62

Hayden Software
- Pie Writer/Multi 80-col. $ 95
- Pie Writer/Standard $ 95

Hayes Microcomputer Products
- Hayes Terminal Program $ 65

Highlands Computer
- E-Z Ledger $ 39

Howard Software
- Creative Financing $129
- Real Estate Analyzer II $119
- Tax Preparer 1983 $145

Business & Utilities

Insoft
- Transforth II $ 79
- Ald System II $ 49
- Graforth II $ 49

ISM
- Mathemagic $ 59
- Graphmagic $ 59

IUS
- Datadex $ 99
- Professional Easywriter $115
- Tellstar Level I $ 27
- Tellstar Level II $ 52
- Forth Development System $ 89
- Professional Easymailer $ 99
- Original Easywriter $ 65
- Original Easymailer $ 45
- Easymover $ 62
- Pro. Easywriter/Mailer Combo $189
- Orig. Easywriter/Mailer Combo $ 89

Kensington Microware
- Format II $159

Link Systems
- Datafax $119
- Datalink $ 62
- Link Index $ 90
- Link Video $ 34
- Link Disk $ 45

LJK Enterprises
- Letter Perfect with Mail Merge $ 95
- Data Perfect $ 69
- Edit 6502 $ 69

Lotus
- Executive Briefing System $125

Micro Lab
- The Invoice Factory $129
- The Asset Manager $129
- The Tax Manager $ 95
- Data Factory 5.0 $189
- Wall Streeter $189
- Payroll Manager $189

Microsoft
- Tasc Compiler $119
- Fortran-80 (Req. Z-80) $139
- A.L.D.S. (Req. Z-80) $ 79
- Basic Compiler (Req. Z-80) $249
- Cobol-80 (Req. Z-80) $459
- muLisp/muStar-80 (Req. Z-80) $125
- M-Sort-80 (Req. Z-80) $125
- muMath/muSimp-80 (Req. Z-80) $159
- Time Manager $ 95
- Multiplan (Apple Dos or Z-80) $175

Micro-Sparc
- Ampersoft $ 34
- Directory Master $ 20

Business & Utilities

Systems Plus
- Accounting Plus — G/L $239
- Accounting Plus (GL, AP, AR) $499
- Accounting Plus (GL, AP, AR, INV) $629

Muse
- Super Text 40/80 $115
- Super Text 40/56/70 (Hi-Res) $ 79

Novation
- Deaf Disk $ 23
- Software Diskette $ 23

On-Line
- Screenwriter II $ 82
- The Dictionary $ 65
- Lisa 2.5 $ 55
- Lisa Educational System $ 75
- The Artist $ 55
- Screenwriter Professional $129
- The General Manager II $149

PBL Corporation
- Personal Investor $ 95

Peachtree
(All Peachtree products require a Z-80 card)
- Series 40 Acc't Packages ea. $239
- Series 80 Acc't Packages ea. $239
- Peachtext $289
- Spelling Proofreader $179
- Mailing List Manager $209

Penguin Software
- Complete Graphics System $ 49
- Special Effects $ 27
- Graphics Magician $ 39
- Additional Fonts & Char. Sets $ 15
- Complete Graphics System II (Apple Tablet Version) $ 79
- Special Effects (Apple Tablet Version) $ 49

Select
- Selectword Processing Sys. $299

Sensible Software
- Sensible Speller $ 79
- Image Printer $ 27
- Super Disk Copy III $ 20
- Multi Disk Catalog III $ 17
- Disk Recovery $ 20
- Quickloader $ 17
- The Bug $ 34

Silicon Valley
- Word Handler II $129
- List Handler $ 59
- Dictionary $ 79

Stoneware Products
- D.B. Master $145
- D.B. Master Utility Pak #1 $ 65
- D.B. Master Utility Pak #2 $ 65
- Graphics Processing Sys. (Standard) $ 49
- Graphics Processing Sys. (Prof.) $119

Synergistic
- The Data Reporter $159
- Integer Basic Compiler $ 95
- Global Program Line Editor $ 45
- The Inventory Manager $ 95

Software Publishing
- PFS: Report $ 85
- PFS $ 85
- PFS: Graph $ 85

Sorcim/ISA
- Supercalc (Req. Z-80) $179
- Spellguard (Req. Z-80) $125

Southeastern Software
- Data Capture 4.0 $ 47
- Data Capture 4.0/80 $ 59

Southwestern Data (New)
- Apple Doc $ 27
- Ace $ 27
- List Master $ 27
- Speed Star $ 65
- Correspondent $ 39
- Z Term (Req. Z-80) $ 65
- Z Term "The Professional" (Req. Z-80) $ 99
- P Term "The Professional" $ 85
- Ascii Express "The Professional" $ 85
- Disk Library $ 34
- Double Time Printer $ 65
- Financial Management System II $ 45
- Munch-a-Bug $ 34
- Printerographer $ 34
- Routine Machine $ 45

Visicorp
- Visicalc 3.3 $165
- Visiplot $139
- Visitrend/Plot $199
- Visidex $165

Education

- Globe Master $27
- Gertrude Secrets $49
- Gertrude Puzzles $49
- Rocky's Boots $49
- Snooper Troops #1 & #2 Each $30
- Juggles Rainbow $30
- Bumble Games $39
- Rhymes & Riddles $20
- Story Maker $26
- Face Maker $26
- Delta Drawing $45
- Compu-Read $23
- Spelling Bee w/Reading Primer $27
- Fractions $34
- Decimals $34
- Master Type $27
- Type Attack $27
- Wordrace $17
- Dueling Digits $20
- New Step by Step $59
- Spanish for Traveler $39
- Whole Brain Spelling $23
- Solar System $27
- Speed Read Plus $39
- PSAT Word Attack Skills $34
- S.A.T. Word Attack $34
- Harcourt Brace S.A.T. Series $59
- Color Me $20
- Anatomy $27
- U.S. Constitution Tour $20
- Algebra I $34
- Algebra II $20
- Algebra III $27
- The Linguist $ 27
- Know Your Apple $ 23
- E-Z Learner $ 24
- Crossword Magic $ 34
- Counting Bee $ 20
- Compu-Math/Arith. Skills $ 34
- Planetary Guide $ 23
- Star Gazers Guide $ 22
- Astro Quotes $ 17

Computer Outlet

Park Place — Upper Level
1095 E. Twain — (702) 796-0296
Las Vegas, Nevada 89109

Call Toll Free **800-634-6766** Order Line Only

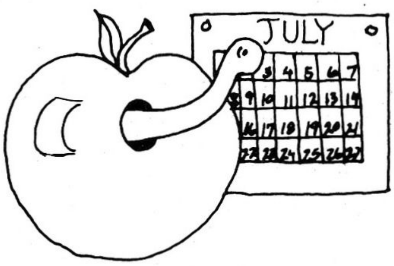
Information Order Inquiries (702) 369-5523

We accept Major Credit Cards

Mon.-Fri. 8 A.M.-6 P.M.
Sat. 9 A.M.-5 P.M.
Dealer Inquiries Invited

Circle 254 on Reader Service card.

Calendar



June 27–July 29
International Institute on Microcomputers in Education
Stanford, CA
contact:
International Institute on
Microcomputers in Education
216 School of Education
Stanford University
Stanford, CA 94305
(415) 497-2102

July 10–11
The Role of Microcomputers in Music Education
River Grove, IL
contact:
Pearl Anshell
1226 North Rossell Ave.
Oak Park, IL 60302
(312) 386-7733

July 11–15, July 18–22
Microcomputers in Education Summer Institutes in Math, Science and Computer Literacy
Trinity College, VT
contact:
Technical Education Research
Centers (TERC)
8 Eliot St.
Cambridge, MA 02138
(617) 547-3890

July 13–15
The American Production and Inventory Control Society, 1983 Summer Seminar
Seattle, WA
contact:
APICS Society Headquarters
500 West Annandale Road
Falls Church, VA 22046
(703) 237-8344

July 14–17
Origins '83—The 9th Annual International Gaming Show
Detroit, MI
contact:
Metro Detroit Gamers
083 Info
PO Box 656
Wyandotte, MI 48192

July 25–28
SoftFair—Software Development, Tools, Techniques and Alternatives
Arlington, VA
contact:
SoftFair
PO Box 639
Silver Spring, MD 20901
(301) 589-3386

July 25–29
Siggraph '83—10th Annual Conference on Computer Graphics and Interactive Technique
Detroit, MI
contact:
Siggraph '83 Conference Office
111 East Wacker Drive
Chicago, IL 60601
(312) 644-6610

August 1–5
LOGON '83—Technology, Mythology and Literacy in the New Age
Montana State University, Bozeman
contact:
Michael Sexson
English Dept.
Montana State University
Bozeman, MT 59717
(406) 994-3768

August 13
Solar Design II: Computer-Aided Solar Design
Falmouth, Cape Cod, MA
contact:
Merryl Alber
New Alchemy Institute
237 Hatchville Road
East Falmouth, MA 02536
(617) 563-2655

August 22–26
The National Conference on Artificial Intelligence
Washington, DC
contact:
American Association for
Artificial Intelligence
445 Burgess Drive
Menlo Park, CA 94025

August 30–September 2
Second International Conference on Databases
Cambridge, England
contact:
H. D. Rose
The British Computer Society
13 Mansfield St.
London W1M OBP, England
phone: 01-6370471

September 12–14
1983 IEEE International Conference on Computer-Aided Design
Santa Clara, CA
contact:
Dr. Wilham J. McCalla
Hewlett-Packard Co.
11000 Wolfe Road
Cupertino, CA 95014

September 14–16
EUROMICRO 1983 Symposium
Madrid, Spain
contact:
EUROMICRO
TH TWENTE
PO Box 217
Dept. INF room A312
7500 AE Enschede, The
Netherlands
phone: (31) (53) 338799

September 19–23
9th World Computer Congress—IFIP '83
Paris, France
contact:
IFIP '83
4 et 6, Place de Valois
F-75001 Paris, France

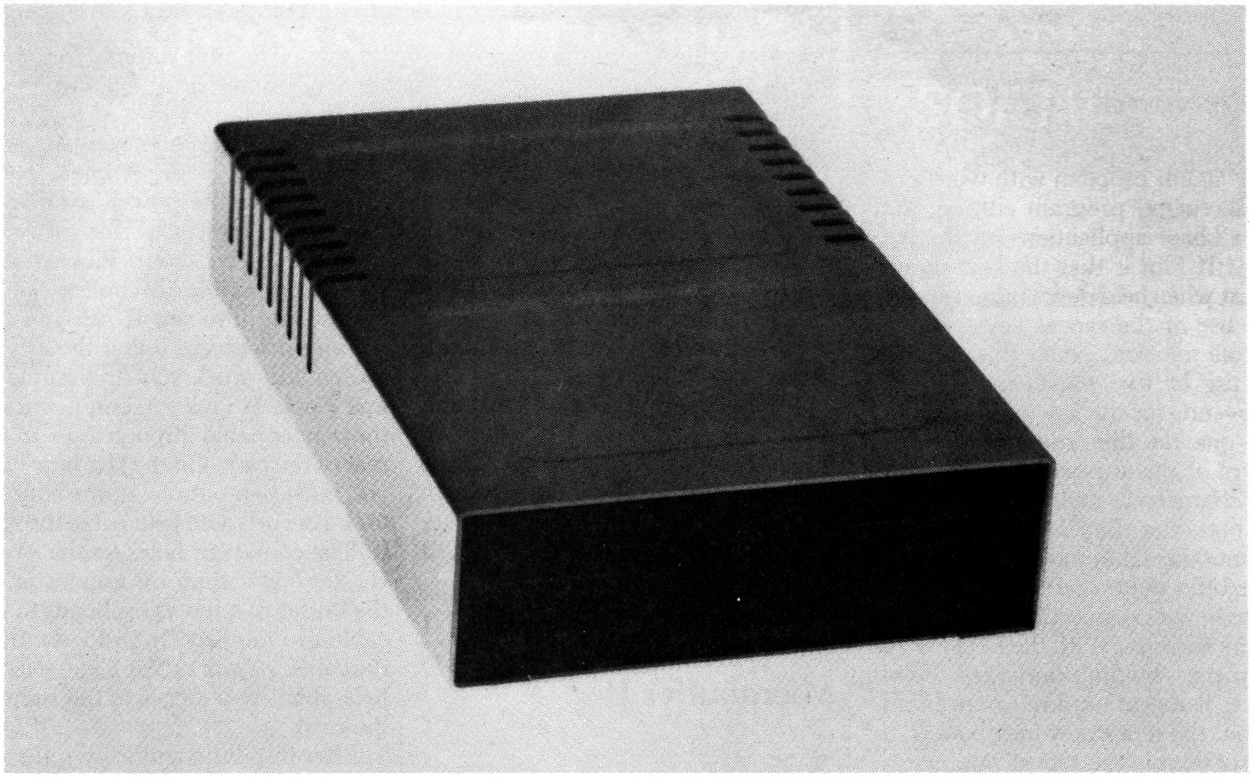
September 22–25
1983 Computer Showcase Expo
New York, NY
contact:
Lynn Burdett
Interface Group Inc.
300 First Ave.
Needham, MA 02194
(617) 449-6600, ext. 288

*If you are organizing, or otherwise know of, an event important to Apple users, and would like it listed in the **inCider Calendar**, please drop us a line at **Pine Street, Peterborough, NH 03458**. Include the name of the event, the date, the location, and the name of a contact for further information.*

—the editors

RGB designs

MINI TAUR II DISK DRIVE



HARDWARE AND SOFTWARE COMPATIBLE WITH

Apple II*, Apple II+*, and Apple 2e*

Franklin ACE* 1000 and 1200

Basis 108*

FEATURES

100% Compatible With ALL Apple Software

Half Track and Full Track Accessing Capability

35 Track and 40 Track Operation

Band Stepper Provides Faster and More Precise Head Positioning

Twice the Storage in the SAME Space as a Standard Apple Drive

User Selectible Write Protection Switch Featuring Three Modes

- Normal Write Protect Operation
- Read Only Always Write Protected
- Write Protect Disable Allows Use of Opposite Side of Diskette

Color Coordinated to Apple Computers

90 Day Warranty   1 Year Warranty Optional

For Dealer and Distributor Information Call:

RGB designs

3375 Woodward Avenue
Santa Clara, CA 95050
408/748-0400

*Apple, Apple II, Apple II+, and Apple 2e are registered trademarks of Apple Computer, Inc. Franklin Ace is a trademark of Franklin Computer Corporation. Basis is a trademark of Basis Incorporated.

Circle 217 on Reader Service card.

Hardware Reviews

Repeaterrrr

A major problem with word processing, program editing, and spreadsheet applications on the Apple II/II Plus is that the keys do not repeat when held down (auto-repeat). The use of the repeat key is only a partial solution, since this requires the use of two hands most of the time—one for the key to be repeated and one for the repeat key. Rapid moves of the cursor or repetition of any character is awkward.

Enter the Repeaterrrr—a small circuit board that mounts on the keyboard encoder board connector, with a single wire connection to a resistor on the encoder. With the Repeaterrrr in place, holding any key down makes it repeat until released. In addition, the repeat key may be used to nearly double the repeat rate.

Installation is quite simple and does not require removing the base plate of the Apple. A tool is supplied to release the retainer clips on the encoder board, and you then carefully remove the board from the connector. The Repeaterrrr is slipped onto the 25-pin connector, and the jumper wire connector is placed on the appropriate resistor wire (no soldering is needed). If you want to make the shift key modification, a connection point is supplied on the Repeaterrrr board. The encoder board is then put back in place and you're ready to go. Total installation time is about 15 minutes.

There is a potentiometer on the Repeaterrrr board to adjust the time delay before repeating when a key is held down. It can be set from .3 seconds to over one second. This allows the user to tailor the responsiveness of the feature to his/her own tastes. The Repeaterrrr can also be turned off, for use with those games that require keys to be held down, or when the auto-repeat function is not desired. Access to the adjustment only requires the top of the Apple to be removed. The Repeaterrrr is quite accessible when installed. A Basic program is supplied in the documentation that will tell you the exact delay setting.

I have been using the Repeaterrrr for two months without problems. The auto-repeat is fast, but not so fast that you can't stop where you want to with reliability. I have rarely needed to use the repeat key (renamed by the manufacturer the ZippKey), but when you *really* want to move through a text file or spreadsheet, you can.

The Repeaterrrr is only $24.95 and it comes with a 30-day free trial period and a one-year warranty. The manufacturer is High Order Micro Electronics Corporation, 17 River Street, Chagrin Falls, Ohio 44022. ■

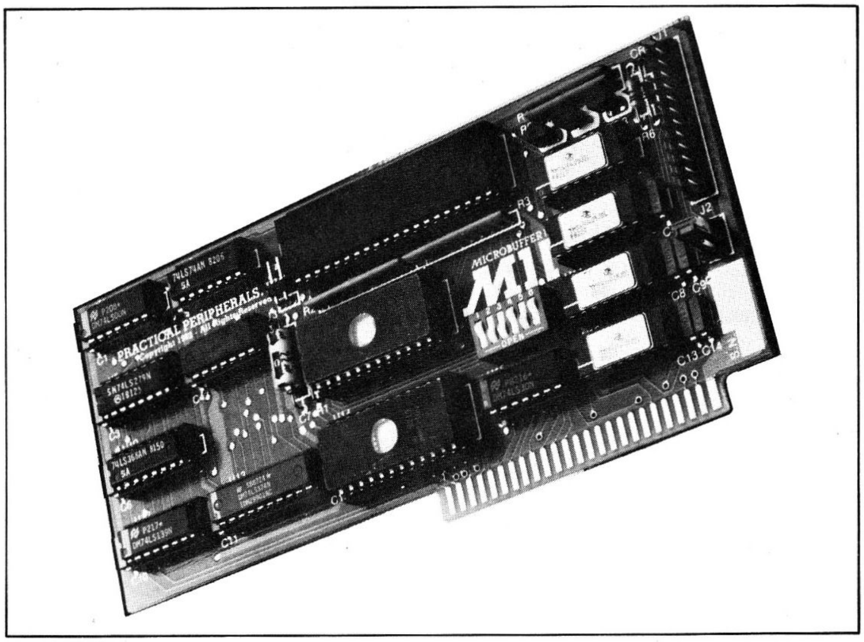
Raney Ellis
Plattsburgh, NY

Microbuffer II

When I received the Microbuffer II for review, I had planned to spend a good part of the night installing the machine and setting up the cables to my Centronics printer. To my surprise the hardest part of installing the Microbuffer was opening the box in which it was shipped. The unit comes set up for a specific printer and includes a customized cable.

To install the interface you need only remove your Apple's cover (making sure the power is off) and select the interface slot of your choice. To be both Pascal- and CP/M-compatible the buffer should be installed into slot 1, but you can use any slot except 0 and the Microbuffer will work. On the Apple II and Apple II Plus you can just route the printer cable through a slot in the rear of the case. On the IIe, however, you must pop out one of the connector holes and route your cable through it. The connector holes on the Apple IIe are lower than the connector on the buffer so a few extra bends in the cable are required to properly align the cable. (It works best if you select a hole about two inches to the right of the card.)

After installing a new peripheral I usually just turn on the power and watch for smoke. In this case, however, I did not own the test unit so I decided to check the manual. As it turned out, the manual offered the perfect command for checking out my installation: The Microbuffer II has a verify command that examines



The Microbuffer II from Practical Peripherals.

EASTCOAST SOFTWARE

ecs... inflation-fighting prices.

APPLE SOFTWARE

HARDWARE

HARDWARE	LIST	SALE
8080 Co-Processor Board	$845.00	$645.00
ABT Keypad	125.00	98.00
Adam & Eve Paddles	39.95	27.50
Amdek 12" Green Mntr	179.00	148.75
Amdek Color I	399.00	336.25
Apple Cat II Modem	389.00	282.15
Applicard (6MHZ)	595.00	460.00
BAM 16 K Card	129.00	75.00
Bufferboard-16K	175.00	135.00
Bufferboard-64K	259.00	175.50
Bufferboard-16K Upgrade	28.00	12.45
Corona 5MB Winchester	2195.00	1975.00
Corona 10MB Winchester	2695.00	2425.00
Echo 2 Speech Synthesizer	199.95	175.85
Graffiti Card	125.00	95.25
Grappler +	165.00	128.00
Joyport	49.95	35.50
Kraft Joystick	64.95	44.90
Kraft Paddles	49.95	34.50
Micro-Sci A2 Disk Drive	345.00	295.25
Micro-Sci Controller Card	100.00	81.70
Microbuffer II-16K	259.00	199.00
Microbuffer II-32K	299.00	230.00
Microbuffer In-Line-64K	349.00	292.50
Micromodem II	379.00	279.65
Micromodem II W/Term	**409.00**	**295.00**
Microtek Apple Dumpling-GX	159.00	105.50
Microtek Dumpling-16K	249.00	169.75
Microtek Dumpling-32K	299.00	207.35
Microtek Parallel Interface	139.50	62.85
NEC 1201 12" Green Mntr	285.00	159.00
NEC 1260 12" Green Mntr	149.95	125.00
NEC 8023 Printer	645.00	475.00
NEC JC1212 Color Mntr	399.00	310.00
Pkaso Epson 80/100	175.00	145.00
Pkaso NEC 8023	175.00	145.00
Prowriter 8510 AP Printer	595.00	500.00
Prowriter 2 1550-P Printer	995.00	750.00
Ramcard (Microsoft)	**99.95**	**75.00**
Ramex 128 Board	499.00	406.25
Rana Controller Card	135.00	100.55
Rana Elite 1 Disk Drive	379.00	306.25
S.A.M.	124.95	86.35
Sanyo 12" Green Mntr	260.00	198.75
Sanyo 13" Color Mntr	470.00	390.00
Sanyo 9" Green Mntr	200.00	155.00
Saturn 128K RAM Board	499.00	436.25
Saturn 32K RAM Board	219.00	191.25
Saturn 64K RAM Board	349.00	305.00
Smarterm Standard	345.00	281.45
Softcard (Microsoft)	345.00	232.45
Softcard Premium System	695.00	500.00
SSM Apple Modem Card	299.00	263.50
SSM Transpak-2	448.00	377.50
Starwriter F10 Printer	1895.00	1562.50
Sup'r Mod Universal	69.95	55.30
Superterm 80-Column Card	349.95	260.00
System Saver Fan	**89.95**	**72.25**
Taxan 12" Green Mntr	169.00	124.40
Taxan 12" Amber Mntr	179.00	136.95
Tech Designs Joystick	59.95	41.25
TG Game Paddles	39.95	27.50
TG Joystick	**59.95**	**41.40**
TG Joystick-Apple 3	64.95	47.00
TG Select-A-Port	59.95	41.40
TG Trakball	64.95	46.50
Trak 510 Disk Drive	329.00	275.75
Tymac Parallel Interface	139.00	75.40
USI PI-2 12" Green Mntr	210.00	154.55
USI PI-3 12" Amber Mntr	249.00	162.00
USI PI-4 9" Amber Mntr	215.00	154.55
Versacard	199.00	157.00
Videx Enhancer II	149.00	110.00
Videx Function Strip	79.00	60.00
Videx Softswitch	35.00	27.25
Videx Videoterm	345.00	239.00
Videx Videoterm W/Softswitch	**375.00**	**260.00**
Wildcard	129.95	110.00
Wizard 16K RAM Card	95.00	62.85
Wizard 80-Column Card	249.00	169.75

HOME/EDUCATION

HOME/EDUCATION	LIST	SALE
Algebra I	$39.95	$27.60
Algebra II	39.95	27.60
Algebra III	39.95	27.60
Bumble Games	60.00	41.50
Bumble Plot	60.00	41.50
Compu-Math Arithmetic Skills	49.95	34.50
Compu-Spell System	29.95	20.70
Delta Drawing	59.95	43.00
Facemaker	34.95	24.00
Gertrude's Secrets	75.00	51.85
Gertrude's Puzzles	75.00	51.85
Home Accountant	**74.95**	**48.05**
Juggles Rainbow	45.00	31.10
Kindercomp	29.95	20.70
Know Your Apple	34.95	24.15
Master Type	39.95	27.60
Preschool IQ Builder	23.95	17.00
Rocky's Boots	75.00	52.50
SAT Word Attack	49.00	33.90
Snooper Troops #2	44.95	32.50
Snooper Troops #1	44.95	32.50
Spelling Bee	29.95	20.70
Story Machine	34.95	24.00
The Menu II	**39.95**	**28.90**
Typing Tutor II	24.95	17.55

WORD PROCESSING

WORD PROCESSING	LIST	SALE
1st Class Mail	$99.95	$64.10
Apple Writer II Pre-Boot	**19.00**	**13.50**
Bank Street Writer	69.95	48.35
Dictionary (On-Line)	99.95	69.00
Easy Writer (Pro)	175.00	121.00
Format II	250.00	170.00
Letter Perfect	149.95	103.75
Magic Window II	150.00	103.60
Magic Window (Original)	99.95	69.00
Pie Writer	149.95	109.25
Screenwriter II	**129.95**	**89.00**
Screenwriter Pro	199.95	138.25
Sensible Speller (New)	125.00	86.50
Supertext 40/80	175.00	120.95
Supertext Form Letter	100.00	69.00
Supertext Address Book	49.95	34.50
Supertext 40/56/70	125.00	86.50
Word Handler	199.00	137.55
Wordstar	**495.00**	**342.00**

GRAPHICS

GRAPHICS	LIST	SALE
Alpha Plot	$39.50	$27.25
Apple Mechanic	**29.50**	**20.35**
Complete Graphics Sys	69.95	48.35
E.Z. Draw 3.3	49.95	34.50
Frame Up	29.50	20.35
Graphics Magician	59.95	41.40
Hi-Res Secrets	125.00	86.35
Higher Text Extended	**22.50**	**16.50**
Special Effects	39.95	27.60
The "Slide" Show	**49.95**	**35.50**
The Artist	79.95	55.25
Type Faces	20.00	13.85
Zoom Grafix	49.95	34.50

UTILITIES

UTILITIES	LIST	SALE
ASCII Express-Pro	$129.95	$94.00
Bag of Tricks	**39.95**	**27.60**
Data Capture 4.0	64.95	44.85
DOS Boss	24.00	16.60
Flex Text	29.50	20.35
Global Program Line Editor	65.00	47.00
Locksmith 4.1	**99.95**	**72.00**
Multidisk Catalog 3	25.00	17.25
Nibbles Away II	**69.95**	**53.25**
Pronto DOS	29.50	20.35
Super Disk Copy III	34.95	24.65
Tasc Compiler	175.00	121.50
Tip Disk #1	20.00	13.85
Utility City	29.50	20.35
Videx Videoterm Utility Disk	37.00	25.45
Videx Visicalc Pre-Boot	**49.00**	**34.00**
Videx Visicalc Memory Exp.	89.00	73.00

GAMES

GAMES	LIST	SALE
AE	$29.95	$20.70
Ali Babba & The 40 Thieves	32.95	22.80
Arcade Machine	59.95	41.40
Aztec	39.95	27.60
Beagle Bag	29.50	20.35
Cannonball Blitz	34.95	24.00
Canyon Climber	29.95	20.95
Castle Wolfenstein	29.95	20.70
Choplifter (Special)	**34.95**	**23.55**
Cosmic Balance	39.95	28.90
Crisis Mountain	34.95	24.00
Crush, Crumble & Chomp	29.95	20.70
Deadline	49.95	34.50
Empire I, II or III	39.95	22.75
Flight Simulator	33.50	26.00
Frogger (Special)	**34.95**	**23.55**
Jawbreaker	29.95	20.70
Knight of Diamonds	34.95	24.00
Legacy of Llylgamyn	**39.95**	**28.90**
Mask of the Sun	39.95	27.60
Miner 2049er	39.95	28.90
Olympic Decathalon	29.95	20.70
Pinball Construction Set	39.95	27.60
Seafox	29.95	20.70
Spy's Demise	29.95	20.70
Starcross	39.95	27.60
Temple of Apshai	39.95	27.60
Thunderbombs	29.95	20.70
Time Zone	99.95	69.00
Ultima II	59.95	41.40
Ulysess & Golden Fleece	34.95	24.00
Upper Reaches of Apshai	19.95	13.75
Wizard & The Princess	32.95	22.75
Wizardry	49.95	35.50
Zaxxon	**39.95**	**27.60**
Zork I, II or III	39.95	27.60

SUPPLIES

SUPPLIES	LIST	SALE
Dysan Disks	$60.00	$41.50
Elephant Disks (SS/DD)	**37.00**	**25.00**
Elephant Disks (SS/SD)	34.00	21.90
Epson MX-100 Ribbon	25.00	16.65
Epson MX-80 Ribbons (2)	28.00	12.55
Flip-N-File	**29.95**	**18.15**
Head Cleaning Kit-Perfect Data	29.95	19.20
Maxell MD1 Disks	51.90	30.00
NEC 8023 Ribbons ¢(2)	19.95	17.10
Printer Paper-20# Perf		32.20 (+ Shipping)
Smith Corona TP1 Film Ribbon	7.95	6.35
Starwriter Film Ribbon	5.50	4.50
Starwriter Nylon Ribbon	5.75	4.65
Trunk Floppy Disk Storage	29.95	21.50
Verbatim MD525 Disks	**49.00**	**28.90**

BUSINESS

BUSINESS	LIST	SALE
Accounting + Gen. Ledger	$395.00	$270.50
Accts. Payable-Continental	250.00	160.25
Accts. Receivable-Continental	250.00	160.25
Business Forecast Model	100.00	79.65
DB Master	**229.95**	**158.70**
DB Master Stat Pac	99.00	69.50
DB Master Utilities	99.00	69.50
Desktop Plan	250.00	179.00
General Ledger-Continental	**250.00**	**160.25**
General Manager II	229.95	158.90
Multiplan	275.00	190.50
Payroll-Continental	250.00	160.25
PFS File II or 2E	125.00	85.95
PFS Graph II or 2E	125.00	85.95
PFS Report	125.00	85.95
PFS Report 2E	125.00	85.95
Prop. Management-Continental	495.00	317.35
Versaform	389.00	263.00
Visicalc II or 2E (Special)	**250.00**	**172.00**
Visidex	250.00	179.00
Visifile	250.00	179.00
Visischedule	300.00	215.00
Visitrend/Plot	300.00	215.00

Apple® is a registered trademark of APPLE COMPUTER INC.

EASTCOAST SOFTWARE

VISA®

110 West Caracas Avenue
Hershey, PA 17033

master charge
THE INTERBANK CARD

MAIL YOUR ORDER OR CALL **(717) 533-8125**

Shipping 1% ($2.50 min.)
C.O.D. Add An Additional $2.50
Foreign FPO-APO Orders We Ship
Air Mail...Add 7% (U.S. Currency Only)
No Minimum Order
Visa-Mastercard-C.O.D.-Check
Prices Subject To Change

Circle 362 on Reader Service card.

```
]LIST 60,90
60  PRINT
      : PRINT "APPLE II PLUS OR ROMCARD
       SYSTEM MASTER"
70  REM
80  REM  --POKE LANGUAGE CARD FIN
     DER
90  POKE 768,0
      : POKE 769,173
      : POKE 770,0
      : POKE 771,224
      : POKE 772,72
      : POKE 773,173
      : POKE 774,129
      : POKE 775,192
      : POKE 776,104
      :
```

Figure 1.

the internal ROM software and prints a page of test data on the printer.

When I had demonstrated to myself that the unit worked in my configuration I decided to return to the command section of the manual and experiment with some of the other commands.

The first thing I noticed was that the Microbuffer II supported the Apple II Parallel Printer Card commands, making it compatible with all my software. It uses control-I—just like the Apple card—as a signal to the interface software that the data which follows is for interface configuration and not to be passed to the printer. It supports the K command for automatic linefeed control and the N for setting line length and turning off the Apple video. It also allows you to change the control-I character to another control character.

The Microbuffer also supports approximately 20 commands not found on the Apple interface. These new commands include: the ability to control the printer's bell, margin and page length control, Basic formatting, high order bit enable/disable, screen dump of both text and graphics, a transparent mode, a maintain mode, a buffer zap command and the verify mode mentioned above.

The formatting controls allow you to produce nicely paged printouts of Basic or text data. The L command allows you to set the left margin to any value from 0 to 255. The N command sets the line length to any value from 0 to 255. By using the left margin and line length command you are also effectively setting up a right margin. A third command (P) sets page length value. The Microbuffer will automatically print six blank lines more than the page length value. This can be used to set top and bottom margins or just to skip over the paper perforation.

The unit has one very useful (but paper-wasting) mode called the Basic Listing Formatter. When this mode is enabled it will break multiple command Basic lines into separate lines. (See Figure 1 to see an example of a formatted listing.) This mode is great for checking Basic listings for errors even though it uses a lot of paper.

The Microbuffer supports a command that sounded like an answer to my Pascal printer problems. This command, when set up and sent to the Microbuffer, maintains the current state of the Microbuffer until either a reset occurs or the command is deactivated. With this command I could set up my desired margins and turn off the auto-linefeed while in Basic, then boot up Pascal using a PR#6 and still have my margins set and the auto-linefeed turned off. This is what I thought anyway. I was not able to make this command operate in this manner, though. Every time I booted up Pascal, it had forgotten everything I had set. I do not believe there was a problem with the Microbuffer; I believe it was an error in how I was setting up the interface. It also could be that the SYSTEM. STARTUP program I use to set my keyboard and display configuration may interfere with the Microbuffer setup. I did not try this command before booting CP/M, but it should be capable of performing a similar task if entering CP/M from Basic after setting up the Microbuffer and issuing a maintain command.

Another useful function is the transparent mode. When in this mode the Microbuffer allows all these funny control characters (which some printers require to command them into different print modes) to be passed through the interface instead of being interpreted as commands to the Microbuffer. This is useful in graphics modes where any bit pattern may be passed through the interface to generate a graphics printout. It is also useful for word processor outputs where special control characters are required to command the printer to generate nonstandard operations such as subscripts, superscripts and underlining.

Graphics

The Microbuffer also supports a graphics capability, which is invoked by the G command. When this command is invoked a subset of seven new graphics commands is allowed. These commands allow you to print either hi-res page, mixed hi-res and text, double size images, 90-degree rotated images, or inverse images. The printer I own does not have a graphics mode, but a friend tested the graphics modes with an Epson printer. Figure 2 shows an example of a hi-res graphics page printed in the inverse mode. Figure 3 shows the same picture but printed in the emphasized mode and rotated 90 degrees.

I tried the Microbuffer with both Apple Pascal 1.1 and CP/M. In both cases the operating systems recognized the printer as being connected to the system and I was able to print to the printer with no special fixes. The Microbuffer designers utilized the same trick the 80-column video board designers use to fool operating

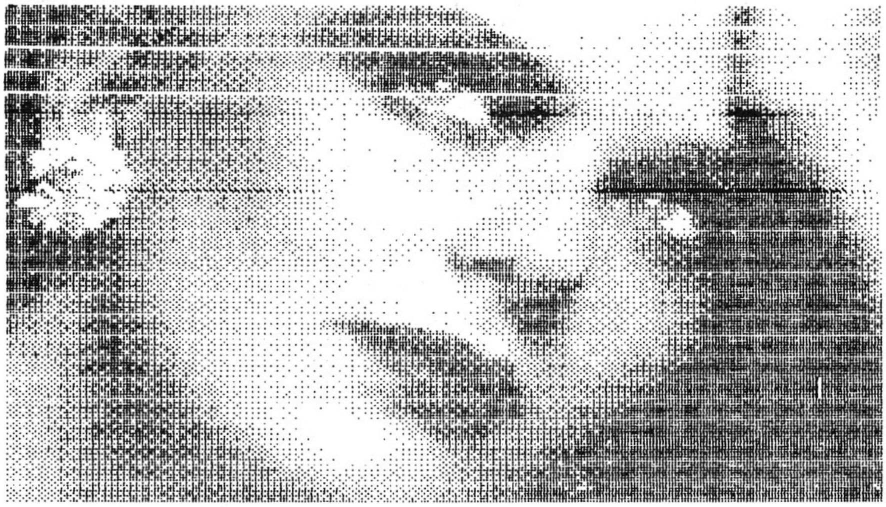


Figure 2.

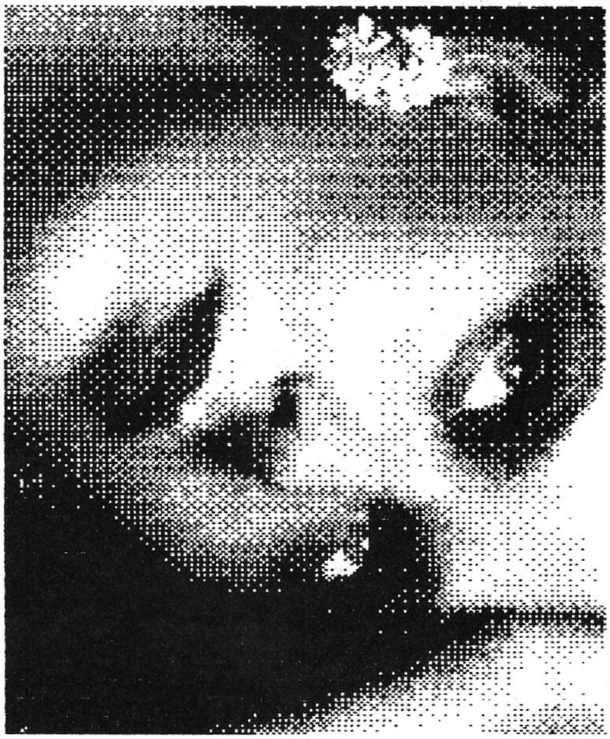


Figure 3.

systems: They make the software driver in the interface appear to be an Apple serial card.

Well, it has taken me long enough but I am finally getting to the most important feature of the Microbuffer—the buffer. The buffer acts as a storage point for data as it is being sent from the program to your printer. It is not the Apple or the interface that causes your printouts to take forever, but the printer. With the Microbuffer the Apple can dump a full 32K characters (16K on the lower priced version) into the Microbuffer as fast as the Apple can move the data, and the Microbuffer will stack up the data in the buffer while the printer plods along. This frees up the Apple for other tasks and, according to the Microbuffer manual, for a typical hobbyist printer, this could result in a time saving of nine minutes. (Of course if your listing is longer than 32K characters then the Microbuffer's operation will resemble a typical printer interface when the buffer becomes full.)

Conclusion

The Microbuffer II tested out well in the Apple IIe system. The cost of a 16K Microbuffer is $259 and the 32K Microbuffer is $299. I believe these prices to be in line with the product you are receiving. Yes, you can probably purchase a parallel interface card for $160 that will enable you to print text and graphics, but it will not provide data buffering.

If you do a lot of printing the Microbuffer will allow you to unchain your computer from your printer and get on to other projects. ■

Oliver Holt
Amherst, NH

Appli-Card

Imagine for a moment the power of a 4 or 6 megaHertz Z-80 computer board with 64K of high speed RAM, 2K to 8K of ROM, a parallel port, port decoding, an expansion interface on board and CP/M capability. Sounds like a nice system to work with, doesn't it? Think how powerful this computer board would be if it could use another processor for I/O processing, including console printer and disk interfacing. What if the processor was part of a low cost computer with low cost peripherals and lots of hardware and software? Sound like a fantasy or dream?

Dream no more! The Appli-Card, a Z-80 computer board for the Apple II computer, is already here. This single board computer drops into just about any slot of the Apple II and turns it into a CP/M system with almost 64K of RAM. The board is a marvel of power and speed. It makes many S100 systems seem slow in comparison.

The Appli-Card is a complete Z-80 computer on a card. If you were to

change the on-board ROM, add a 5-volt power supply, an RS-232 interface and a disk drive, you would have a full Z-80 computer system. However, you would have to interface the whole thing with expensive peripherals. Personal Computer Products, the maker of Appli-Card, has done this for you by using the Apple II as the I/O processor for their Z-80 computer card. The advantages are simple—the Appli-Card's Z-80 runs CP/M and the user's application program at their full speed, while the Apple II's 6502 handles all of the keyboard, console, disk and peripheral I/O. This scheme allows both processors to run at their full rated speeds because they don't have the same data and address bus or memory space. The two systems communicate via a single parallel port, through which requests and data are passed from one to the other.

> "The advantages are simple—the Appli-Card's Z-80 runs CP/M and the user's application program at their full speed."

The Appli-Card itself is a high quality PC board that can plug into any slot of the Apple II (except slot 0). The board needs only +5 volts from the Apple; but it has the circuitry for decoding the port address from the Apple bus so that the card and the Apple can communicate.

The board also has an expansion interface. Two different extender boards, 64K and 128K RAM versions, have been designed to plug into the Appli-Card's expansion interface. They use special software and the Appli-Card's existing 64K RAM to give the user a RAM disk of either 128K or 192K for Apple DOS, or a RAM disk of 64K or 128K for CP/M. The use of either RAM extender board allows the Appli-Card to run CP/M Plus and its bank select memory.

The Appli-Card is a great improvement over other Z-80 cards because it runs at two or three times the clock speed of some cards and because the Apple's 6502 handles I/O work. I have the 6 MHz version of the Appli-Card and it is over three times faster than other Z-80 cards that use the Apple's Q3 clock signal to run at about 2 MHz.

Using WordStar has become a pleasure instead of a chore. The WordStar program itself is great, but since it is a disk based system, everything comes to a halt when it has to go to the disk. Disk access is fairly slow because the Z-80 and the 6502 are sharing the workload of reading the disk; the 6502 actually reads the disk and the Z-80 fills the WordStar text buffer. Other Z-80 cards have to share the same data and address buses (very slow), but the Appli-Card allows the two processors to run independently.

While the hardware may be the guts of any system, the software is the brain. And this system has quite a brain! The software support for the Appli-Card is impressive, and the utilities included make using it easy. The Appli-Card uses the Apple II as its I/O processor by loading the computer with a series of software drivers that handle the keyboard, video output and disk I/O. These drivers make the Appli-Card easy to configure because a driver can be altered or installed to meet a particular need.

The most notable capability of the drivers is the Softvideo feature. The Softvideo driver makes the Appli-Card a snap to use with any CP/M software. It allows the software to think it is "talking" to a popular terminal, such as the Televideo 920, and then remaps the Apple's video screen or an 80-column card. The translation allows for 80-column cards with or without inverse video or Soroc 120/140, Televideo 920/950, Hazeltine 1500, VT-52 or Adds Viewpoint terminals. Softvideo also allows for 255-column horizontal scrolling when using the Apple's 40-column screen, the upper/lowercase entry and display. There is even a mode that allows 70 columns using the Apple's hi-res screen. A program called CONFIGSV allows the user to control the features of the Softvideo driver including configuration for the Apple's 40-column screen, the 70-column hi-res mode, 80-column cards and even external terminals.

The utilities that come with the Appli-Card include the standard CP/M utilities (i.e., PIP, STAT, etc.); COPYFRMT, a disk format and copy program; ADOSXFER, a file transfer program that allows transfer of files between DOS and CP/M; CONFIGSV, the Softvideo configuring program; INSTALL, a program for installing drivers such as Softvideo to the system; and DLDRIVER, a program for downloading new drivers to the Apple. All of these utilities are menu driven and easy to use.

The Appli-Card uses the standard Apple CP/M disk format, and almost all of the software that is sold for Apple CP/M can be used with it. However, since the Appli-Card is a *true* CP/M system, almost any software downloaded from other CP/M systems can be run on it. There is only one group of software that will not work with the Appli-Card—software modified to take advantage of the hybrid nature of some other Z-80 cards. The Apple version of WordStar, version 3.0, does direct disk access to the Apple disk drives in the hope of speeding up the system. While it does increase the speed somewhat, it makes this version of WordStar a dedicated program for Apple CP/M only. Version 2.26 of WordStar is the standard CP/M version and works perfectly well with the Appli-Card. Any program that will directly access the Apple II's hardware will not work. However, most programs access the system through CP/M.

The documentation supplied with the Appli-Card has recently been revised and it supplies the user with the basic information required to use the system and its utilities. If more information is required for a particular software project or OEM application, it is available from Personal Computer Products Inc. for a small charge.

Overall, the Appli-Card comes up with exceptional ratings in all categories. Both the hardware and the software are excellent, the documentation is good, and the support is also very good.

The Appli-Card is available from Personal Computer Products Inc., 16776 Bernardo Center Drive, Suite 202, San Diego, CA 92128. Price is $295 for the 4 megaHertz board. ■

Jay D. Weiss
West Hartford, CT

Wildcard

In contrast to copy programs on disk, such as Locksmith and Nibbles Away, Wildcard is a firmware peripheral board, with an attached cable, that plugs into any vacant Apple slot except 0 or 6. In general, Wildcard will copy any program that resides completely in the Apple's memory and does not require multiple disk access. Its use requires a 16K memory expansion card, though many of the final backup copies can be run on a 48K machine. I have found no hardware compatability conflicts with other boards co-residing with the Wildcard: parallel printer, modem, 80-column and CP/M.

Wildcard's installation is simple. The only evidence of its presence is the cable, with a push button for the operator's use that must be kept at hand when the board is activated.

Copying Process

The first few times, the copy process may seem somewhat involved. First boot a game disk in drive 1. Since some games have an initial graphics display, before the disk accesses the game, it is important to wait for the second and final disk access. When the second display (sample run or instruction screen) appears, press the red button at the cable's end. All screen activity stops. Press return. At that point, the first Wildcard menu appears. It contains the choices: BOOT, CLEAR AND BOOT, RESTART, and MONITOR. Remove the game disk and insert the Wildcard system disk in drive 1. Press B for BOOT.

Menu 2 appears. Its options are: SAVE, LOAD, RESTART, CLEAR AND BOOT, TURNKEY SYSTEMS, UTILITIES, AND OPTIONS. Press 0 for OPTIONS. Since our example involves an arcade

> "It allows the user to view and modify the two text screens and hi-res screens and save those modifications."

game, we select GRAPHICS, ALL (graphics), PRIMARY (HGR), and HI-RES. Other options are for all text (usually business programs), text screen, mixed text and graphics, and setting peripheral slots. Return to Menu 2 and select TURNKEY SYSTEMS.

Menu 3 appears. Its three options are AUTOBOOT (48K), RAM AUTOBOOT (64K), and WILDCARD MAIN MENU. We choose the first option. After the system disk loads that program, we're instructed to place a blank disk in drive 1; then, press return. The message CREATING DISK appears on the screen. In about two minutes, the final message BOOT or TURNKEY appears. Press B. The time involved to make a copy is no more than that required by nibble copiers.

Wildcard Advantages

What is the advantage of the Wildcard copy over a nibble-copied verson? For one thing, you can make a backup of the Wildcard backup by using any simple copy program (e.g., COPYA on the Apple 3.3 system disk).

Next, it is possible to compress some 48K Wildcard copies so that they can be transferred, using the FID program, to another disk. More than one program can be put on a disk if you can compress the Wildcard version (though that is not always possible).

Finally, you can interrupt a game at a certain level and save it at that level. I found this feature very helpful when I was reviewing a somewhat frustrating game program. Rather than going through the early stages of the game repeatedly to reach new levels of difficulty, I saved it at progressively higher levels and resumed play at that higher level.

New Features

The distributor has recently sent me a new utilities disk that allows the user to attempt compressing the three major binary files, created on the original Wildcard copy, into a single binary file. It also permits a disassembled listing from a given point in a program to be dumped to the screen or the printer. Finally, it allows the user to view and modify the two text screens and hi-res screens and save those modifications.

This review does not cover all the features of the Wildcard. I think it is a worthwhile addition for several reasons. It has features not available on nibble copiers. Where it will compress 48K programs, you can save money on backup disks. Several programs can be put on a single disk rather than requiring a separate disk for each program. It also teaches a great deal about programming and the disk operating system.

What are the drawbacks? First, the casual user will not be able to copy multiple access disks conveniently. Second, because most utilities use only drive 1, there is substantial disk swapping. Finally, there is the documentation. While it is adequate, it is not at the same quality level with the board and the software.

The manual is incomplete. The sometimes skimpy level of detail and the varying levels of writing require too much effort from the reader. This device and its accompanying software have very sophisticated features. However, if a user does not have complete instructions for taking advantage of all those features, they are nearly useless.

By the time this review is published the distributor says there will be three important developments: 1. There will be a more complete, professionally-written manual. 2. There will be another utility disk with very desirable features—two disk drives to produce copies and the selection/options routine will be greatly simplified. 3. There will be a version compatible with both the II Plus and the IIe. The new version will cost $10 more. Current owners can send a copy of their original system disk to the distributor, along with $10, and they will receive the new options.

My final evaluative question is always, "Is the product worth$139.95?" In this case, I am buying the Wildcard sent to me for review purposes.

Wildcard is available from East Side Software Co., 344 East 63 Street, Suite 14-A, New York, NY 10021. ■

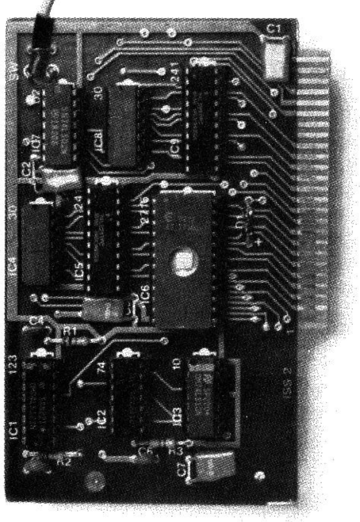
William E. Evans
Somerset, NJ

NEW FOR APPLE II AND IIe

WILDCARD™

MAKES BACK-UP COPIES OF PROTECTED SOFTWARE QUICKLY, EASILY, WITH JUST A PUSH OF A BUTTON.

New software locking schemes have rendered even the latest generation of copy programs virtually unusable. Locksmith™, Nibbles Away™ and other "Nibble copiers" require complicated parameter settings, much patience and great effort to use. More often than not, the results are disappointing. WILDCARD is different. Rather than copying disks track by track, WILDCARD ignores the disk and any copy protection encrypted on it. Instead, WILDCARD takes a snapshot of memory in your Apple® II. Now you can make back-up copies of protected software with the push of a button.



Software is not copy protected. System requirements: Apple II Plus with 64K and DOS 3.3 or Apple *IIe*. Franklin Ace also supported.

*Wildcard does not operate with CP/M® or other microprocessor based software.

FEATURES

- □ Hardware copying device... push button operation.
- □ Copies 48K memory resident software, most 64K software.
- □ No programming experience or parameters necessary.
- □ Backs up DOS 3.2 and DOS 3.3 disks.
- □ Creates DOS 3.3 unprotected and autobooting disks.
- □ WILDCARD lives in any slot. Undetectable by software.
- □ Produces autobooting disk in 2 minutes.
- □ Copies are DOS 3.3 compatible.
- □ Copies become accessible for alterations.
- □ Simple, easy-to-use software included.

WILDCARD ***Utility Disk 1*** also included, featuring:

- □ Automatic program compression and BRUN file maker.
- □ Multiple programs can be placed on the same disk.
- □ Recreates basic files to load and save.
- □ Files can be placed on a hard disk...and more.

WILDCARD $139.95

Order direct from East Side Software Co., 344 E. 63 St., Suite 14-A, New York City 10021, 212-355-2860. Please include $3.00 for shipping and handling. Orders outside continental U.S. please add $10.00 for shipping and handling. Mail and phone orders may be charged to MasterCard and VISA.
N.Y. State residents add sales tax.
Dealer inquiries welcome.

IMPORTANT NOTICE: The WILDCARD is offered for the purpose of enabling you to make archival copies only. Under the Copyright Law you, as the owner of a copy of a computer program, are entitled to make a new copy for archival purposes only and the WILDCARD will enable you to do so. The WILDCARD is offered for no other purpose and you are not permitted to utilize it for any other use, other than that specified.

inCider

Index to Advertisers

*This advertiser prefers to be contacted directly.

For further information from our advertisers, please use the Reader Service card.

Circle 385 on Reader Service card.

Don't Pick Just Any Card

Pick the APPLI-CARD™ and APPLI-DISC™ combination and be a Sure Winner!

If you have an Apple® computer, we can deal you a once-in-a-lifetime hand that just can't be beat.

Now you can plug our APPLI-CARD into your Apple and have total CP/M capability. Adding the APPLI-DISC you'll have the potential to speed overall program execution by greater than 10 times while increasing your data storage by more than 30%.

We give you a choice of either a 64K or 128K ram extender. Add this to the APPLI-CARD's existing 64K and you have a 128K or 192K APPLI-DISC. Eliminate the need for an expensive add-on disk drive and end up with a faster system with more storage and more memory for a lot less money!

We've dealt you an unbeatable combination. Now it's up to you to call and cash in on a great deal. Ask about our 8088 Card, another innovative product from PCPI.

4 MHz 64K APPLI-CARD w/APPLI-DISC Software	$295
4 MHz 64K APPLI-CARD w/ 64K Ram Extender / APPLI-DISC Software	$395
6 MHz 64K APPLI-CARD w/APPLI-DISC Software	$375
6 MHz 64K APPLI-CARD w/ 64K Ram Extender / APPLI-DISC Software	$495

PCPI
Personal Computer Products, Inc.

THE COPROCESSING CO.
16776 Bernardo Center Drive
San Diego, CA 92128
(619) 485-8411

Prices subject to change without notice

Book Reviews

> "If you're just curious as to how your computer might control your house, this book is a good beginning."

Screen Writer II Made Easy

by Judy Mandell

PO Box 7063
Charlottesville, VA 22906
Softcover, $5.95

Have you ever been impressed by a program's power but at the same time terribly frustrated by your inability to harness that power? On-Line Systems' Screen Writer II word processor is a perfect example of this. Loaded with features and selling for a reasonable price, Screen Writer is one of the most difficult Apple word processors to master. Part of the blame can be foisted on the manual provided by On-Line Systems. Beginning users need something more elementary. At first glance *Screen Writer II Made Easy* looks like the way to fill this void.

The author, who plays the part of "I," tells the reader, "you," about the Screen Writer II features needed by "most users most of the time." She does this in 18 lessons. The first 14 lessons cover basic skills like saving files and editing while the final four lessons illustrate the production of form letters. When you consider that the entire book encompasses just five 8½-by-11 sheets of paper it becomes obvious that there is little room for detail.

The brevity of *Screen Writer II Made Easy* creates several problems. Readers will gain instant gratification but little in the way of comprehensive understanding. If you want to fully utilize Screen Writer II you'll still have to pay your dues. This book only delays the inevitable.

A second flaw stems from the book's grade school style writing. The poor grammar would be easy to overlook if the contents were more substantive. An example of this patronizing style is the author's statement, "To print a letter on individual sheets of stationery, you must first remove the computer paper from your printer."

Screen Writer II Made Easy is a well-intentioned book attempting to meet the widely acknowledged need for better documentation. Unfortunately the amateurish presentation and low informational value make it a dubious purchase. ■

Timothy Daniel
Oxford, OH

The Apple Connection:
An Introduction to the Techniques and Principles of Apple Computer Interfacing.

by James W. Coffron

SYBEX Inc.
2344 Sixth St.
Berkeley, CA 94710
Softcover, $12.95

So what can you do with your Apple? You've had it for some months now, have learned a little Basic, and you're no longer afraid to tinker a bit with the insides. You've found that keeping up an electronic checkbook is too much paperwork, while that educational game for six-year-olds costs $60.

A computer systems engineer wrote this book to challenge you to hook up your Apple to the outside world and perform some useful functions, such as detect burglars who would try to steal your glorified typewriter. The book's aim is to give the novice "all the essential information for interfacing and controlling external devices," such as around your home. The come-on is just what the computer store salesperson said—maybe you can save some money by monitoring your furnace or something!

We learn that an Apple is a good machine to attempt this on because the necessary electrical signals are readily accessible. What holds people back is, as usual, the software—writing a program or designing a logic circuit to do the job of massaging these signals.

Here is where the book is especially useful for novices, as it employs only Basic commands such as PEEK and POKE instead of the more common, rapid and difficult machine language routines. The schematics give only the outline of the logic and do not detail how to solder or wire-wrap. If you buy the I/O boards recommended in the text you can get through most of the book without any kit building.

The author's wordy repetition, especially in the passive tense, will annoy readers beyond the beginner's stage. He takes 74 pages to explain how to write a simple Basic program to turn on some light-emitting diodes. The novice, on the other hand, doesn't get enough details to finish anything practical. Coffron gives most of the theory, but wanders off just when we reach the meaty parts.

In the end it's not clear how practical this gadgetry really is. The home security application developed in chapter 5 is just silly. Once you buy all those switches and string all that wire, you find that leaving your computer on all the time exposes it to power surges coming through the giant antenna you have constructed! It would be simpler to use the game-port paddle connector if you need a cheap burglar alarm or digital thermometer, but Coffron doesn't even mention this standard Apple conversion.

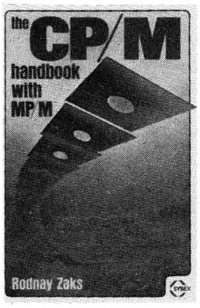
In order to really control your house, you'll need a clock in your Apple, and then the potential to interrupt your Apple when you're using it. None of these are discussed in the book.

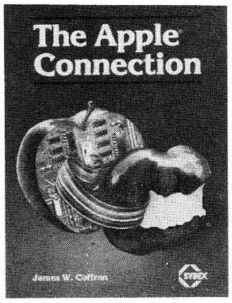
The denominator in the equations on page 185 should be 255, not 256. A glossary does explain some of the electronics and computer jargon used. The reading is high school level.

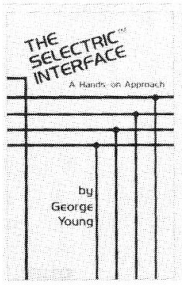
If you're just curious as to how your computer might control your house, this book is a good beginning. You already know how it controls your life. ■

Eric Eldred
Derry, NH

inCider BOOK SHELF







THE BOOK OF APPLE SOFTWARE—edited by Jeffrey Stanton, Robert P. Wells, Ph.D., and Sandra Rochowansky. Are you bewildered by the thousands of Apple II programs on the market? Here is the advice you need, with descriptions, ratings, and evaluations of over 500 of the most popular programs for the Apple II. You'll find reviews of business, education, game, and utility programs, and advice on hardware options and software vendors. BK1265 $19.95.

THE APPLE CONNECTION—by James W. Coffron. Connect your Apple to household appliances for greater control. With this book you will learn about elementary interfacing and about BASIC programming, including input/output techniques and devices, building real systems, and even analog to digital and digital to analog conversion. All programs are written in BASIC and no prior electronic knowledge is required. BK1262 $12.95

MICROBOOK: DATA BASE MANAGEMENT FOR THE APPLE II—by Ted Lewis. This book provides you with an affordable data base management system for your Apple II. These programs turn your Apple II into a combination filing cabinet, information gathering/retrieval system and data processing engine. Written in Pascal, the programs simulate a library. Information is maintained and broken down into books, chapters and pages and index to pages. Photographs of the Apple II screen are abundant, and they show you step-by-step the effect of each of your entries. Microbook can be used for almost any application involving the storage and retrieval of information. BK1261 $19.95

THE CP/M HANDBOOK (with MP/M)—by Rodnay Zaks. A complete guide and reference handbook for CP/M—the industry standard in operating systems. Step-by-step instruction for everything from turning on the system and inserting the diskette to correct user discipline and remedial action for problem situations. This also includes a complete discussion of all versions of CP/M up to and including 2.2, MP/M and CDOS. BK1187 $14.95.*

MASTERING CP/M—by Alan R. Miller For advanced CP/M users or systems programmers who want maximum use of the CP/M operating system, this book takes up where the *CP/M Handbook* leaves off. It will give you an in-depth understanding of the CP/M modules such as CCP (Console Command Processor), BIOS (Basic Input/Output System), and BDOS (Basic Disk Operating System). It explains the incorporation of additional peripherals to the system, console I/O, the use of the file control block and much more. It also includes a library of useful macros and a comprehensive set of appendices. BK1263 $15.95

SOME COMMON BASIC PROGRAMS, APPLE II EDITION—By Lon Poole et al. A powerful collection of financial, statistical, home management and mathematics programs—76 in all. Each program is presented with BASIC source code, operating instructions and descriptions. If you're a beginning programmer you can learn from this book what well designed and documented programs look like. BK1232 $14.99

THE CUSTOM APPLE AND OTHER MYSTERIES—by Winifred Hofacker and Ekkehard Floegel. This is *the* guide to customizing Apple software and hardware, published by the folks at IJG. It contains such hands-on information such as: data acquisition and control applications, Programming the 6522 Internal timer, Constructing the 6522 I/O board, An Eprom Burner for the Apple Computer, An Eprom/RAM board, The Apple Slot Repeater, and much, much more. BK1246 $24.95.

APPLE GRAPHICS AND ARCADE GAME DESIGN—by Jeffrey Stanton The only book available that explains how to design arcade games from start to finish through the use of text, flow charts and working examples. Learn how to speed up your graphics, and the theory of how to design a playable game. This book requires a solid foundation in BASIC programming on the Apple II. BK1259 $19.95

THE VISICALC BOOK—APPLE EDITION—by Donald H. Beil. If you are presently using Visi-Calc on your Apple II or want to learn more about its expanded uses, this book is for you. It will show you how to build a model, enter your data, and begin to explore all of your "what if" questions about any aspect of your business. There are also a large number of practice problems included in this book to help you become a skilled user. BK1268 $14.95

THE APPLE II USER'S GUIDE—By Lon Poole, Martin McNiff, and Steven Cook This guide is the key to unlocking the full power of your Apple II or Apple II Plus. Topics include: "Applesoft and Integer BASIC Programming"—especially how to make the best use of Apple's sound, color and graphics capabilities. "Machine Level Programming," "Hardware Features"—which covers the disk drive and printer, and "Advanced Programming"—describing high resolution graphics techniques and other advanced applications. Well organized and easy to use. BK1220 $16.95

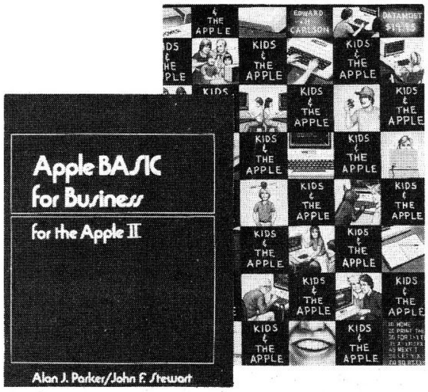
NEW

APPLE BASIC FOR BUSINESS: for the Apple II—by Alan J. Parker and John Stewart. Unlike most introductory BASIC books, this book uses files extensively. It is written specifically for the Apple II microcomputer with DOS Version 3.2. All programs presented are compatible with DOS Version 3.3. With the emphasis on problem-solving, the focus of this book is the point at which problem elements meet language capabilities. BK1247 $15.95

ASSEMBLY LANGUAGE PROGRAMMING FOR THE APPLE II—by Robert Mottolz. This comprehensive, easy to understand introduction provides solid groundwork for getting started in assembly language programming on the Apple II." Many subroutines written in assembly language are provided, and most explanations are shown with equivalent examples in BASIC. There's an excellent section on hexadecimal arithmetic included, as well as appendices for further study. BK1249 $12.95

APPLE MACHINE LANGUAGE—by Don Inman and Kurt Inman. APPLE MACHINE LANGUAGE builds upon your previous knowledge of BASIC, and teaches you the machine language in small, easy, completely illustrated steps. Following this guide, you will be able to write machine language programs directly, using the Apple System Monitor. Each new program is thoroughly presented in functional blocks, with sketches of how each step will actually appear on the video screen. Soon you will be entering and executing your own machine language programs, with predictable results! BK1248 $14.95

THE SELECTRIC INTERFACE—by George Young. You need the quality print that a daisy wheel printer provides but the thought of buying one makes your wallet wilt. SELECTRIC™ INTERFACE, a step-by-step guide to interfacing an IBM Selectric I/O Writer to your microcomputer, will give you that quality at a fraction of the price. George Young, co-author of *Kilobaud Microcomputing* magazine's popular "Kilobaud Klassroom" series, offers a low-cost alternative to buying a daisy wheel printer. SELECTRIC INTERFACE includes: step-by-step instructions, tips on purchasing a used Selectric, information on various Selectric models, including the 2740, 2980, and Dura 1041, driver software for Z80, 8080, and 6502 chips, tips on interfacing techniques. With SELECTRIC INTERFACE and some background in electronics, you can have a high-quality, low-cost, letter-quality printer. Petals not included. BK7388 (125 pages) $12.97



MOSTLY BASIC: APPLICATIONS FOR YOUR APPLE II, BOOK 1—by Howard Berenbon. This book provides you with 28 ready-to-use, BASIC language programs which have been completely tested and debugged for use on your Apple II. Includes a telephone dialer, digital stop watch, spelling test, a house buying guide, a gas mileage calculator, and many others useful to businessmen, hobbyists, scientists, and computer enthusiasts. BK1251 $12.95

MOSTLY BASIC: APPLICATIONS FOR YOUR APPLE II, BOOK 2—by Howard Berenbon. A second gold mine of fascinating BASIC programs, including two dungeons that test your math and history abilities and another one that's strictly for fun, eleven household programs, a monthly savings plan and six more on money or investment, two that test your level of ESP, and more—32 in all! Excellent for beginning or advanced computerists. BK1252 $12.95

KIDS AND THE APPLE—by Edward H. Carlson. Whether you are a kid, a parent, or a teacher, this book is something unique. It starts with the bare bones introduction to programming, leads quickly to more interesting programs, and gives anyone who uses it a complete knowledge of Applesoft BASIC. Lively illustrations, notes to parents and teachers and questions for the reader are sprinkled throughout the book. While this guide is aimed at 8–16 year olds, adults will find it equally attractive as a beginning book for use the Apple personal computer! BK1253 $19.95

For Toll Free Ordering Call 1-800-258-5473

*Use the order card in this magazine or itemize your order on a separate piece of paper and mail to *inCider* Book Department • Peterborough NH 03458. Be sure to include check or detailed credit card information. No C.O.D. orders accepted. All orders add $1.50 for the first book, $1.00 each additional book for U.S. and foreign surface. $10.00 per book foreign airmail. Please allow 4–6 weeks for delivery. Questions regarding your order? Please write to Customer Service at the above address.

Software Reviews

Computer Gin Rummy

Any gin rummy fanatics out there? For those evenings when you can't find anyone to play, Computer Gin Rummy makes an ideal opponent. All games are played in solitaire fashion with the Apple as your opponent.

For those with only a passing familiarity with gin rummy, a brief rundown of the rules seems in order. Each player holds ten cards in his/her hand. The idea is to form runs of three or more cards, either all the same number (e.g., three kings) or three consecutive cards of the same suit (e.g., four, five, six of hearts). If all the cards in your hand are complete runs (also referred to as melds), you have a gin and win the hand. A gin scores 25 points plus those points in the opponent's hand that are not part of a meld.

You can also win by "knocking." If the points for all cards in your hand that are *not* part of a meld add up to ten or less, you may knock. The opponent then has the opportunity to play off any of the cards in your hand that have not been melded. For example, suppose your hand includes a run from the eight of hearts to the jack of hearts. If your opponent has either the queen or seven of hearts, they may be played on your hand, thus reducing his/her point count. After this is completed, if your points are fewer than your opponent's, you score the difference between the two hands. If the opponent has fewer points, he/she scores the difference plus a bonus of 25 points.

Computer Gin Rummy also offers two variations of gin—knock and one-meld. In knock, you may knock at any time during the game, whether you have fewer than ten points or not. There are no layoffs at the end (as in gin above). The score is the difference between the unmelded points in the two hands. If you have a rummy—all cards melded—an additional 25 points is scored. If you knock and your opponent has fewer points, he/she scores the difference plus ten bonus points.

One-meld is played to a rummy only; all cards must be part of a meld. The winner receives the total of all cards held in his/her hand plus 25 bonus points.

Special Features

Cards are displayed on the hi-res screen; the Apple's cards are covered at all times and yours are face up. The number of cards remaining in the deck and those in the discard pile are always shown. When it is your turn, you first take the card from the top of the deck (D command) or the card your opponent discarded (U for upcard). You may then either discard by pressing a letter for the card's position in your hand (A–K), knock or gin by pressing W for win, or rearrange the cards in your hand (R). You may rearrange your cards as often as you like during your turn.

All play is directed from the keyboard. If you make a mistake and discard the wrong card, the program gives you three seconds to change your mind—a nice feature. Until you become used to the keyboard play, this will come in very handy. You may also toggle the sound off or on. Personally, I find the sound of the cards being rearranged rather cute.

Evaluation

Computer Gin Rummy is undoubtedly the quickest opponent I have ever played—computer or human. No long, tedious waits here! It plays an above average game and gives you a real run for your money. You may find, as I did, that rearranging cards with key presses and discarding cards in the same fashion is distracting. In many cases, fear of throwing the wrong card by mistake made me forget which discards the Apple drew. In this way, the Apple picks up an additional edge.

I discovered a small bug in the program's hand evaluation routine. In a hand of one-meld I held two five-card runs (four to eight of clubs and four to eight of diamonds). No matter how many times I tried, the program insisted that I was not a winner. Since the rules specifically state that five-card runs are acceptable, perhaps the error was caused by the fact that the runs were identical but in different suits.

Otherwise, I find the program superb. Everything considered, if you enjoy gin (the card game, that is), you're sure to enjoy Computer Gin Rummy.

Computer Gin Rummy is sold for $29.95 by DataMost, 9748 Cozycroft Avenue, Chatsworth, CA 91311. ■

Steven Schwartz
Pittsburgh, PA

Ice Demons

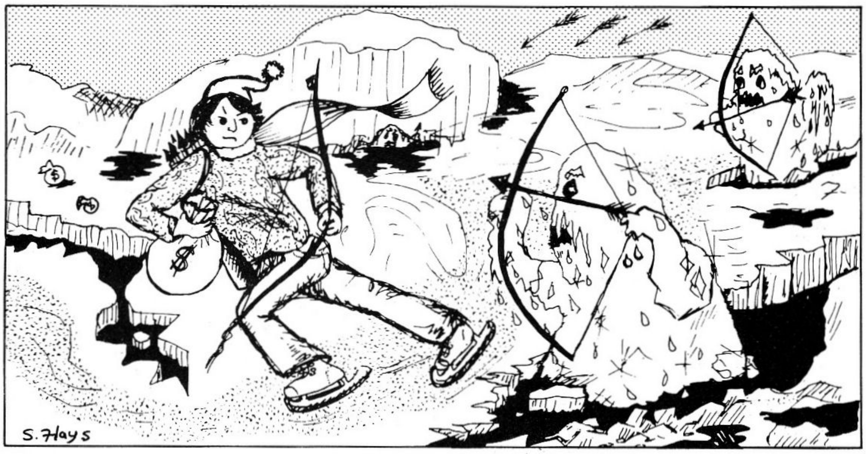
An idle hour spent skating on a frozen pond seems like a pleasant way to spend an afternoon—still more pleasant if, as you skate, you encounter packets of money lying unattended on the ice.

But remember the old saying: "There's no free lunch." Suppose, as you reach down to retrieve the cold cash, that sneaking up behind you is a powerful demon poised to tear you limb from limb. Don't look now, but. . .

That's the premise of Ice Demons, an arcade game from Morningstar. As the program begins you are given the option of playing solo, in a two player team or against a human player, competing to see who can defeat the greatest number of monsters and win the most treasure. You are in control of one of two adventurers, Tom or Ray. Using a game paddle, you can make your adventurer skate around the high-resolution graphics screen, snatching treasures and firing arrows at attacking demons.

And what of the title players, the Ice Demons? They are a race of monsters that live under the ice, breaking through to devour the occasional adventurer foolish enough to skate past their lairs. Mere contact with these creatures is instantly fatal; but the clever hero, pressing rapidly on his paddle button, can defeat the detestable horde by barraging them with arrows. Of course, they will be coming from all sides, shooting arrows of their own, paralyzing you, leaving you helpless as they close in. The moral: Keep moving!

Some demons are decidedly more



dangerous than others. One group, the Widowmakers, will make a sudden leap at your adventurer, giving you barely enough time to react. Another group, the Willowisps, are seemingly impervious to the arrows that fly right through them. The Fireball comes after you, hurtling out of nowhere. And the High Master... well, that's better experienced than described. The more dangerous and difficult it is to kill a demon, of course, the more points you win.

Ice Demons is moderately difficult to master. The most important task is learning how to press the paddle button fast enough to shoot a barrage of arrows; holding it in is the command that makes your hero skate. It is more than a little frustrating to find yourself skating into the arms of a demon when what you really want to do is fire arrows at him. Once you have this basic technique mastered, you can develop some of the other skills suggested in the documentation: catching arrows, playing possum in hopes that a demon will pass right over you, and "nosing," which is using an arrowhead feature of the hero shape to block monsters emerging from their holes and to shield yourself.

You begin the contest with five lives. The game is finished when you've been killed for the fifth time. After each game the program gives you the opportunity to record your score, letting you see if progress is being made. The high scores of teams and of two player competitions are also displayed.

Youngsters as well as adults should enjoy making the heroes skate around the screen as they shoot arrows and dodge monsters at the simplest levels of skill. Arcade game enthusiasts will find challenge in breaking through to higher levels of play.

Visually, the game begins with a fine hi-res graphics display and a display of the various shapes representing the adventurers in the simulation. The theme of Bach's Toccata in D minor sets an appropriately threatening mood.

This is a game that offers far more than you encounter on the first play. The world of the Ice Demons is worth repeated visits.

Ice Demons is manufactured by Morningstar, 39 Florence St., San Francisco, CA 94133. The game is designed for any 48K Apple II or II Plus with 3.3 DOS. The suggested price is $29.95. ■

Brian J. Murphy
Fairfield, CT

Zargs

Once again, the Earth is threatened by hordes of aliens from outer space. This time they're from the region around the Crab Nebula. The mighty Zarg warship, in Earth's orbit, must be completed if you are to repel the alien menace. Before the Zarg mastership construction can be completed, four interstellar battleships must be launched and docked with the Zarg superstructure. It is your job, as battleship commander, to fly the needed ships and dock them with the Zarg mastership. The world is depending on you!

Zargs has four stages of play; each stage increases in difficulty as the player moves up the levels. The object in the first three levels is to successfully fly four of your ships through a variety of obstacles and dock them with the orbiting mastership. Only after successfully docking four of your ships does the game progress to the fourth stage—the stage in which the alien attack begins.

The game begins with an arsenal of five ships. Since four ships must be docked before you can move to the top level of play, there is not much room for error. In level one you must launch your ship and fly through an obstacle course of other aircraft, deadly clouds and hot air balloons. At level two the hazards move in horizontal rows across the screen, with just enough room to slip your ship between as they pass. Contact with any of the objects will cause the immediate destruction of your craft.

Points are awarded for each row of obstacles passed and an orbit bonus of 250 points begins counting down as soon as you launch. The quicker you attain orbit the more bonus points you receive. Fortunately, the hazards in level one move slowly; getting into orbit is accomplished without much difficulty. The trick is to get there as quickly as possible.

Flying your ship to the orbit window automatically moves you into level two. This time the hazards are moving faster—in vertical rows up and down the screen. Again, you must launch your ship and fly to another orbit window, avoiding the space debris and alien craft.

In level two extra points can be earned by shooting down alien space craft. If you earn enough points you will be awarded an extra ship. Go for it! It takes four ships to complete the mastership and you begin the game with only five.

You may stay at level two, earning extra points, until an alarm sounds to warn you that your weapon will soon become inoperative. When you hear

APPLE®...MEET THE OPERATOR™

MODEM TECHNOLOGY WITHOUT PAYING THE PRICE ON THE BOX

If you're sold on telecomputing but hung up on the price, check out THE OPERATOR. For just $159.95 *complete,* you'll find no other modem that gives you crisper Apple communications.

And you don't have to be a technical wizard to get started. THE OPERATOR includes all the sophisticated technology on a single circuit board. No external box. No need for expensive add-ons. THE OPERATOR is ready to plug into any communications system, world-wide, the day she arrives.

This modem comes with complete documentation even a beginner can understand, *plus* start-up software programs (on disk), so you don't have to worry about added expenses or frustrating delays.

SPECIAL INTRODUCTORY OFFER

Order THE OPERATOR today and be our guest for 60 minutes on T•I•M•E•C•O•R's™ international communications network via GTE Telenet.

YES, I need THE OPERATOR.™ Rush her to me today!

Name ____________________

Company ____________________

Address ____________________

City ____________ State ____ ZIP ______

Phone (   ) ____________ ☐ Dealer ☐ Business ☐ Home

Computer ____________________

☐ Check enclosed ($159.95*) ☐ Send more information

*Mass. residents add 5 % sales tax. Shipping & handling included for continental U.S.

T•I•M•E•C•O•R™
Four Longfellow Place, P.O. Box 8928, Boston, MA 02114 **IC-2**

$159.95 COMPLETE

NO EXTERNAL BOX
HARDWARE • DOCUMENTATION
START-UP PROGRAMS (ON DISK)

FEATURES

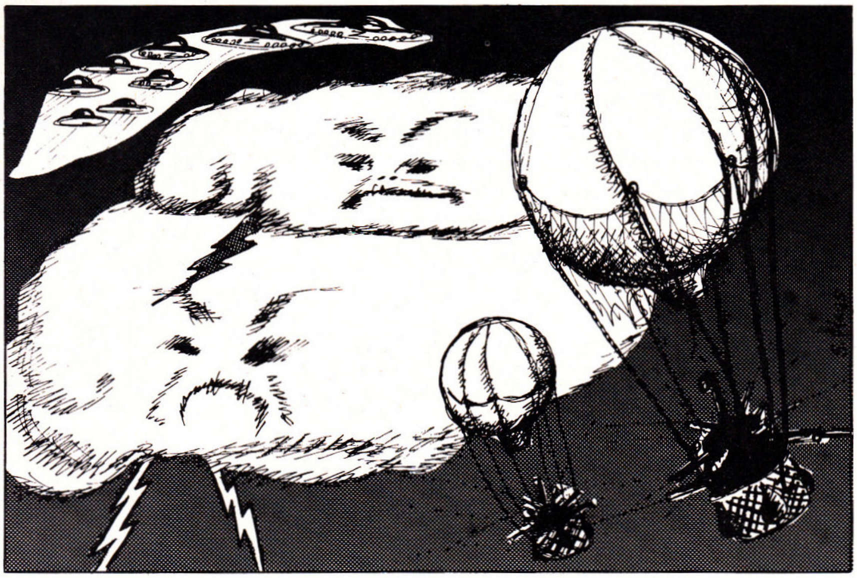
- 110/300 baud • Full & Half Duplex
- Auto Answer w/Disconnect
- Rotary, Touch-Tone,™ Key Set Compatible
- Totally Self-Contained (No RS-232 Required)
- Single Modem Chip Reliability
- Interfaces with Most Communications Packages

FOR FASTER SERVICE CALL TOLL FREE
MasterCard and VISA only
1-800-852-5000

THE OPERATOR is a trademark of T•I•M•E•C•O•R.
Apple is a registered trademark of Apple Computer, Inc.
Touch-Tone is a trademark of American Telephone & Telegraph.

T•I•M•E•C•O•R™
REACHING PEOPLE THROUGH TECHNOLOGY
The International Modem Exchange Corporation
Four Longfellow Place, P.O. Box 8928, Boston, MA 02114 VOICE: (617) 720-4090 MODEM: (617) 720-3600

Circle 76 on Reader Service card.



the alarm it's time to fly to the second launch window and prepare for docking with the Zarg ship. If you succeed in piloting your ship to the launch window without being clobbered by fast moving debris and alien craft, you get the chance to try your hand at deep space docking.

Level four requires a steady eye and good digital (as in finger, not electronic) coordination. In the vacuum of space with no weight or friction the laws of inertia make themselves known. A body in motion tends to stay in motion until acted upon by an outside force. If you're good (or lucky), that outside force won't contact the mastership at any point other than the docking port. Contact anywhere other than a docking port will destroy your ship, and you will find yourself back at level one with one ship less than when you started.

At the third level any motion applied to your ship will continue in the same direction until cancelled by an equal and opposite force. This, in my opinion, is the best part of the game. The action is very smooth and the docking procedure is a fair challenge.

It's too bad levels one and two don't offer the same degree of challenge and creativity.

This game is written in the GraForth language. With the exception of the docking level of play and the transformation of the mastership, the graphics of this game are rather ho-hum. This game does not do the capabilities of GraForth justice.

After finally docking four ships with the Zarg mastership, the superstructure transforms itself into earth's only hope to defeat the aliens. As soon as the Zarg has completed its transformation, the aliens begin their attack. The battle continues until the Zarg is destroyed (most probable) or until the aliens surrender (not likely).

At the beginning of the battle you are given a protection rating based on your total points from levels one, two and three. Each time you are hit by an attacking alien your protection is lowered, and each time you hit an alien your protection rating increases. Your fire power is limited to four weapons, each pointing to the cardinal directions of the screen. Because the huge mastership is not maneuverable, quick action and reflexes are required to hit the alien craft as they fly past.

With the exception of the docking sequences in level three, this game lacks creativity. All of the action is

Get Machine Language speeds for your Apple Games, Sorts, Utilities, Home programs, Number Crunching, and virtually everything else you program . . . Without having to program in Machine Language.

Typical speed improvements for your programs will range between 2 and 25 times faster than Applesoft! MacroSoft is even faster than Compiler-generated programs! Up to 10 times faster.

PROGRAM PERFORMANCE (relative to Applesoft)

MacroSoft is *not* a compiler. It is an Applesoft-like language that works with THE ASSEMBLER by Micro-SPARC to let you write programs that are converted directly into stand-alone Machine Language programs.

And MacroSoft adds a lot of programming goodies that were left out of Applesoft. Features like: HSCRN, HFIND, IF-THEN-ELSE, REPEAT-UNTIL, WHILE-WHILEN, BUTTON, SOUND, PAUSE and more will make your life and your programming a whole lot simpler . . . and much much faster.

We're betting that once you've tried MACROSOFT and THE ASSEMBLER you won't be satisfied with anything less . . . ever again.

Incidentally MacroSoft uses THE ASSEMBLER, which just happens to be the best Apple Assembler on the market. When you buy MacroSoft + THE ASSEMBLER you get a 100+ page User Guide for THE ASSEMBLER and a 50+ page User Guide for MacroSoft.

MacroSoft and THE ASSEMBLER are an unbeatable team! For a limited time, you can get both for $99.95. That's a saving of $20.00 over the individual purchase prices.

MACROSOFT and THE ASSEMBLER are available from your local Apple dealer. If he does not yet stock them, add $1.50 shipping/handling and send your order to Micro-SPARC Inc., Systems Division, P.O. Box 639, Lincoln, MA 01773. (617) 259-9710. (MA resident add 5% sales tax). We accept Visa and Mastercard.

MacroSoft requires an Apple II with Applesoft in ROM or an Apple II Plus with 48K memory, a disk, DOS 3.3, and THE ASSEMBLER.

Apple is a registered trademark of Apple Computer Co.

msi micro-sparc, inc.

making the complex . . . simple

P.O. Box 639
Lincoln, Mass. 01773

systems division

Circle 96 on Reader Service card.

played from the keyboard and the speed of the action is so slow that only the beginning game player will be challenged.

Zargs is available from Insoft Inc., 10175 SW Barber Blvd., Suite 202B, Portland, OR 97219. Price is $34.95. ■

Richard Brown
Oakland, CA

Evolution

In Evolution, you begin as an amoeba and try to rise up the evolutionary ladder to become—ta-dah—that supreme being, a human. To accomplish this feat, you are given five lives. You must successfully perform a different task in each of six scenarios. If you complete these without losing your lives, you're awarded a bonus life and treated to a special animated intermission show. You then return to the amoeba state, but at a higher level of difficulty.

Play is controlled with a joystick or

from the keyboard. The keys or button have different effects depending upon the scenario. As an amoeba, you must consume all of the DNA strands scattered about the screen while avoiding contact with spores, microbes and antibodies. Movement is allowed in any direction. Pressing the space bar (or paddle button) creates a temporary shield around the amoeba.

In scene two, you are a tadpole (looks more like a frog to me) trying to catch flies. You are positioned at the bottom of a pond and can only move from side to side. To catch the flies and avoid the fish that are attempting to eat you, the space bar (paddle button) is used to make you jump.

If you catch your quota of flies, you're ready for scene three. Here you are a rat that must burrow through dirt in search of cheese. Snakes, however, follow through the tunnels created and try to eat you. As a defensive measure, you may use the space bar (button) to drop a pile of dung, which, for reasons unknown to this reviewer, is lethal to the snakes. Your goal for this scenario is to collect five pieces of cheese.

In the fourth scene, you are a beaver intent on building a dam. To do so, you must fetch five sticks from the right side of the screen and add them to the dam on the left. But menacing alligators want to make a snack of you!

In scene five, you are a gorilla attempting to protect a hoard of oranges from a band of thieving monkeys. Knock five monkeys off the platform overhead (by hitting them with coconuts) and you advance to the final scenario.

As a human in the sixth state, you must dispatch ten mutants with a hand laser. Be careful. If you miss, the laser fire bounces off the sides of the screen like a billiard shot. Also, the mutants resent being shot at and occasionally fire back.

Standard arcade features have been included in Evolution: freeze play, toggle sound off/on, change from keyboard to joystick (and vice versa) and retention of the ten top scores. As an added attraction, a demo mode displays the action in the various scenarios and shows what's in store for you.

Play may be initiated at either a beginner, intermediate or expert level. Beginners start at level 1, intermediates at level 7, and experts at level 13. The same scenarios described above are used for the three modes, but at greater levels of difficulty.

The graphics and music are exceptional in Evolution, although the music is a little loud. A press of the S key, however, stops the music without affecting the regular game sounds. All in all, if this game gets the advertising and press it deserves, it should be a real winner. Even if it doesn't, buy it anyway. Evolution may be frustrating at times, but it's a lot of fun and shows off the Apple at its best.

Evolution is sold by Sydney Development Corporation, 129–444 Camino Del Rio South, San Diego, CA 92108. It costs $39.95. ■

Steven Schwartz
Pittsburgh, PA

Kaleido-Sound

Kaleido-Sound is truly a product for the video age. A fun addition to Passport Designs' more practical music systems and programs, Kaleido-Sound provides programs that react to the frequencies of music and produce kaleidoscopic designs on a color monitor.

System Requirements

Kaleido-Sound requires a 48K Apple II Plus, a disk drive with controller, a color monitor or television with rf modulator and a sound source. It is compatible with the Apple IIe, according to Passport Designs. The sound source can be either your stereo system or a tape player/recorder. Passport Designs provides the cable you need to connect the sound source to the Apple II cassette port.

The single disk comes with a two-page card of excellent instructions. After reading them, I was able to set up and begin running the first program within five minutes. This is an advantage if you are planning to use Kaleido-Sound for party entertainment. While the operating instructions note that the display might sometimes "freeze" and give instructions (press the space bar and check the connections) for correcting the problem, this did not occur while I was using Kaleido-Sound. Later, however, I used the freeze capacity of the space bar to hold the picture on the screen for photographs.

Themes

There are five different themes or types of pattern. You can easily change the colors and response time of each. To change the response time, you use the directional arrows. The number keys change the colors according to the tables shown on the instruction cards.

The first two themes are low-resolution graphics—a mirrored pattern and a hexagonal pattern of rectangles. Each of these uses ten colors and provides a beautiful show.

The other three themes are high-resolution graphics and are limited to five colors. They are differing patterns of moving dots (fireworks, mirrored kaleidescope and bubble kaleidoscope) which respond to different frequencies by placing the colors in different positions on the screen. I found these less appealing than the low-resolution graphics which resembled more closely the pattern of a kaleidoscope.

Music

Because the program reacts to the frequencies of the music, I experimented with several different types of recordings. A violin and piano duet produced a sparse reaction—not impressive at all—due to the narrow range of sound input. An orchestral piece by Beethoven produced a more complete series of images, and a recording of one of Bach's Brandenburg Concerti produced a magnificent display when using the shortest response time, due to its many interwoven sound patterns.

Rock music, with a steady background beat and multiple instruments also produced a full multicolored pattern. The most important factor in producing a full pattern is to have input consisting of a variety of sound frequencies. This is true for both low- and high-resolution graphics patterns. However, if you are used to very loud rock music, you will need to be extremely careful. Excessive volume levels may damage the cassette input circuit as it accepts only a 1-volt (peak to peak) signal. There is an advantage to this. If you are using Kaleido-Sound as a conversation piece, your guests will actually be able to hear the conversation!

Kaleido-Sound is a system that is designed to be used purely for fun, and as such, it's a great success. For those of us who are used to the constant visual images of the video age, listening to Gershwin while watching kaleidoscope patterns in shades of blue can provide a relaxing alternative to the evening news. And the use of multiple bright color patterns could be used to animate a party.

Kaleido-Sound has a retail price of $39.95 and is published by Passport Designs Inc., 116 North Cabrillo Highway, Half Moon Bay, CA 94019. ■

Janet Meizel
Davis, CA

The Graphics Magician

Here is a graphics development program for the Apple that provides five editors to allow complete shape animation. You can manipulate up to 32 objects. Included in this package is a hi-res picture/object builder suitable for any video artist or game designer. This design tool lets you develop software for the commercial market. The single disk is accompanied by a 32-page manual and revisional supplement.

I bought The Graphics Magician with the hope that it would help me develop computer assisted instructional programs for my seventh grade geography classes. I would judge my computer expertise to be a five on a scale of one to ten, so judge my comments accordingly.

First among the five editor routines is the Shape Editor. This routine lets you develop seven shapes, all similar in form, at one time. By toggling bits, the shapes may be shifted slightly to create sequential forms such as walking feet, wiggling antennae, fluttering wings, and more.

This process is by no means easy because you must consider many variables prior to the final compiling of the shape table. Colors change as dots are shifted horizontally. Toggling the bit to either high or low changes the color combination.

A two pixel wide border must be designed around the shape to prevent erasure of the background. Each shape must be adjusted in its 1 through 7 position and still maintain a 255 byte maximum storage size.

The Graphics Magician provides for full menu control in all editor modes. In this case, the Shape Editor's A command allows you to test your shape definition in its animated form while you control the speed with your paddle.

The Path Editor is *nifty!* Upon my first attempt to run a completed shape across its path, I saw my fluttering butterfly follow its predetermined trail. There are eight different directions for plotting a path. You may also control the shape to hesitate along its screen journey. The path sequence has three possible destinations: repeating the path, ending with the shape erased or ending with the shape held on the screen. You may also create several small paths, combine them, or switch from one to another.

When you first place the Magician into your disk drive, you'll want to preview the excellent demonstration of a marching alphabet, a multitude of moving shapes and high quality graphically composed pictures. This was my first experience at seeing my Apple display such vivid colors and sharp lines in a picture.

Later, using the Picture/Object Editor, you'll discover that your artistic talents are limited only by your haste to try out a new idea. Your joystick or paddles will become paint brushes of varying sizes. With a flick of the wrist you swirl your brush on a palette of 100 brilliant hues and start creating. This particular editor is easy to use, and is itself worth the price of the package.

But wait—there's more! A Super Shape Editor is included. This routine allows you internal control of color and scale with your shapes. Like the Picture Editor's feature, the Super Shape Editor has a data compacting feature that puts more on a disk. A binary file transfer utility permits you to transfer files from one disk to another while giving you the starting address and length of each file. That little utility is worth its weight in gold by itself.

Penguin Software (830 4th Ave., Geneva, IL 60134) is to be commended for allowing the owner to freely copy this disk for backup purposes. Each disk is labeled with a registration number which may be filed with the manufacturer. Purchase price is $59.95. Penguin also permits an author to publish software that utilizes The Graphics Magician for only the cost of proper credit. An update service for revised editions is also available to the purchaser.

You can tell I like The Graphics Magician. However, a fair evaluation should show both the strengths and weaknesses. In my estimation, the final Animation Editor of this program is a test of frustration. This is due to the lack of specific directions and examples.

My self-rating of computer expertise fell to a rank two after glancing at the manual that clearly states "...easy to use for the beginning programmer...." There are no sample run examples to guide the user through the shoals.

Perhaps software manufacturers will someday gather together to set a standard based upon agreed definition for terms such as beginner or most advanced. This excellent program would be improved by documentation based upon computer expertise.

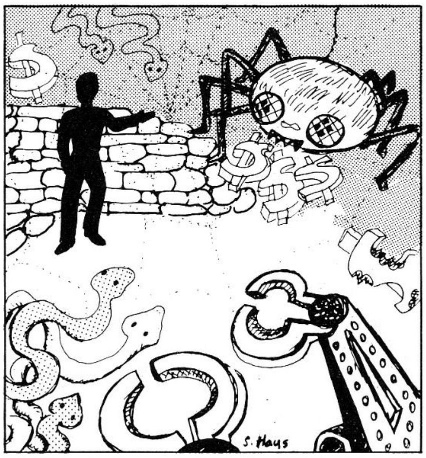
As it is, The Graphics Magician has given me more than my money's worth in new things to learn about developing hi-res graphics with my computer. ■

Bob Blaske
Hobe Sound, FL

Editors note: Penguin Software has recently released a new version of the Graphics Magician that includes a new manual. According to Penguin, the new version has simplified editing procedures.

Money Munchers

Are you ready for another maze game? Well, take your standard Pac-Man-type game, remove the power pellets, create a random maze at each level, replace the dots with dollar signs and the ghosts with three new types of monsters, and you're left with Money Munchers.



The object of the game is to amass as many dollars as possible while avoiding contact with the monsters. The monsters are indestructible; and if one touches you, you're dead.

You begin each game with a little man that you guide through the maze with a joystick or with one of two possible key groupings on your Apple keyboard. As each maze is successfully cleared, you are awarded an additional man. If, however, you fail to clear the initial maze, the game ends. In all cases, the game is over when you run out of men.

Each of the monsters has its own special characteristics. In the first maze, you encounter the money munchers who resemble robotic pliers. The goal of the money munchers is to eat the dollar signs before you reach them. They are not particularly interested in catching you; they move blindly through the maze in search of dollars.

In the second maze, spiders are added. The spiders guard the dollars; after you have cleared most of the maze, they home in on small groups of dollars and make them very difficult to capture.

In the fourth and succeeding mazes, snakes join the money munchers and spiders. The snakes do not care about dollars. They want to eliminate you! Since you have no defenses or weapons, your only recourse is to move faster than the monsters. The game responds well to the keyboard controls. Most of the men will be captured in the maze's blind alleys or when surrounded by two or more of the monsters. Beyond the fourth maze, difficulty is *not* increased by adding more or faster monsters. The program simply generates more difficult mazes with more dead ends.

Like most current games, Money Munchers lets you freeze the action at any time. At the start of each new maze, you are treated to a brief rendition of "We're in the Money." Each time you are captured by a monster, you'll hear a musical version of a childhood taunt (nyah, nyah, nyah, nyah, nyah). It's cute at first, but it wears thin very quickly. Also, I don't know how they did it, but Money Munchers is the loudest game I've ever played. Thankfully, the sound toggle works well.

The graphics are well done. The various monsters are realistic and show much fine detail. The display switches to a low-resolution kaleidoscope while each new maze is being created. The program occasionally prints a message of encouragement if you are doing well.

Technically, it's a good game. But is it fun? Yes. . . in the early mazes. Much of the fun involves striving for and obtaining higher scores. Once the snakes appear though, it's generally all downhill. It takes several lives to complete most mazes at this point unless you are very lucky. In addition, practice doesn't seem to help much. After an afternoon of play, any score above a certain level seemed to be more a matter of luck than skill. In the final analysis, I found the game to be more frustrating than fun.

Money Munchers is distributed by Datamost, 9748 Cozycroft Ave., Chatsworth, CA 91311. Price is $29.95. ■

Steven Schwartz
Pittsburgh, PA

DataFax

Some programs are so easy to use that hardly any documentation is needed; but you face limits in what you can do. Others have a fat manual that you must dip into even to get started; but once you've learned the system, it becomes second nature. DataFax, a database program from Link Systems, is definitely in the latter category.

After booting up DataFax it is wise to immediately back up the master disk. It took me three tries before I successfully created a back-up disk that would work. When I finally was able to back up one disk, I tried for a second with no success. This might indicate a need to check out my drive speed and/or alignment, as the DataFax back-up routine verifies the back-up disk after copying.

At any rate, I had no trouble creating data disks. It is hard to get an idea of how much can fit on one disk. The manual, however, indicates that if you tend to fill them up, you are using too small a storage system. Why not go to a hard disk! This little hint precedes a section on how to split a full database onto two or more disks. The method seems somewhat clumsy, but it could be worse. After all, if you do go out and buy a hard disk, the same data files can be put back together on it.

DataFax stores data in "folders" that are filed under one or more keywords, which you may choose as you go along. These keywords are automatically alphabetized, or in the case of dates in the MM/DD/YY format, put in chronological order. This is important later on, when doing searches through the folders.

Screen editing is well thought out, with control keys that move the cursor word by word, up, down and sideways, tab over to the next tab stop (every five spaces), delete or add a line (with buffer to remember the last line deleted, in case you want to retrieve it), add or delete a character, delete to end of screen, or mark a keyword. If you don't like the set of control keys given, you may decide to change them to match other software, such as your favorite word processor. The reference card contains the default set of editor keys, with space for you to write in your changes.

Searches come in two flavors: scan and examine. Scan mode comes up with a list of first lines of records containing the keywords, which can then be examined and edited. Examine mode brings up the first page of the first folder, then successive folders, one at a time.

DataFax offers a variety of ways to search. As mentioned before, the keywords are put in order as they are chosen; you may search for a range of keywords, such as all those beginning with C, all folders entered between 4/10/83 and 4/20/83, or all those with keywords having the word CAT in them—even in the middle of the keyword. You may also use the Boolean AND and OR operators, with parentheses, to find any of a combination of keywords. When one search is completed and you want to find only a subset of that group, a special keyword, LAST, can be used with other keywords, so that a search may be successively refined with a minimum of extra typing.

The same search facility can be used to select folders to be printed, with an option to review each folder on the screen before deciding to print it. This option also allows for printing single sheets of paper as opposed to form feed printing. And everything I've just said is applicable for deleting folders.

Printing does not have to include the whole folder; options exist to choose which lines will be printed, how many pages of each folder to print, whether page and folder separators and blank lines are to be printed, and how many blank lines are to be printed between pages and folders. The keywords may be printed, if desired, to the right of the folder printouts.

I had a brief chance to try out DataFax on an 80-column system. That's really the way to go! The pages are still 40-column, but on the right side there is the list of keywords for that folder. With 40 columns, going between the current folder and its list of keywords requires extra keystrokes; you can't see them both at once.

The use of templates is discussed and discouraged. It wastes time and space, because the template needs to be stored with each folder that uses it. If a search is to be done only for authors, prefixing their names with an identifier, such as A., is suggested, using those names then as keywords.

Sorting of any subset is possible as well. Two keyword ranges are needed, one for the group to be selected and the other for a range whose order is important. The example given is SORT TOYOTA OR CHEVETTE BY 1978. . . 1982, where the result would be a set of folders of those cars in order by year.

There are many hints and suggestions about using DataFax more efficiently. Even now, as I scan through the manual I discover new applications. It reminds me of the year or two I took to find all the idiosyncrasies of my word processor!

The different levels of menus (and DataFax is completely menu-driven) are confusing. The Command Tree is a diagram included as part of the reference card and is best ignored until you know enough not to be scared by its complexity. It took me a while to anticipate just what the screen will look like after any command.

Inputting the data is the most time consuming part of the process. Between saving one folder and starting a new one, there is an interval of disk spinning (both master and data disks) that may take five seconds or more—long enough to be irritating to the impatient, but still generally quite fast. Both the folder and the keywords associated with it must be saved to disk each time. This does allow for fast access, later, however.

Multiple pages require a bit more disk spinning. You need to call for another menu, and then DataFax stores page 1 and goes on to the next page. This is time consuming and, in general, is not recommended. When examining folders, only the first page will come up. The second page must be loaded again through the page menu.

DataFax uses Pascal. The user need not know anything about the language, but the program is very fussy about peripherals. Slot 0 must have a language card or a 16K RAM card. The disk controller card must be in slot 6 (although drives in slots 4 and 5 may be used). Slot 3 must contain either an 80-column card or nothing. A printer interface card is optional for slot 1 and a telecommunications card or modem is optional for slot 2. The tutorial advises removing any 80-column card for the purposes of learning the system, although it can make use of the larger display. Once you are within those specifications, you may begin.

DataFax is versatile, it searches quickly, and it is well documented. Its biggest plus is the versatility of searches. In the office it could fill a multitude of needs: cataloging for inventory or suppliers, researching, or mailing a list with individual information files. Also, it could be used for project organization or for anything where a diverse set of information must be linked together.

DataFax is produced by Link Systems, 1640 19th St., Santa Monica, CA 90504. The suggested retail price for the Apple II, II Plus and IIe format is $199; for the Apple III format, $249. ■

Tobi Hoffman
Ashland, MA

Apple DOS Software

Accent
Graphic Solutions $121.88

Applied Software Technology
Versa Form $297.88
Versa Form/HD $379.88

Artsci
Magic Window $83.88
Magic Mailer $61.88
Magic Words $61.88
Magic Pkg. Combo $176.88
Magic Window II $120.88

Broderbund
Payroll $301.88
Receivables $301.88

Context
The Connector $142.88

Continental Software
Home Accountant $64.88
1st Class Mail $64.88
CPA #1-GL $194.88
CPA #2-AR $194.88
CPA #3-AP $194.88
CPA #4-Payroll $194.88
CPA #5-Proprty Mngmnt $376.88

Cybertronics
Karel the Robot $188.88

Dakin 5
Programming Aid $74.88
Depreciation Planner $299.88
Budget Planner $119.88
Business Bookkeeping $299.88

Datamost
R.E.I.P. $105.88
Write-On $105.88
Personal Ledger $105.88
Market Technician $105.88
Property Manager $227.88

Datasoft
BASIC Compiler $83.88
LISP Interpreter $101.88

Denver Software
Pascal Tutor $105.88
Pascal Programmer $228.88
Financial Partner $194.88

Don't Ask
S.A.M. Mouth $101.88

Eagle
Money Decisions $156.88

Hayden Software
Pie Writer $120.88
Pie Writer/Multi-80 $120.88

DC Hayes
Terminal Program $82.88

Insoft
TransForth II $101.88
ALD System II $64.88
GraForth II $64.88
MatheMagic $75.88
GraphMagic $75.88
Prism $24.88
Mathe/Graph Magic $120.88

IUS
Pro EasyWriter $138.88
Pro EasyMailer $120.88
Pro EasyWriter/Mailer $231.88
EasyWriter/Mailer $116.88
EasyMover $83.88

Datadex $120.88
Datadex/HD $231.88
TellStar II $68.88

Kensingtom
Format II $120.88

Link Systems
Datallex $156.88
Datalink $83.88
Link Index $153.88
Link Video $49.88
Link Sampler I $53.88
Link Disk $60.88

LJK
Letter Perfect $120.88
Data Perfect $89.88
Edit 6502 $89.88

Lotus
Executive Briefing $156.88

Microcom
Microterminal $72.88

Micro Lab
Invoice Factory $157.88
Merger $46.88
Learning System $120.88
VisiBlend $46.88
VisiFactory $64.88
Asset Manager $157.88
Data Factory 5.0 $231.88
Wall Streeter $231.88
Payroll Manager $231.88

Microsoft
Applesoft Compiler $138.88
ADIOS muMath muSimp-8 $194.88
Time Manager $120.88
Multi-Plan (Apple DOS) $213.88

Micro-Sparc
TeleText $68.88

Muse
Data Plot $53.88
Address Book $46.88
Form Letter $83.88
Super Text Pro $82.88
Super Text Home $101.88

PBL
Personal Investment $116.88

Penguin
Complete Graphics $61.88
Special Effects $38.88
Graphics Magic $53.88
Fonts/Characters $24.88

Sierra/On-Line
ScreenWriter II $105.88
Dictionary $83.88
Pro Combo Pak $157.88
LISA 2.5 $68.88
LISA Educational $98.88
E.P.F. IV $68.88
General Manager $179.88

Silicon Valley
Word Handler $163.88
Dictionary $102.88
List Handler $75.88

SOF/SYS
Exec. Secretary $194.88
Exec. Speller $64.88

Softech
UCSD P-system $472.88
BASIC & RunTime $176.88
Xenofile $46.88
Softeach $101.88

Software Publishing
PFS: Report $101.88
PFS:File $101.88
PFS: Graph $101.88

STC
Pro Time & Billing $301.88
Reading Development $138.88
Apartment Manager $301.88
Inventory $213.88
IFO $157.88
The Creator $157.88

Southeastern
Data Capture $57.88
Data Capture 80 $76.88

Southwestern
Speed Star $83.88
Correspondent $53.88
ASCII Express $105.88
P Term Pro $105.88
Double Time Printer $83.88
Financial Mng. $57.88

Stoneware
D.B. Master $178.88
D.B. Master Pak #1 $82.88
D.B. Master (HD) $378.88
D.B. Master Pak #2 $82.88
D.B. Master Stat Pak $82.88
Electronic Price Sheet $85.88
Graphics Process $150.88

Sublogic
Graphics Combo $98.88

Systems Plus
General Ledger $287.88
GL/AP/AR $626.88
GL/AP/AR/Invent $780.88

T & F Software
PMP 2000 $171.88
Secure-a-Text $46.88

Terrapin
Terrapin LOGO $287.88

T.M.Q. Software
File Fax $119.88

VisiCorp
Desktop Plan $211.88
VisiPlot $171.88
VisiTrend/Plot $251.88
VisiDex $211.88
VisiTerm $90.88
VisiCalc 3.3 $211.88
VisiFile $211.88
Business Forcasting Model $90.88

Winner's Circle
Disk Encryption $53.88

Apple CP/M® Software

(Requires Z80 Softcard)

Ashton-Tate
dBase II (Z80) $529.88
FPL(Z80) $529.88

CommSoft
Roots/M (Z80) $154.88

CommShare
Target Calc(Z80) $47.88
Master Planner(Z80) $251.88
Application Pkg(Z80) $47.88
Planner Calc Pkg(Z80) $77.88

Digital Marketing
Milestone $691.88

Digital Research
CBASIC $121.88
CB-80 $380.88
Pascal/MT+ $269.88
Speed Programming $158.88

DJR Associates
FMS-80 (Z80) $689.88
FMS-81 (Z80) $288.88
FMS-82 (Z80) $288.88

Fox & Geller
Quick Code (Z80) $229.88
Dutil (Z80) $84.88

MicroPro
WordStar (Z80) $316.88
SpellStar (Z80) $180.88
MailMerge (Z80) $164.88
CalcStar (Z80) $109.88
DataStar (Z80) $210.88
SuperSort (Z80) $180.88
WS Customization $346.88
Word/Mail/Spell $584.88
WordStar/MailMerge $448.88
Word/Spell Star $448.88

MicroSoft
FORTRAN-80 (Z80) $155.88
A.L.D.S (Z80) $103.88
BASIC-80 Compiler (Z80) $303.88
COBOL-80 (Z80) $565.88
muMath/muSimp (Z80) $195.88
muLisp/muStar (Z80) $158.88
M-Sort 80 (Z80) $155.88
Multi-Plan (Z80) $214.88

Peachtree
Series 9
Peachtext (Z80) $350.88
Proof reader (Z80) $214.88
PeachCalc (Z80) $265.88
Mail List Mng. (Z80) $265.88
Telecommunications (Z80) $265.88
PeackPak 9 (Z80) $349.88
Series 40
GL (Z80) $282.88
AP (Z80) $282.88
AR (Z80) $282.88
Inventory (Z80) $282.88
Payables (Z80) $282.88
PeachPak 40 (Z80) $350.88
Series 80
GL (Z80) $282.88
AP (Z80) $282.88
AR (Z80) $282.88
Inventory (Z80) $282.88
Payables (Z80) $282.88
PeachPak 80 (Z80) $414.88
Spelling Proof (Z80) $213.88
Blacks Dic. (Z80) $112.88
Stedmans Dic. (Z80) $112.88

Prompt Doc
Manual Maker (Z80) $194.88

Quality Software
GBS (Z80) $492.88

Quic-N-Easi Products
Quic-N-Easi AG (Z80) $210.88
Quic-N-Easi PRO (Z80) $278.88

Select
Select WP (Z80) $377.88
Teach/M (Z80) $61.88

Sorcim
SuperCalc (Z80) $229.88
SpellGuard (Z80) $155.88

Southwestern
Z-Term (Z80) $84.88
Z-Term PRO (Z80) $121.88

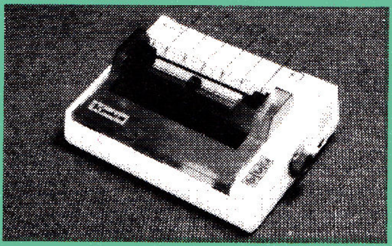
Taurus
CP+ (Z80) $121.88

HIGH TECHNOLOGY AT AFFORDABLE PRICES

THE BOTTOM LINE

MILFORD, NH 03055-0423 □ TELEPHONE (603) 881-9855

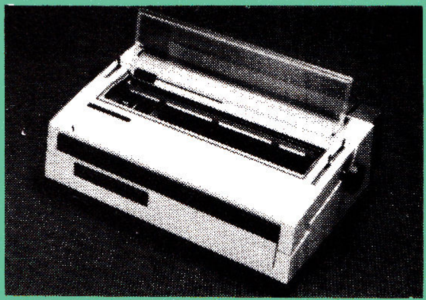
Star Micronics
GEMINI-10



$339.88 UPS DELIVERED

- 100 characters per second, bi-directional, logic-seeking, w/2.2K buffer
- 5 fonts, w/italics, double width, emphasis & proportional spacing
- Subscripts, superscripts, underlining, backspace, emphasis & double strike
- 120 x 144 dots/inch hi-res graphics, 6 x 6 block graphics

Silver Reed
EXP-550

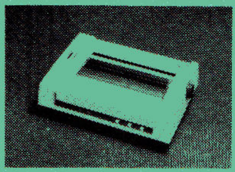


$719.88 UPS DELIVERED

- True letter-quality with Diablo code compatability
- 16 characters per second, 132 columns, bi-directional
- Subscripts, superscripts, underlining, backspace,
- Parallel interface standard

PRINTERS

C. Itoh



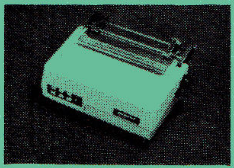
C. Itoh Prowriter $399.88
w/RS-232C $609.88
C. Itoh Prowriter 2 .... $734.88
w/RS-232C $789.88
C.Itoh F-10 Starwriter, 40 cps
Parallel or RS-232C $1379.88
C Itoh F-10 Printmaster, 55 cps
Parallel or RS-232C $1679.88
F-10 Series Tractor $289.88

Diablo
Diablo 620 .... $1029.88
Diablo 630 .... $1969.88
Diablo 630 Tractor $289.88

Epson
FX-80 .... $CALL
FX-80 F/T .... $CALL
MX-80 .... $CALL
MX-80 F/T .... $CALL
MX-100 .... $CALL
Com-Riter (Letter Quality)
Parallel or RS-232C .... $CALL

IDS
IDS Prism 80 .... $1079.88
above w/graphics $1139.88
above w/sheetfeed $1249.88
above w/4-color $1539.88
IDS Prism 132 .... $1229.88
above w/graphics $1309.88
above w/sheetfeed $1419.88
above w/4-color $1699.88
IDS Microprism $574.88

Mannesmann-Tally

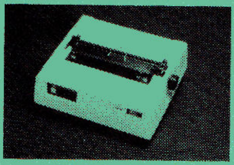


MT-160-L $789.88

NEC
NEC 3530 .... $1759.88
NEC 7730 .... $2399.88

PRINTERS

Okidata
Microline 82A $419.88
82A Okigraph ROM $49.88
82A Tractor $59.88
82A Roll Paper Holder $49.88
Microline 83A $679.88
83A Okigraph ROM $49.88
Apple Okigraph II $59.88
Microline 84 w/graphics & tractor
Parallel, 200 cps $1024.88
Microline 92 .... $524.88

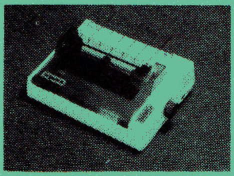


Microline 93 .... $884.88

Qume
Qume Sprint 11+ $1539.88

Smith-Corona
Smith Corona TP-1 .... $519.88
Specify either 10 or 12 cpi,
& parallel or RS-232C interface

Star Micronics



Gemini 10 .... $339.88
Gemini 15 .... $499.88

PRINTERS

Printer Interfaces
PKASO "Smart" Interfaces
Print/graphic interface for Apple & Franklin centronics parallel output. Specify C.Itoh, Centronics, Epson, IDS*, NEC or Okidata printers.
PKASO card & cable $139.88
*IDS Color card & cable $159.88

Parallel Card w/cable $89.88

Orange Micro
Grappler + ... $149.88
Bufferboard (16K) $149.88

Accessories
2-way Printer Switch
Parallel or RS-232C $119.88
Trace Soundtrap $119.88
8K Microfazer $154.88
64K Microfazer $219.88
128K Microfazer $319.88

CALL FOR PRICES on Cannon, Centronics, Panasonic, Ricoh, Silver Reed, & others

MONITORS

USI
USI Pi-2 (12" green) $159.88
USI Pi-3 (12" amber) $189.88
USI Pi-4 (9" amber) $159.88

Amdek
300G (12" green) $149.88
300A (12" amber) $159.88
Color I (Comp.) $319.88
Color II (Hi-res RGB) $649.88
Color III (RGB) $379.88
Apple RGB Interface $159.88

Electrohome
1302-1 (Medium RGB) $319.88
1302-2 (Hi-res RGB) $599.88
Super Color Option $209.88

Zenith
ZVM 121 (12" green) $139.88

MODEMS

DC Hayes
Hayes Micromodem II w/Terminal Software $319.88
Hayes 300 Baud $224.88
Hayes 1200 Baud $539.88

Novation
Apple Cat 300 .... $319.88
Apple Cat 1200 .... $609.88
Apple 1200bd Upgrade $349.88
103 SmartCat $199.88
103/212 SmartCat $499.88
RS-232C Card $99.88
Multi-Line Junction Box $39.88

PERIPHERALS

Kensington
System Saver Fan $79.88

MicroSoft
Premium Package—A 16K RAMCard, Z80 SoftCard, Videx VideoTerm w/soft video switch, & the Users Guide $474.88
Z80 & Videx $444.88
Z80 SoftCard $229.88
16K RAMcard $74.88

Mountain Computer
CPS MultiFunction $164.88
RAMplus + 16K $159.88
RAMplus + 32K $179.88
ROMplus + ... $139.88
ROMwriter Card $154.88
5mB Hard Disk $2364.88
10mB Hard Disk $2829.88

Rana Systems
Elite 1 Disk Drive $289.88
Disk Controller $104.88
Elite 2 Disk Drive $474.88
Elite 3 Disk Drive $614.88

Saturn Systems
32K RAM Board $179.88
64K RAM Board $274.88
128K RAM Board $389.88

SSM
AIO Multi-Function Card $199.88

Videx
VideoTerm $204.88
VideTerm Combo $234.88
Enhancer II $109.88

Vista
A-800 Disk Controller for 8" disks $399.88
Vision 80 Card $269.88

Information & Orders
(603) 881-9855
Orders Only: (800) 343-0726
No Hidden Charges

FREE UPS shipping on all orders—No extra charge to use credit cards—COD orders accepted ($10 fee added)—All products shipped factory fresh with full warranty (we are authorized for warranty work on many products—just ask)—No purchase orders, foreign or APO orders accepted—Minimum $50 per order—Personal checks take 3 weeks to clear—This ad prepared in April.

MasterCard VISA

HIGH TECHNOLOGY AT AFFORDABLE PRICES
THE BOTTOM LINE

AMERICAN EXPRESS ®

MILFORD, NH 03055-0423 □ TELEPHONE (603) 881-9855

New Software

edited by Tom Woods

Public Domain Software User Group Programs

CP/M Users Group and SIG/M Users Group programs are now available on Apple 5¼-inch disks from Elliam Associates, Suite 47, 24000 Bessemer St., Woodland Hills, CA 91367.

Many programs are accompanied by document files of instructions. Games include Adventure and Startrek. There are utilities such as directory and file catalogers, disassemblers and format converters. Communications routines include file transfer protocols for use with modems as well as bulletin board supervisory programs for establishing your own RBBS.

Elliam sells the material in three ways: by the user group disk number, by category or by "custom- building" disks for you with a selection of specific user group files. Reader Service number 459.

Mr. Krypto

Mr. Krypto is a program of word games designed for use by people of all ages. Unscramble topsy-turvy words, decode words and unravel zany sayings and famous quotations. Easy or more challenging options can be selected. Contact Micro Program Designs, 5440 Crestline Rd., Wilmington, DE 19808. Price is $29.50. Reader Service number 440.

Take a Risk!

Actuarial Microcomputer Software, 3915 A Valley Court, Winston-Salem, NC 27106, offers the Risk Simulator, a program that uses the Monte Carlo method to estimate probability distributions associated with various risk situations. Applications range from the estimation of automobile maintenance expenses to employer self-funding of health care benefits. It runs on the Apple II Plus and IIe. Price is $187.50. Reader Service number 441.

Comsail I

Comsail I simulates a fast, 30-foot sailboat sailing in near identical conditions around a plotted course. The player's performance is rated with a numerical score. Text displays show boat speed, compass heading and elapsed time. All standard sailing dynamics are present in the program, including tacking into the wind, luffing and critical angle. Tips on sailing are given and help is available if the sailor runs into trouble. Contact Single Source Solution, 2699 Clayton Rd., Concord, CA 94519. Price is $59.95. Reader Service number 442.

Bat-Stat

Rainbow Computing Inc., 19517 Business Center Dr., Northridge, CA 91324, offers Bat-Stat, a program that keeps track of cumulative batting statistics for each player on a baseball or softball team.

Player and team statistics are given for the current game and for the season. Ten statistical categories are provided: At Bats, Runs, Hits, Batting Average, Doubles, Triples, Home Runs, Sacrifices, Walks, and Runs Batted In. Bat-Stat automatically computes batting averages. It runs on the Apple II Plus and IIe. Price is $49.95. Reader Service number 443.

Magic Crayon

Magic Crayon allows children to draw pictures and designs on the Apple II Plus and IIe computer's low-resolution graphics screen.

Drawing, color selection and other options are controlled with keystroke commands. Once a picture has been created, the young artist can store and redraw it at another time. The user makes decisions and gives the computer instructions. He/she can "teach" the computer to draw pictures and then watch as it reproduces on screen the pictures they have created.

Magic Crayon is recommended for preschool to fourth grade children. Contact C&C Software, 5713 Kentford Circle, Wichita, KS 67220. Price is $35. Reader Service number 446.

Crime Wave

In Crime Wave, a new game from Penguin Software, 830 4th Ave., Geneva, IL 60134, lawlessness is rampant in the Big City. The minions of crime have been rambunctious of late, and the police force has its hands full coping with the rampage. With killer machines, the criminals can destroy the forces of law and order. Hop into your blue Cruiser and round up the bank robbers before they get their Robot Rammers onto the streets and turn orderly traffic into a demolition derby. Price is $19.95. Reader Service number 444.

Hey Diddle Diddle

Hey Diddle Diddle, a collection of 30 classic nursery rhymes, is offered by Spinnaker Software, 215 First Street, Cambridge, MA 02142.

Hey Diddle Diddle is designed to help children understand how words and rhymes create poetry. It lets children take fragmented thoughts and rearrange them into coherent verse.

There are three different ways for children to play and learn. For pre-readers, Storytime is a continuous display of 30 rhymes with pictures and music. Storybook lets beginning readers see each rhyme formed in slow motion with words and pictures. For more advanced readers, Rhymegame presents two levels of play. The child can choose to unscramble either the first four lines or the entire eight-line rhyme. Price is $29.95. Reader Service number 445.

Quadrilaterals

A new program from Reader's Digest Services Inc., Pleasantville, NY 10570, combines the sounds and graphics of a video game with textbook quality instruction that introduces students to the basic properties of quadrilaterals.

The Quadrilaterals program is composed of 15 instructional units that cover all aspects of quadrilaterals—from types (squares, rectangles, parallelograms and trapezoids) and parts (vertices, sides, angles and diagonals) to related mathematical theorems. The program uses action illustrations to demonstrate different geometric concepts covered in the individual instructional units. Price is $34.95. Reader Service number 447.

Graph 'N' Calc

Desktop Computers Software, 303 Potrero St., Santa Cruz, CA 95060, offers a new Apple III version of Graph 'N' Calc, a graphics and statistical analysis program. Graph 'N' Calc allows Apple III users to prepare a variety of stacked and side-by-side bar charts, line charts, pie charts and combined line and bar charts.

The program includes a set of commonly used statistical and financial calculations. Complex calculations such as exponential smoothing, linear regression, net present value and internal rate of return are executed with a single keystroke. Price is $249. Reader Service number 449.

Get the Idea

Getting the Main Idea is an educational game for the Apple II/II Plus that takes players around the world. As players pass each city, a graphic depicting an aspect of the city is displayed. When players land on designated spaces, they are given the opportunity to read a selection and give an answer.

In Context Clues, a player attempts to uncover a hidden treasure by winding in and out of paths leading to four different treasure sites. When landing on designated spots, the player gains a chance to define a word taken from a short reading selection. Both games are $49.95. Contact Learning Well, 200 South Service Rd., Roslyn Heights, NY 11577. Reader Service number 448.

Stock Analyzer

N-Squared Computing, 5318 Forest Ridge Rd., Silverton, OR 97381, offers the Stock Analyzer. The program incorporates major technical analysis functions for a detailed study of individual stocks.

Capabilities include high-low-close-volume displays with linear or semilog auto-scaling, price-volume indicators, relative strengths, momentums, smoothing, averaging and point and figure charting. Price is $295. Reader Service number 451.

Flex-I-Term

Flex-I-Term is a telecommunications package that performs terminal support functions.

The first module—Supermodem—is a machine language terminal program that operates the terminal during time sharing operations. Supermodem may be modified for a variety of interactive environments through the use of Configurator, the second module. There are standard menu selections for interfacing with CompuServe. The program integrates video, keyboard and modem output devices. Contact Single Source Solution, 2699 Clayton Rd., Concord, CA 94519. Price is $99.95. Reader Service number 450.

Roll-a-File

Roll-a-File is a program that stores, retrieves and searches an electronic card file. Designed for salesmen, businessmen and other professionals who want to keep track of their clients on an electronic filing system, it allows the user to search for clients according to criteria such as name, age, account number and zip code.

Print-a-File is an adjunct to Roll-a-File. It allows the user to print mailing lists and phone lists using the same selection techniques as Roll-a-File.

Both programs run on the Apple II and are priced at $79 and $99, respectively. Contact Sophisticated Software, 650 Foothill Blvd., La Canada, CA 91011. Reader Service number 452.

CodeWriter

CodeWriter allows Apple II Plus and IIe users to design their own application programs without knowledge of computer programming. It automatically translates English into machine language.

Applications that can be developed by CodeWriter include payables and receivables, mailing lists, invoicing and inventory and production tracking. Price is $399. Contact Dynatech Microsoftware Inc., 7847 N. Caldwell Ave., Niles, IL 60648. Reader Service number 453.

Attendance System

Attendance Reporting System II provides reports that account for student absenteeism and tardiness.

It can accommodate a school of any size and provides eight user-defined absence categories such as illness, truancy and so on. Enrollment/withdrawal statistics are computed. Reports include the Daily Attendance Report, The Student History Report, and a wide variety of homeroom summary reports. Price is $600. Contact School Office Software Systems, 3408 Dover Road, Durham, NC 27707. Reader Service number 455.

Disk Fixer

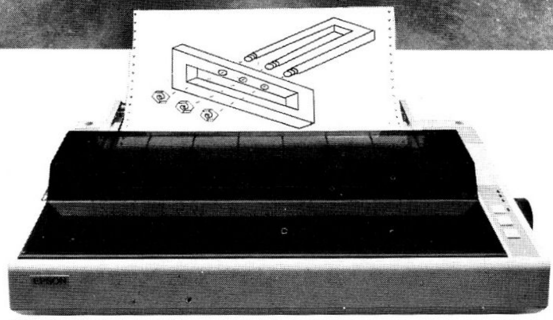
The Disk Fixer is a utility for manipulating, repairing and protecting data stored on disks. The full screen editor allows the user to examine and change any portion of a disk, correct the space usage within files and lock out bad tracks on disks. Directories may be alphabetized for easy location of desired files. The display and search capabilities show where specific hex or ASCII data is located. The user can modify any data including binary files.

Disk Fixer can also be used to resurrect a deleted file, if that file has not been written over. The program is written in machine code for fast operation. Disk Fixer runs on the Apple II and IIe and is priced at $49.95. Contact RCI Marketing, 19517 Business Center Drive, Northridge, CA 91324. Reader Service number 456.

Accounting System

CMA Micro Computer, 55722 Santa Fe Trail, Yucca Valley, CA 92284, offers a new general ledger and payroll system for the Apple IIe, Success Desk Accounting System.

The system features a user-definable 200 account general ledger with a payroll element. The payroll element can handle weekly, bi-weekly, semi-monthly and monthly payroll with up to ten user-definable deduction fields. Current federal tax and Social Security tables are furnished, and federal law changes can be handled by user tax table changes. Three tax tables can be programmed for state or local taxes. Reports include a payroll journal, monthly deposit reports, quarterly audit reports and annual payroll re-caps. Price is $349.95. Reader Service number 454.



ACTUAL APPLE II PRINTOUT USING GRAPPLER AND EPSON MX100.

COMPLETELY REDESIGNED.
NOW, THE GRAPPLER + .

The original Grappler was the first graphics interface to give you hi-res screen dumps from your keyboard. The new Grappler + adds flexibility with on-board printer selection and 23 different commands for text and graphics. Exclusive *Dual Hi-Res Graphics* allow a side-by-side printout of graphics pages 1 and 2.

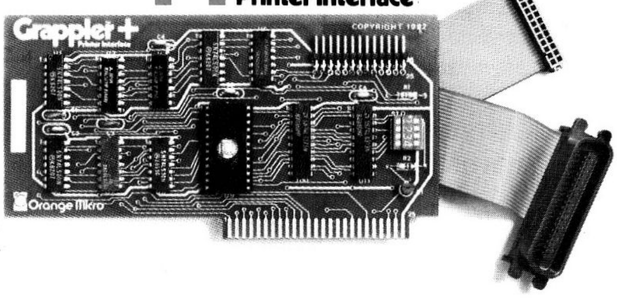
The Grappler + is compatible with the Apple II, II + , IIe and III* computers. Its extensive printer menu has been expanded to include the Apple Dot Matrix, Okidata 84 and Star Gemini, along with most popular printers. In addition, the IDS Grappler + is currently available with color capability, including color graphics screen dumps.

UP TO 64K BUFFER OPTION

An optional Bufferboard can now be added to all existing Grappler and Grappler + interfaces. See your Apple Dealer for details.

*Requires additional software driver.

With The Grappler™ + Printer Interface



NOW AVAILABLE THE BUFFERED GRAPPLER +

The best of both worlds . . . the Buffered Grappler +

All of the popular Grappler + features with the time-saving benefits of a printer buffer.

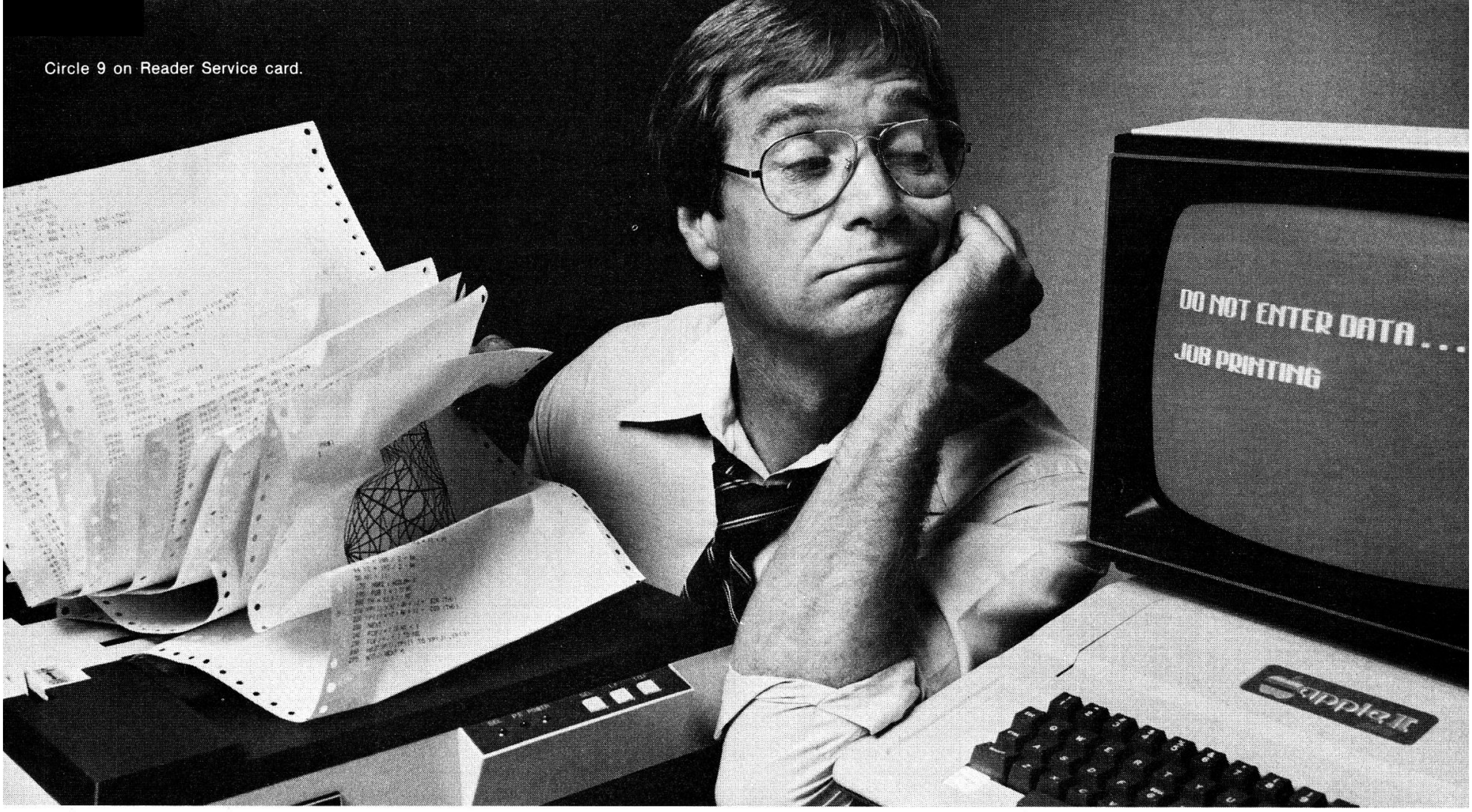
- 16K of Buffer
- Expandable to 64K
- Interfaces with all popular matrix printers

Make the most of your Apple and printer, with the Grappler + or Buffered Grappler + .

Orange Micro Inc.

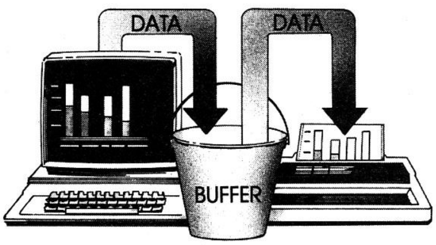
1400 N. Lakeview Ave.,
Anaheim, CA 92807 U.S.A.
(714) 779-2772 Telex: 183511 CSM
Foreign Dealer Inquiries Welcome

Circle 9 on Reader Service card.



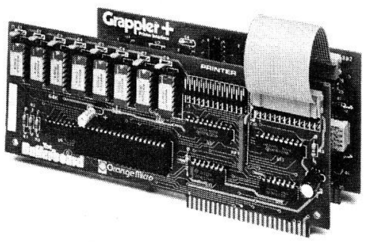
If your printer uses your Apple® more than you do, you need The Bufferboard.™

If your Apple is locked into the "PRINT" mode so much that you've taken up solitaire to kill the boredom, you need a buffer. And if your computer is the Apple II or III, the only buffer for you is The Bufferboard. Expandable to 64K of storage, The Bufferboard stores an instantaneous **bucketful** of print data from your computer. Then it feeds the data to your printer at its own printing rate. Your Apple is set free from driving your printer and is ready for more data from you.



Take your existing interface—and buffer it!

Only The Bufferboard has a simple Interface-Docking System. No bulky boxes or expensive power supplies are needed because The Bufferboard fits right into your Apple—and **docks** onto your existing printer interface. The result is convenient



and economical buffering of most popular printer interfaces, including the Grappler + ™ interface, Epson interface, and Apple printer interface. Thirty seconds and a single hook-up are all you need to end the printer waiting game forever.

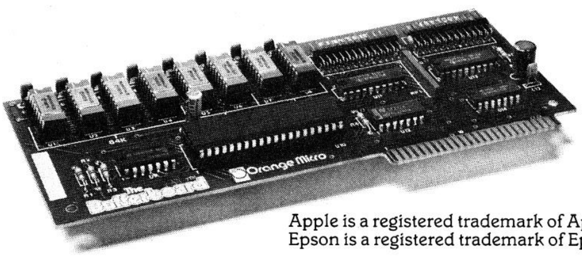
Up to 20 letter-size pages stored at a time.

The Bufferboard comes standard with 16K, and is expandable to 32K or 64K of buffering capacity with the addition of memory chips. This "bucket" will hold up to 20 pages of a print job, allowing you freedom to use your Apple.

The Bufferboard—designed exclusively for the Apple Computer.

Specifications:
• Versions for Grappler + interface, Epson interface, Apple interface, and other popular printer interfaces • 16K buffer standard • Upgradeable to 32K or 64K • Automatic memory configuration • Automatic self test • Includes interface docking cable.

The Bufferboard is made by Orange Micro, Inc.; the same people who brought you the popular Grappler + printer interface. Both the Grappler + and The Bufferboard are now available at your local Apple dealer.



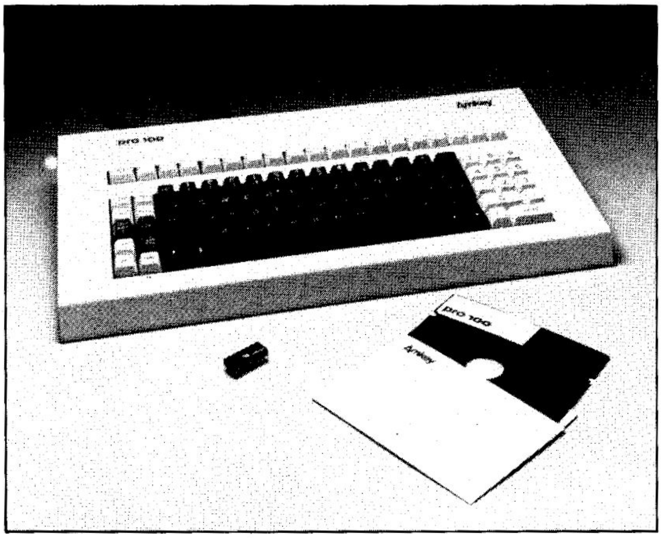
Orange Micro inc.
1400 N. Lakeview, Anaheim, CA 92807
U.S.A. (714) 779-2772
TELEX: 183511 CSMA

The Bufferboard™
For Apples and Printers

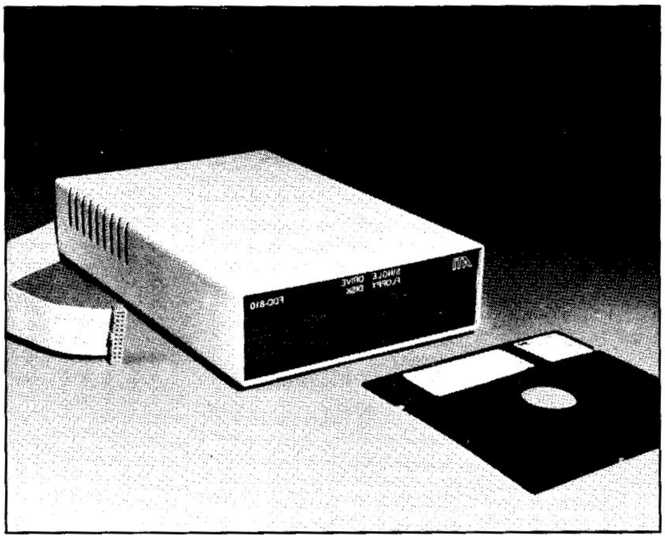
Apple is a registered trademark of Apple, Inc.
Epson is a registered trademark of Epson America, Inc.

New Products

edited by Tom Woods



The Pro-100 detachable keyboard from Amkey Inc.



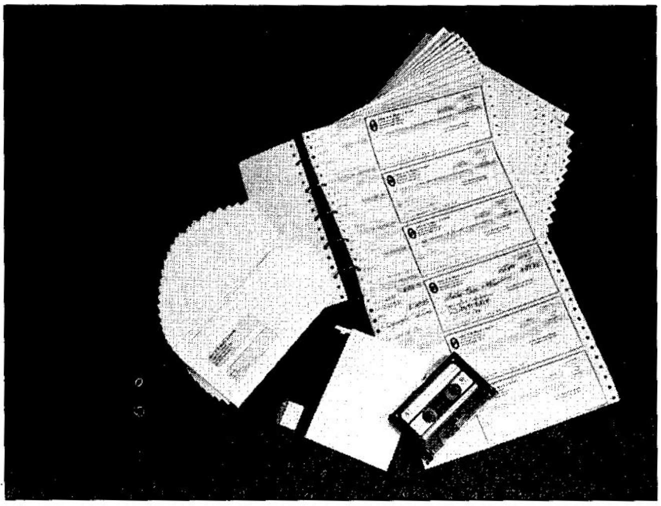
The FD-810 from Algol Technology Inc.

Detachable Keyboard

Amkey Inc., 220 Ballardvale St., Wilmington, MA 01887, offers the Pro-100, a detachable keyboard for the Apple II/II Plus.

The Pro-100 offers 100 keys supporting all existing Apple functions. It includes horizontal and vertical cursor movement, separate number pad with enter key, auto-key, power-on indicator, 22 VisiCalc keys and 25 Apple Basic keys.

The VisiCalc keys help the user access VisiCalc commands such as /SL, /SS, /C, /DR, / – and /SQ. The Apple Basic keys help the user access Basic commands. Supplied with the Pro-100 is a pre-boot disk containing Basic disk utilities and a word processing and graphics program that demonstrates the programming flexibility of the function keys. The keyboard and software is priced at $265. Reader Service number 460.



Fan-fold checks from Synergistic Software.

Disk Drive

The FDD-810 disk drive features a rear-panel, locked switch that overrides the write-protect notch on disks. The switch allows use of the reverse side of disks. It comes with a cable and connector for direct plug-in to the Apple. Price is $350. Contact Algol Technology Inc., 303-3 Convention Way, Redwood City, CA 94063. Reader Service number 478.

Fan-fold Checks

Synergistic Solutions, 4715 Shepherd Rd., Mulberry, FL 33860, offers fan-fold checks and accessories designed for use in the home or small cottage business.

The checks are 9½ inches wide and are fan-folded three to a page. They can be used on any tractor, pin or friction feed printer capable of printing 10 characters per inch, 80 columns per line. Accessories include a data ring checkbook for storage and dual window envelopes. A package of 200 checks, checkbook, dual window envelopes and a cassette version of the Checkbook-Checkwriter II program is priced at $59.95. Reader Service number 474.

Video Board

A video board that provides the Apple IIe with RGB video signals that enhance the resolution and color quality of the supplied composite video is available from Telemax Inc., 780 Lorraine Dr., Box 339, Warrington, PA 18976.

The VCB-2e can be used with 80-column text so that color graphics and text are displayed on one RGB monitor. Price is $169. Reader Service number 475.

eRAM

Quadram Corporation, 4357 Park Drive, Norcross, GA 30093, offers eRAM-80, a peripheral card designed to double the

amount of text that can be displayed on the Apple IIe.

With eRAM 80 the display screen can be programmed for either standard 40-column or extended 80-column text display. It allows IIe users to switch between the two formats. It adds 40 extra characters to the IIe's regular 40-character format, allowing display of 80 characters per line. Price is $159. Reader Service number 470.

SooperSpooler

The SooperSpooler printer buffer contains 16K bytes of memory, one parallel input port and one parallel output port. Up to 127 character spaces can be converted to one byte to increase memory space for columnar documents.

Pages can be formatted to almost any parameters, and titles of up to 70 characters can be added to each page. Continuous readout of the amount of data stored—or the amount still to be printed—is indicated on the face of the SuperSpooler. The unit contains its own power supply and cabinet. Price is $349. Contact Compulink Corporation, 1840 Industrial Circle, Longmont, CO 80501. Reader Service number 476.

Grappler

Orange Micro Inc., 1400 N. Lakeview Ave., Anaheim, CA 92087, offers a new printer interface, the Buffered Grappler Plus.

Memory expansion on the Buffered Grappler Plus is achieved with the installation of additional memory chips. Two chips increase buffering to 32K; six chips can boost printer memory to 64K. At 64K, 20 pages of text can be stored for printing, leaving the Apple ready for its next task.

Graphics features, accessible by command codes, include 90 degree rotation, double sizing and inverted printing. Text features include screen dumping, margin and page length setting, and wordaround. The Buffered Grappler Plus Mixed Mode Screen Dump allows graphics and text to be printed together. It is compatible with most dot matrix printers. Price is $175. Reader Service number 477.

P1 Printer

NEC Information Systems, 5 Militia Drive, Lexington, MA 02173, offers the PinWriter P1 dot matrix printer. The P1 is capable of printing at 180 characters per second in a high speed mode, and 90 characters per second in high density mode.

The P1 offers Centronics type interface; proportional spacing, bold face printing, elongated characters and continuous underline are standard features. Maximum print line length is 136 characters, and vertical line spacing can be set for three, four, six or eight lines per inch. Price is $700. Reader Service number 472.

APPIC/G

SSM Microcomputer Inc., 2190 Paragon Dr., San Jose, CA 95131, offers a parallel printer interface, the APPIC/G, for the Apple II and IIe. APPIC/G includes an APPIC interface card, cable and Graph-It graphics software.

The APPIC parallel interface works with any standard parallel printer. With Graph-It, the user can print graphs, charts, equations, grids, logos and hi-res graphics. The APPIC/G package is $129. Reader Service number 463.

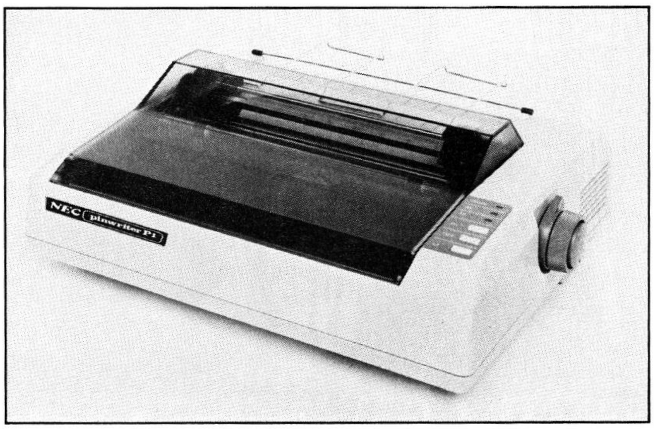
EPS Keyboard

The EPS Detachable Keyboard features 12 special function keys that give up to 48 commands when used with EPS PROMWARE Modules. Each module comes with a special function label strip that identifies the commands programmed into the special function keys. The keyboard has a complete word processing layout with travel key switches and multi-function edit keys. Price is $399. Contact Executive Peripheral Systems Inc., 800 San Antonio Rd., Palo Alto, CA 94303. Reader Service number 479.

Arcade Board

The Arcade Board is a peripheral board that upgrades the color graphics and sound generating ca-

The SooperSpooler intelligent printer interface from Compulink Corporation.



The Pinwriter P1 Printer from NEC Information Systems Inc.


ANOTHER DISCOVERY FOR
orangeers
THE ORANGEPEEL™
POWERFUL. EXPANDABLE.
AT $790, REMARKABLE.
Yesterday, buying a computer meant compromising. Powerful units carried powerful prices. Affordable units were little more than toys.
Today, there's the ORANGEPEEL. The powerful, expandable, highly versatile microcomputer priced at a remarkable $790.
• FULLY APPLE II® COMPATIBLE.
• LANGUAGE FLEXIBILITY — Modular design permits the use of all disc based Apple II software, CP/M,® UCSD-p.*
• MULTIPLE CPU's (as available).
• 76K RAM.
• DISC BASED — requiring standard Apple II drive.
• DETACHED KEYBOARD — Full ASCII upper and lower case.
• HIGH AND LOW RESOLUTION B/W GRAPHICS.
• 3 I/O SLOTS EXPANDABLE TO 7.
• ORANGE BUS FOR INTERNAL EXPANDABILITY.
• 6 MONTH REPLACEMENT WARRANTY — (Extended one year optional).
THE ORANGEPEEL COMPUTER.
We anticipate the future.
See your local ORANGE dealer today.
U.S.A. Distributor
Compusource Inc.
3112 Hennepin Avenue South
Minneapolis, Minn. 55408
612-827-2951
orange computers inc
orange computers inc.
10-1495 bonhill road
mississauga, ontario l5t 1m2
(416) 677-7782
*Apple II is a registered trademark of Apple Computers Inc.
*CP/M is a trademark of Digital Research Inc.
*UCSD-p is a trademark of The Regents of The University of California.
Circle 153 on Reader Service card.
Available in Canada, USA, Hong Kong, Singapore, Australia and Japan.

The EPS Detachable Keyboard features 12 special function keys.

pabilities of the Apple II. Some coin operated arcade machines (i.e., Moon Patrol, Star Gate and Donkey Kong) use the same video and sound chips that are used in the Arcade Board. The Arcade Board can be programmed entirely in Basic as well as machine language. Contact Third Millennium Engineering Corporation, 1015 Gayley Ave., Suite 394, Los Angeles, CA 90024. Price is $250. Reader Service number 480.

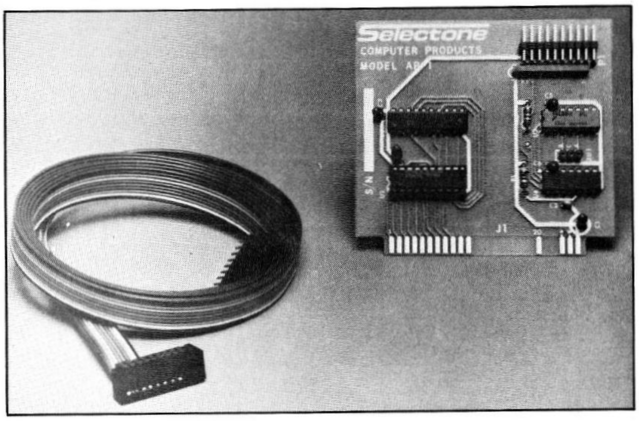
Selectone Interface

Selectone Corporation, Computer Products Division, 28301 Industrial Blvd., Hayward, CA 94545, offers the AP-1 printer interface card for the Apple II Plus.

The AP-1, designed for use with the Selectone Micro-Typer System, features standard Centronics parallel protocol, including a cable assembly and connector for direct connection to the Micro-Typer interface kit. The AP-1 plugs directly into any one of the peripheral slots of the II Plus. Price is $95. Reader Service number 471.

Word Processing

In *Word Processing with your Microcomputer*, an introduction to word processing, readers learn how



The AP-1 printer interface from Selectone Corporation.

HUNTINGTON COMPUTING

#155	16K Ramboard (limited supply)	**$39.99**
#64	ATV RF Modulator	**$29.99**

All CMS at a discount
All Compucovers at a discount
All Hartley at a discount
All Leraning Co. at a discount
All Math City at a discount
All Sensible at a discount

#3257	Alpha Plot	**$34.69**
#3054	Apple Barrel II	**$17.69**
#8982	Apple Key Chain	**$5.00**
#8980	Apple Lapel Pin	**$3.00**
#3259	Apple Mechanic	**$24.99**
#1600	Apple Necklace	**$10.00**
#1963	Apple Panic	**$25.39**
#1668	Apple Spice	**$26.89**
#3901	Apple Writer Extended	**$29.69**
#154	Arcade Adapter	**$35.59**
#1967	Arcade Machine	**$49.99**
#1220	Atlantis	**$33.99**
#2005	Bag of Tricks	**$33.89**
#1960	Bank Street Writer	**$50.89**
#3251	Beagle Bag	**$24.99**
#1068	Beer Run	**$19.99**
#2010	Beneath Apple Dos book	**$16.89**
#2003	Beneath Apple Manor	**$25.39**
#1223	Bolo	**$29.69**
#921	Bomb Alley	**$50.89**
#9601	Bug Byter	**$40.29**
#7960	Business Pac 100	**$89.99**
#1458	Carverns of Freitag	**$25.39**
#8360	Ceemac	**$63.69**
#8999	Child's Play	**$19.99**
#9860	Chinese Lessons	**$26.29**
#707	City Dies at Night, A	**$25.39**
#9423	Comparative Shopping	**$26.29**
#7980	Compu Poem	**$15.95**
#1917	Comput-Math Algebra	**$33.89**
#9014	Computer Almanac ... SALE	**$17.99**
#1763	Crimewave	**$14.99**
#1222	Crisis Mountain	**$29.69**
#7140	Cross Clues (SRA)	**$29.69**
#6380	Crossword Magic	**$39.99**
#1720	D.A.R.K.	**$29.69**
#1607	DB Master	**$179.00**
#1108	Dark Crystal	**$33.89**
#7234	Delta Drawing	**$50.89**
#1126	Dictionary, The	**$79.99**
#7155	Diet Planning	**$21.19**
#4000	Disk Storage box (90 disks)	**$19.99**
#6509	Dolch Vocabulary	**$33.89**
#3250	Doss Boss	**$20.39**
#6753	Dracula (educational)	**$29.69**
#9015	ESP	**$14.99**
#3652	Electric Duet	**$25.39**
#2026	Electricity	**$10.19**
#7233	Face Maker	**$29.69**
#6021	Family Roots	**$169.00**
#131	Fiberbilt Case for Apple	**$59.00**
#3252	Flex Text	**$25.39**
#68	Flip Sort disk holder	**$24.00**
#3260	Frame-Up	**$24.99**
#1106	French Mystery House	**$21.19**
#1131	Frogger	**$29.69**
#9017	GGA Lower Case Chip	**$14.99**
#992	Galactic Adventure	**$50.89**
#9600	Game Show	**$34.29**
#9008	Generic Games	**$9.99**
#33	Gibson Light Pen	**$296.59**

Softlights

By Barbara Huntington

This ad should break in July so we'll talk about children's programs in honor of our daughter Melody's 4th birthday.

We really are getting good stuff for kids — **Sticky Bear ABC, Sticky Bear Numbers, Sticky Bear Bop** and **Old Ironsides,** all from Xerox are **$33.89** each.

We also have the prestigious Plato series from Control Data. Each one is **$50.99** (list is $60). Some titles are **French Vocabulary, German, Spanish, Computer Literacy, Fractions, Whole Numbers, Basic Number Facts, Decimals** and **Physics-Elementary Mechanics.**

We had a holdup on the documentation for **Child's Play,** (We've got our daughter, Melody, on the cover, naturally.) but by the time you read this it will finally be out. Melody loves the little ant that runs through a maze in one of the programs. We're moving with the trend for lower software prices so Child's Play is only **$19.99.** It's great for preschoolers.

Also for preschoolers is Software Production's **Alpha Beasts and Co.** (ABC) (#7291) and Versa's **Alphabet Squares** (#7953), each at **$25.39.** ABC has beautiful graphics. Alphabet Squares teaches beginning letters with a little face that moves. When your child moves the face onto the picture of an Indian for the letter I, the picture fills the whole screen. If the wrong answer is chosen, nothing happens — only positive reinforcement. We carry Spinnaker, Learning Company, DCM and many other educational products. For older kids, Harcourt-Brace has a good **SAT** program we sell for **$59.39** (#7080).

PROGRAMMERS

Fred likes the program **Busywork** from Datum Consultants. It takes some of the drudgery out of programming by offering ready-to-use subroutines for your own program. Writing input routines, checking input, centering titles, writing title pages and menus are included. It has a 60-page booklet, a demo and gobs of good subroutines. #8400 for **$35.99.** Lets hear it for Dale Ludwig, the author.

TAKING A STAND

Need a good stand that matches your Apple, holds drives, monitor and leaves room for your Super Fan? We have the **(#34) Micro Stand 2.2** with solid steel construction still for only **$38.99** plus shipping. (We only charge $2.00 shipping for **software** anywhere in the U.S., but we gotta charge more for hardware.)

Betty, Barbara, Becky, Sally, Pat, Madge, and Fred say "Hi!"

—Barb H.

#9543	Integer Fix	**$19.99**
#7975	Investment Decisions	**$84.99**
#0000	Invoice Factory	**$99.99**
#6100	J&S Gradebook	**$29.29**
#87	Joystick, Kraft	**$55.19**
#6280	Kennedy Flight Planner	**$43.89**
#2151	Krell MIT LOGO	**$76.39**
#8774	Learning to Understand Card Catalog	**$13.99**
#3350	Letter Perfect (special)	**$99.89**
#6022	Lineages	**$89.00**
#8261	Lovers and Strangers	**$25.39**
#1135	Lunar Leep (special)	**$19.99**

#2700	MCE Buying Wisely	**$38.19**
#1866	MECC Historic Simulations	**$38.19**
#6001	MX-100 Ribbon	**$9.99**
#6000	MX-80 Ribbon	**$4.99**
#6001	MX-80 Ribbon	**$4.99**
#3391	Mastertype	**$33.89**
#9380	Menu Generator	**$29.99**
#2150	Micro Deutsch	**$152.89**
#7550	Microaccountant	**$42.39**
#1206	Microbe	**$38.89**
#1640	Micro Cookbook	**$33.89**
#1866	Miner 2049	**$33.99**
#6450	Mother Goose Rhymes	**$22.39**
#2108	Multiplan	**$199.00**
#2662	Munch-a-bug	**$42.39**
#9411	Nutrition	**$26.29**
#8984	Pacman Ghost Pin	**$2.50**
#1760	Pie Man	**$14.99**
#1164	Pie Writer 80 col.	**$127.39**
#3401	Pinball Construction Set	**$33.89**
#6710	Population	**$21.09**
#7650	Pornopoly	**$25.39**
#9061	Prism	**$16.89**
#3249	Pronto Doss	**$24.99**
#923	RDF 1985	**$33.89**
#8440	Rana Drive	**$289.00**
#9500	Recipe Handler	**$35.09**
#1068	Repton	**$33.89**
#1504	Rescue at Rigel (special)	**$17.99**
#1421	Roadblock	**$25.39**
#1475	Robot War	**$33.99**
#300	S.C. Macroassembler	**$68.00**
#1150	Sargon II	**$29.69**
#1105	Screen Writer II	**$99.99**
#1667	Sea Dragon	**$29.69**
#1974	Sea Fox	**$25.39**
#2908	Sensible Speller	**$106.19**
#7006	Sentence Diagramming	**$21.19**
#1971	Serpentine	**$29.69**
#2184	Sherwood Forest	**$29.69**
#9019	Shift Key Mod	**$9.99**
#1064	Sneakers	**$25.39**
#7230	Snooper Troops #1	**$38.19**
#7231	Snooper Troops #2	**$38.19**
#1121	Soft Porn Adventure	**$25.39**
#1780	Space Cadette	**$29.69**
#6452	Spanish Hangman	**$26.99**
#3654	Spider Raid	**$25.39**

HUNTINGTON COMPUTING
Post Office Box 1297
Corcoran, California 93212

Foreign Orders 209-992-4481
In California 800-692-4146

Outside Calif. 800-344-5106

We take **MasterCard, American Express or VISA** (Include card # and expiration date). California residents add 6% tax. Include $2.00 for postage. Foreign and hardware extra. Foreign (excluding Canada): remit U.S. currency, checks on U.S. banks, use listed charge cards, or make direct wire transfers through Security Pacific Bank, Corcoran, for a $6.00 charge. All overseas orders shipped by air. Send for free catalog. Prices subject to change without notice.

microcomputers can be used to type business letters, keep track of mailing lists, help children with homework and keep a variety of records. Features standard to most word processors are discussed, and an overview of word processing software currently on the market is provided. Printers, special hardware and other supplies used in word processing are discussed. The book is $19.95 (hardback) and $13.95 (paperback). It is available at book stores or from Tab Books Inc., Blue Ridge Summit, PA 17214. Reader Service number 466.

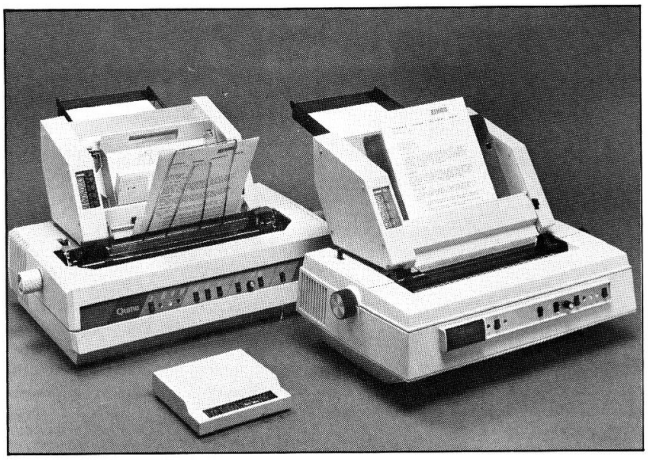
Paper Processors

ZIYAD Inc., 100 Ford Rd., Denville, NJ 07834, offers the Z-200 paper processor for the Qume 5 printer. The Z-200 feeds sheets of paper into serial RS-232C letter quality impact printers from either of two paper trays. The Z-200 feeds cut paper of variable lengths and widths. The paper tray hold up to 200 sheets of paper. A control panel provides audible and visual signals to the operator regarding paper movement, and a microprocessor based serial interface controller enables the user to access different baud rates and protocols.

ZIYAD Inc. also offers a Z-300 paper processor for the Qume 9 printer that provides all of the capabilities of the Z-200, as well as envelope feeding from an integral envelope tray. Price is $2195 for the Z-200 and $2695 for the Z-300. Reader Service number 465.

Starwriter

The Starwriter F-10 printer features 8 bit parallel or RS-232C interfaces, built-in IEEE 488 interface, optional 2K buffer, low-noise operation and standard friction feed with optional bidirectional tractor feed. Price is $1895. Contact Leading Edge Products Inc., 225 Turnpike St., Canton, MA 02021. Reader Service number 464.



The Z-300 and Z-200 paper processors atop a Qume Sprint 5 and Sprint 9 printers.

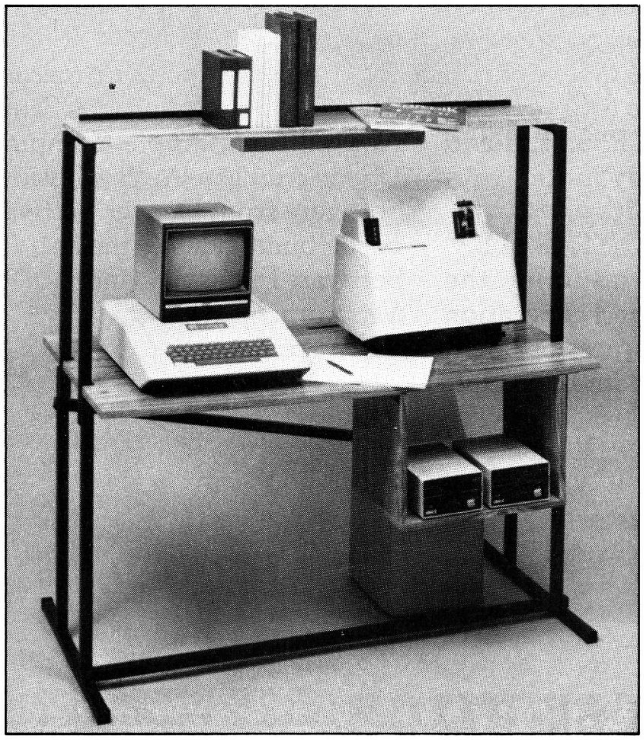
Power Supply

High Technology Software Inc., Box 60406, 1611 NW 23rd St., Oklahoma City, OK 73146, offers a reserve power supply for the Franklin Ace computer.

The RBS-DC supplies instantaneous backup power for the computer and its peripherals during power outages and brownouts. Enough time (1 hour 45 minutes) is available to allow uninterrupted computer operation through temporary power failures. Price is $325. Reader Service number 461.

Space Saver

Compco Industries Inc., 159 W. Walnut St., Painesville, OH 44077, offers a home-use computer table that organizes microcomputer disk drives, monitor, printer, keyboard and manuals into one area. Features include a built-in printer slot and built-in cord and ribbon slot for tangle-free wires. The Compu-Table is made of steel with woodgrain tops. Price is $53.95. Reader Service number 468.



The Compu-Table from Compco Industries Inc.

Taxan Interface

TSK Electronics Corporation, 18005 Cortney Court, City of Industry, CA 91748, offers the Taxan 410-80, an 80-column and RGB interface card.

The card connects all Taxan RGBvision monitors with the Apple IIe. It gives the user a third color cursor for the use in text editing; 16 color, 40-column foreground/background; 16 color LORES with option of mixing 40-column foreground/background text; and 16 color MERES with option of mixing 80-column switch selectable color text. Price is $199. Reader Service number 467.

Voice Input Module

The Voice Input Module converts spoken words to commands or data and is designed to allow voice input with any Apple software including database management, word processing, graphics, business, education and games. Application vocabularies such as Applesoft Basic, Pilot and Controller can be generated.

The Apple Voice Utility Software allows the user to build a vocabulary for specific application software. The user defines two character strings; the first is the spoken word or phrase to be understood by VIM, and the second interacts with the application program in response to a spoken word or phrase. The VIM 1 is designed for the Apple II and includes 2020C Printed Circuit Board, Voice Utility Program, DX-121 microphone and ribbon cables and connectors. Price is $845. Contact Voice Machine Communications Inc., 10522 Covington Circle, Villa Park, CA 92667. Reader Service number 473.

Circle 36 on Reader Service card.

FREE!

A six-pack of tasty Apple* Posters

Here's your chance to get any or all of these beautiful 16 x 24 posters Free! except for just a small postage/handling charge! Commissioned by Datamost, and painted by well-known computer artist, Art Huff, each is a unique experience in design that will grace your home or office.

To get your posters, visit your favorite computer store and pick up a coupon where you see the counter display shown below. Fill out the coupon and mail it to Datamost. For each individual poster you want Free, include a registration card from any Datamost software package. Send only $1.75 postage/handling fee with each order: (Note. Without a Datamost registration card, the Posters are available at only $5.95 each, plus same postage/handling charge.)

Be the first to collect the entire series of these magnificent, exclusive, and FREE Apple* posters from Datamost!

Look for this poster display at your computer store.

DATAMOST™ INC

8943 Fullbright Ave., Chatsworth, CA 91311. (213) 709-1202

YES! I Want To Participate in the Datamost

FREE APPLE* POSTER OFFER

Please send me the posters indicated below
at $5.95 each plus postage/handling ________
6½% sales tax for California residents ________
Postage/handling charge $1.75 $1.75
Circle poster number: 1 2 3 4 5 6 Total: ________

Name ________
Address ________
City ________ State ________ Zip ________
Master ________ Visa ________ No. ________

Send To: DATAMOST™ INC

8943 Fullbright Ave., Chatsworth, CA 91311 (213) 709-1202

*The posters are exclusive products of Datamost, Inc. and are not connected with, or authorized by Apple Computer, Inc. Apple is a trademark of Apple Computer, Inc.