INTERNATIONAL APPLE CORE™
presents

Apple Orchard

VOLUME 2 NUMBER 1 SPRING 1981 $3.50

The IAC is coming to Chicago in May!

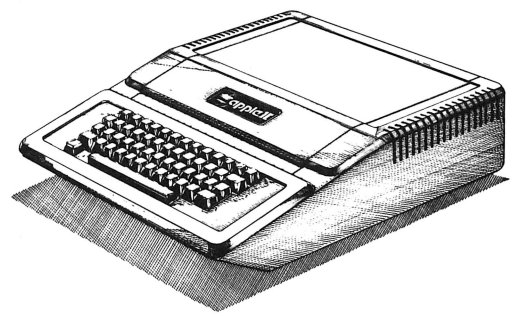
Chicago Convention and Tourism Bureau

specials

VISICALC	124.95	PANASONIC COLOR MONITOR	395.00
DESKTOP PLAN	165.00	CENTRONICS 737	795.00
APPLE TAX PLANNER	95.00	INTERFACE CABLE for 737	25.95
APPLE FORTRAN	159.95	MICROSOFT "RAMCARD"	167.50
MONTY PLAYS MONOPOLY [disk]	29.95	NEC GREEN PHOSPHOR MONITOR	229.95
GAMMON GAMBLER	20.00	VERSAWRITER - Graphics System	219.95

FREE WITH EACH PURCHASE - $400 IN LAS VEGAS VALUES

Apple II+ 48K FREE



— with purchase of 10 meg. CORVUS at Regular Price —

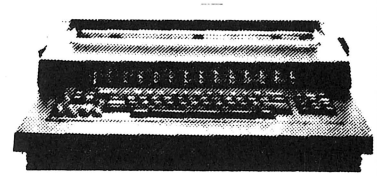
THE SOURCE
AMERICA'S INFORMATION UTILITY

+

D.C. Hayes Micromodem II™

398.00

NEW XYMEX 1000 with Quadra-Pitch 10, 12, 15 Pitch and Proportional Spacing



16K RAMS for $53.00

CORVUS 10 MEGABYTE HARD DISK

NOW! Corvus speaks Apple Pascal!

• Apple Plus (including Apple Pas... Fully compatible hardware/software ...byte disk: IMI-7710 • Proven W...nology • Z-80 based Corvus ... • Comprehensive disk diagn... to 4 disks per system • Syste... on disk 2990. Corvus offers ...stems solution to the mass stor... micro computers in a packag... a briefcase, we provide an inte... disk and personality module. Call ...day for additional information. Get up to s...d with Corvus.

10 MEG NOW IN STOCK
TAKING ORDERS FOR 5 MEG.

PRICE $4350.00

BASF DISKS
$25.95 for 10

APPLE DISK DRIVE
Reg. $525 now $459.95

LANGUAGE SYSTEM
$425.95

APPLE /// 128k
$3995.00

QUME PRINTER 5/45
$2495.00

WORD PROCESSING COMBO!

EasyWriter with a Videx Board only $450.00

Westwood Computer Systems

apple computer Sales and Service

Telephone Orders Only

(213) 475-5467

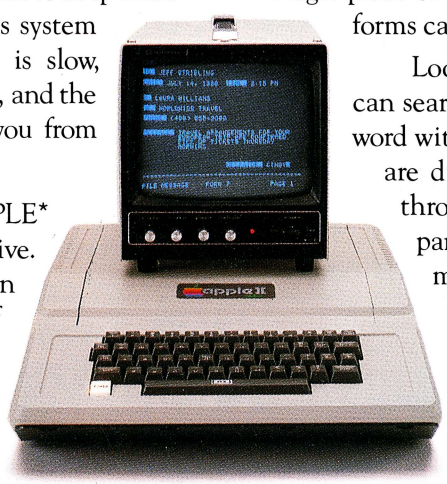
pfs:™ software

Who needs it?

You do.

For small amounts of information a pencil and paper filing system works great. But when you want to keep track of hundreds of pieces of information this system has limitations. Recording information is slow, locating what you want can be impossible, and the system is so time consuming it keeps you from filing information you know is valuable.

With PFS software and an APPLE* computer you have a powerful alternative. Using the concept of designing a form on the screen, PFS lets you create a file of information on any subject you wish without programming. You can catalog your stereo record collection, 35mm slides, magazine articles, daily expenses, or your club's membership list. Using PFS at work you can make better decisions by creating files on inventory, customers, or orders and accessing them in seconds.



requires a 48K, 16-sector disc based APPLE II system

To use PFS you simply design a form on the screen by typing the names of the items you want to store information about. Once the form is created you tab from item to item and fill in the information. This can be a single piece of data or several pages of text. Up to 1000 forms can be stored on a diskette.

Looking up information is just as easy. PFS can search for a number, a single data item, or a word within a page of text. All forms that match are displayed on the screen. You can browse through each one and change, delete, expand, or print it. PFS even has a print formatter that lets you create mailing labels.

PFS software is different. It is not a specialized application package or a complex programmer oriented data base manager. It is a personal filing system that lets you communicate with the computer using meaningful everyday concepts to rapidly create files on any subject matter you require.

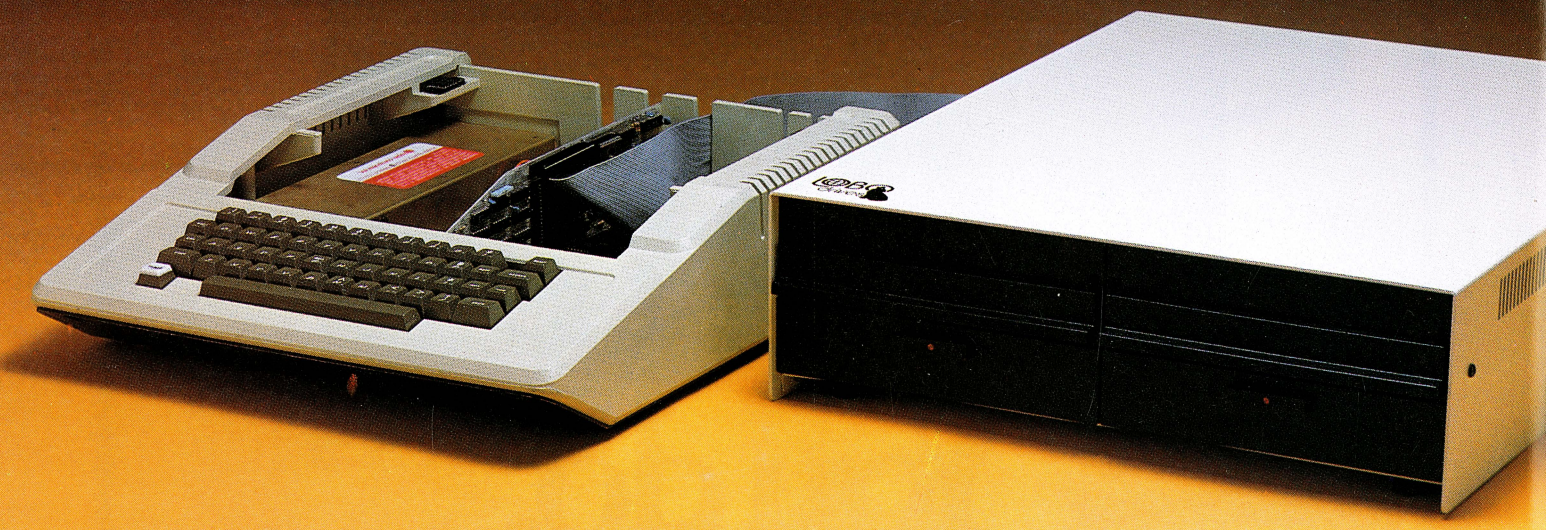
PFS is available through your local dealers. If they don't carry it, have them give us a call at (415) 368-7598 or write to us at Software Publishing Corporation, P.O. Box 50575, Palo Alto, CA 94303.

PFS is a trademark of Software Publishing Corporation.

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

Software Publishing Corporation

4.4 MEGABYTES FOR YOUR APPLE



Double-Sided, Double-Density 8-Inch Drive Capability is Here ... Now!

The new LOBO DRIVES Model LCA-22 Double Density Floppy Disk Controller has been specifically designed to match your APPLE® to the new double-sided, double-density 8-inch floppy disk drives. Now, you can add up to four 1.1 Megabyte drives (4.4 Megabytes total) and realize all the power and potential of your APPLE computer.

Completely Software transparent, the Model LCA-22 will plug into any chassis slot. You are no longer restricted to slot 7. And, the Model LCA-22 is fully compatible with 3.2. (3.3 DOS systems, PASCAL will be available soon.)

LOBO's new LCA-22 Disk Controller and full line of field-proven, high-reliability disk drives (all LOBO products come with a one year, 100% parts/labor warranty) are available at computer retail stores nationwide. Stop in and see a demonstration at your local dealer today.

LOBO drives INTERNATIONAL

LOBO DRIVES INT'L
354 South Fairview Ave.
Goleta, CA 93117
(805) 683-1576

I'm looking to expand my APPLE. Please send me more information on:

- ☐ Model LCA-22 Disk Controller
- ☐ Floppy Disk Drives
  - ☐ 5¼-inch
  - ☐ 8-inch
- ☐ Fixed Disk Drives
  - ☐ 5¼-inch
  - ☐ 8-inch
  - ☐ 14-inch

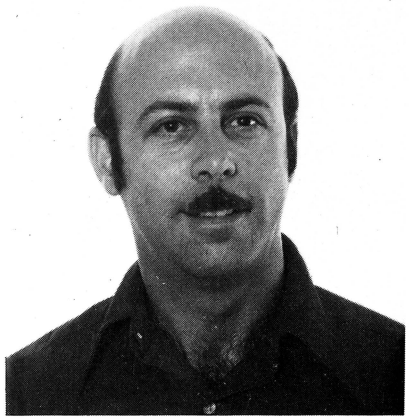
Name _____

Address _____

City _____ State _____ Zip _____

Phone _____
(area code)

PRESIDENT'S CORNER



In the past months the IAC has been receiving many phone calls and letters and the main question asked is **"What can the IAC do for my club?"**

As a member club, you receive APNOTES which Apple Computer, other manufacturers and members give the IAC for distribution. They contain information on hardware/software modifications/fixs and any other data that can help the end user with his/her Apple. As of this time, the IAC has sent over 140 pages of APNOTES.

As a member club, you also receive software disks that have been donated by clubs as far away as Japan and Australia, and as close as Apple Computer itself. So far there have been five official disks released by the IAC to its members, and we hope to have one a month starting in May. If you are a new member club, the IAC does not send back issues of the software, but commences sending the current disk at the time of joining. The software is given **FREE** to our member clubs.

The IAC publishes the **"Apple Orchard",** which is made available at a bulk discount rate to our member clubs. You can also individually subscribe. An additional monthly bulletin is also sent to our member clubs to inform them of late news and developments.

The IAC has committees dealing with all manner of special interests. You can communicate with these committees by writing and making use of their experts.

In the near future we hope to establish a WATS [800] line to answer some of your questions. There is in the planning stage a "scholarship" fund, and we will be video taping the seminars given at our annual meeting so we can send them to you for viewing. As the directors from each region receive *your* inputs we, the IAC, will act on them.

Now back to the question "What can the IAC do for my club?" I would like to change that to read **"What can you as a member club contribute to help the IAC?"** We need programs to distribute to the members, we need articles for the **Orchard,** we need your input. The IAC is a volunteer organization and without INPUT there is no OUTPUT.

Ken Silverman,
President,
International Apple Core

INTERNATIONAL APPLE CORE SPONSORING MEMBERS

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

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

A list of sponsoring members, current through the first of February, 1981, appears on page 68.

(continued on page 68)

INTERNATIONAL APPLE CORE™

PRESENTS

Apple Orchard

Vol. 2 No. 1 **Spring 1981**

Entire Contents Copyright© 1981
by **International Apple Core**

Published for the International Apple Core by
dilithium Press, P.O. Box 1493, Beaverton, OR 97075

Jill A. Scofield – Publisher

Val J. Golding – Editor
Kathryn Hallgrimson – Assistant to the Editor

Tymera Coen – Production Manager

Larry Danielson – Circulation

ADVERTISING REPRESENTATIVES
dilithium Press
P.O. Box 1493
Beaverton, OR 97075
(503) 646-2713

SUBSCRIPTIONS – DEALER INFORMATION
Apple Orchard Subscriptions
P.O. Box 1493
Beaverton, OR 97075
$10/year – Published Quarterly

For information on becoming a member of IAC
please write: International Apple Core
P.O. Box 976
Daly City, CA 94017

IN THIS ISSUE

President's Corner	Ken Silverman	3
IAC Sponsoring Members		3
Select One	The Editor	6
PRINT FRE(ed)	Val. J. Golding	7
Screen Formatting of Text	Peter C. Weiglin	9
An Introduction to Light Pens	Neil D. Lipson	14
Notes on Hi-Res Graphics Routines in Applesoft	C. K. Mesztenyi	17
An Apple II Quickie	Gordon Stallings	19
Practical Super Hires Graphics	R. H. Good	20
Comparing Applesoft Programs for Differences	Charles Boody, David W. Walker, Val J. Golding	22
2 Sided Disks		27
S. H. Lam Routine Utility	Lee Reynolds	28
Double-Size Graphics for the Silentype	Bruce F. Field	30
Matrix Functions with the Apple	Max J. Nareff	36
Programming the Apple to Communicate by Telephone	Terry Guerrant	40
Resurrecting Killed Programs	David G. McDonald	41
Lookit		43
A Note on CP/M	Steve Jenkins	43
Programmer's Aid RAM Test Initializer	Gary M. Comeau	44
Text Train and Sub Search	David B. Garson	45
Word of Caution	D. Laden	45
Getting There Faster in Applesoft Basic	David Bartley	46
Modifying Pascal BIOS to Work with the Paymar LC Adapter	Craig Vaughan, Lee Meador	47
Simple Functions in Applesoft	D. B. Buchler	47
Homebrew to Champagne	Steve Wozniak	51
The Apple II Cassette Interface		57
***SYNTAX ERR		59
New Product Parade	Mark Crosby	61
Application for Membership		66
Sorting in BASIC	Art Mack	68
IAC Membership Information		72
IAC Miscellany	Joe Budge	73
Who was Asking About ASCII?	Val. J. Golding	74
Passing Argument Values to Machine Language Subroutines in Applesoft	C. K. Mesztenyi	77
Low Resolution Graphics in Pascal	Bill Shepard	80
Color Compliments for Black and White	Tom Jacobsen	86
Green Apple Bits		86
Tune Your Apple	Mark Welty	86
Proper Printer Protocol	Val J. Golding	91
Single Drive Copy	Steve Adams	92
Advertiser's Index		95

INTERNATIONAL APPLE CORE

Officers

Jerry Vitt	Chairman	(214) 369-7660
Ken Silverman	President	(415) 878-9171
Michael Weinstock	Vice-President	(516) 360-0988
Dave Gordon	Treasurer	(213) 954-0240
Joe Budge	Secretary	(919) 489-4284

Regional Directors

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

International Directors

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

Committee Chairmen

Apnotes	John Shanes	(804) 746-2711
Apple Fair	David Alpert	(312) 295-6078
Apple Orchard	Val J. Golding	(206) 932-6588
Constitution & Bylaws	Ken Silverman	(415) 878-9171
Education SIG	Ted Perry	(916) 961-7776
Ham Radio SIG	James E. Hassler, WB7TRQ	(307) 632-4934
Handicapped SIG	David McFarling	(420) 467-1878
I.A.C. Software	Neil Lipson	(215) 356-6183
Legal SIG	Butch Clayton	(803) 884-5370
Medical SIG	Dr. Larry L Stoneburner	(714) 953-9151
Newsletter Exchange	David Alpert	(312) 295-6078
Newsletter Library	Major Terry N. Taylor	(213) 372-4134
New Club Assistance	Randy Fields	(415) 775-7965
Standards	Mark Robbins	(303) 750-5813
Telecommunications	Craig Vaughn	(703) 255-2241

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

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

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

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

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

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

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

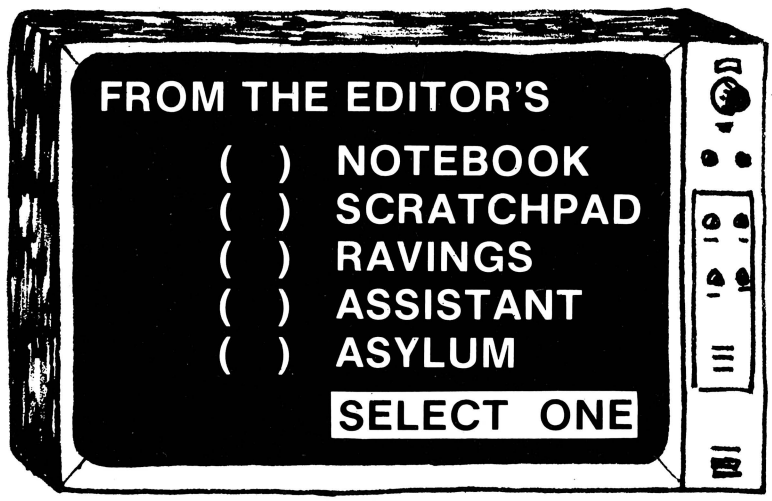
Complete information? It's at your dealer's now. Or, we'll send it to you and include a dealer list. Write us. Call us. Or, circle the reader service card number below.

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

MICROSOFT
CONSUMER PRODUCTS

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

CIRCLE READER SERVICE NUMBER



By the Editor

It seems incredible that an entire year has rolled past since we put the first Orchard to bed and celebrated the new arrival, but it has and we are. Celebrating the arrival of the fourth Orchard, 96 pages of fact and information filled articles and programs.

Leading off this anniversary issue on Page 9 is **Peter C. Weiglin's** *Screen Formatting of Text,* a well written article from the San Francisco Apple Core's *Cider Press,* and of special significance to newcomers to Appledom who want to learn more about the Apple's easy to use screen display commands.

Our second major contribution is by **C. K. Mesztenyi** of the Washington, D.C. *Apple Pi* group, who has greatly expanded on the Applesoft Hi-Res Internals information from the first Orchard, with his *Notes on Hi-Res Graphics Routines in Applesoft.* This feature makes it truly possible for the first time to utilize the speed of Applesoft's Hi-Res routines from Assembly Language.

One of *Call -A.P.P.L.E.'s* bestowals is a substantial rewrite by **David W. Walker** and ourselves of **Charles Boody's** *Comparing Applesoft Programs for differences,* a program that is helpful to the utmost when you can't remember exactly what changes you have made to the program you are working on.

Call -A.P.P.L.E. and Cider Press each appear on the scene again with **Lee Reynolds'** *S. H. Lam Routine Utility,* a handy program that converts machine language to BASIC, and **Max Nareff's** thorough exploration of *Matrix Functions with the Apple.*

Have you ever wondered what ASCII really means? Our explanatory article offers some clues as to how the ASCII standard character set and Apple display set differ, while Washington Apple Pi also puts in another appearance with **Bruce Field's** *Double Size Graphics for the Silentype* and **C. K. Mesztenyi's** really neat method of passing parameters to the A, X, Y and PC registers from BASIC.

Pascal users should not feel neglected either, with a passel of goodies, including *Low Resolution Graphics in Pascal* by **Bill Shepard,** nor is CP/M overlooded, as **Steve Jenkins** contributes *A Note on CP/M.*

In Apple Computer, Inc.'s *Contact,* co-founder **Stever Wozniak** describes some of the events preceeding Apple I and the Apple][, under the title *Homebrew to Champagne*. "Woz" was recently injured while taking off in his aircraft, but is doing very well. We are certain any cards would be well received, and may be sent c/o Apple Computer, Inc., 10260 Bandley Driver, Cupertino, CA 95014.

Finally, in the *International Apple Core* section, you will find the first of several new features, **Mark Crosby's** *New Products Parade,* along with information as to how you, as an individual, can subscribe to the Orchard, or locate a user group near you to join. User groups, in turn, will find how they may join the I.A.C.

Don't forget, we are just scratching the surface. It would take another two or three pages just to describe all the fine material under these covers; best you look for yourself. To make sure you don't miss out on our next big batch of brilliant booleanisms, better buy a subscription. (whew!)

EDITOR RESIGNS, CLUB IN UPROAR

PRINT FRE(ed)

By Val J. Golding

Got your attention, there, didn't I! So commenced a story by Lee Meador in the Ft. Worth Apple User's Group Newsletter, and griping about much the same situation as we are about to espouse. Contributions.

It is not the almighty dollar we are talking about, but the almighty article. This issue of the *Apple Orchard* would not have at all been possible without the efforts of three of the largest user groups in the country: San Francisco Apple Core (Cider Press), Washington, D.C. Apple Pi (Washington Apple Pi) and Apple Pugetsound Program Library Exchange (Call -A.P.P.L.E.). To these three organizations, then, we must first extend our thanks and our congratulations for coming through in a pinch.

It seems inconceivable that the member clubs of the International Apple Core, now numbering well over 200, with just minor exceptions, submitted not so much as one original or reprint story for this issue of the *Orchard*. We have read many, but not enough, issues of the club newsletters to know that there is a wealth of material out there that would be of much interest to many of the readers of the *Orchard*.

If the president of each club, or the editor of its newsletter, would take just a few moments to review material they have recently published and recommend it to us, we would have enough material for the next year or two of the *Apple Orchard*.

The IAC is an organization that provides a multitude of services to its member clubs, not the least of which is free public domain software. In return, each member club has a responsibility, an obligation, to reciprocate in the form of material for the *Orchard*. The *Orchard* was, and remains, the primary source of financial support for the IAC. Without its revenues, the IAC could not survive.

This editorial is no idle threat; it is a statement of fact. We will bandy no words. Your support is needed, and it is needed now.

IN MEMORIAM:

Apple Orchard
Vol. II, No. 2
Summer, 1981

After you play the Temple of Apshai, you can play Sticks and Stones for free.

Within the 200 rooms and catacombs of the Temple of Apshai, untold treasures await you — the hero. All you have to do is elude, outsmart and outwit the beasts, monsters and demons lurking in the dark labyrinth. Spend minutes or hours on this role-playing fantasy — the boldest computer game in our Dunjonquest™ series.

Now, when you order the "Temple of Apshai," you get the "Sticks & Stones" board game for no extra charge. In fact, if you're not satisfied with the "Temple of Apshai," you can return it within 10 days and still keep "Sticks & Stones!"

But don't wait, this special offer is limited. (We'll also send you a catalog outlining our other exciting computer games).

Automated Simulations, P.O. Box 4247, 1988 Leghorn Street
Mountain View, California 94040 Department AO

Please send me the "Temple of Apshai" for:

	Cassette ($24.95)	Disk ($29.95)
TRS-80	☐ 16K, Level II	☐ 32K TRSDOS
APPLE	Not available	☐ 48K Applesoft in ROM
PET	☐ 32K	Not available

(Add $1.00 shipping and handling charge; plus 6% or 6½% tax for California residents.)

Name _____

Address _____

City, State, Zip _____

☐ Check enclosed. Charge to: ☐ VISA ☐ MasterCard

Amount $_____ # _____ Expiration date_____

Or charge by phone: (800) 824-7888, operator 861. In California: (800) 852-7777, operator 861. If you prefer, call these numbers for a list of the computer stores near you.

Verbatim comments:

"Compared to the brands I've used before, Verbatim Datalife™ is the best yet!"
Sandy Tiedeman
Las Vegas, NV

"New Verbatim helped eliminate I/O errors on my Apple."
Richard Adams
Ft. Walton Beach, FL

"I would prefer Verbatim Datalife over any brand I've ever used."
Skip Piltz
Overland Park, KS

"My experience with Verbatim diskettes has been excellent. I've used several boxes over the past few months and they've all been error-free."
Robert Roeder
Las Vegas, NV

"Much stronger, better centering. A definite improvement."
Leroy LaBalle
Marrero, LA

"It has worked perfectly everytime!"
Richard Ruth
Shippensburg, PA

"I like the thicker protective cover."
David Hendel
Lititz, PA

"Great! I have had *no* problems!"
Timothy Roscoe
Mechanicsburg, PA

"So far my Verbatim disks have been performing flawlessly. Not always so with other disks."
Chris Otis
Hoffman Estates, IL

"Runs quieter in the disk drive."
Richard Cannova
Los Angeles, CA

"Verbatim disks are super. They're our standard for quality."
Bob Mills
Mission, KS

"Anything that prolongs the life of a diskette is a plus. Thank you Verbatim for an excellent improvement."
Steve Toth
Piscataway, NJ

"Of the 130-plus Verbatim disks I have, I'm not aware of any problems. I'm sure the improvements will give your disks an even longer life."
Gerald Janas
Warren, MI

"FANTASTIC. Not a single registration problem. Much more reliable than what I had been using."
Gary Sandler
Playa Del Rey, CA

"Verbatim is much more reliable. I wouldn't trust anything else."
Howard Chin
Pomona, CA

"Thank you for the improvements. Just another reason why I'll always use Verbatim."
James Hassler
Cheyenne, WY

We introduced Verbatim Datalife mere months ago. And it's already playing to rave reviews like these.

But don't take anybody else's word for it. Try it yourself, and see if you don't agree it's the best media you've ever tried. For the name of your nearest Verbatim dealer, call (800) 538-1793, in California call (408) 737-7771 collect.

©1981 Verbatim Corporation
Datalife is a trademark of Verbatim Corp.

We play it back, Verbatim!

SCREEN FORMATTING OF TEXT

by Peter C. Weiglin
Cider Press
©1981, All Rights Reserved

One of the more satisfying sights in Apple land is watching as your program successfully runs and information is being displayed on the CRT (Cathode Ray Tube, or TV screen to most of us). Lists of information scroll by—bunched up against the left margin, maybe, and with numbers not quite lined up, but printing, by golly. But soon, the desire to make things look a bit neater wells up inside of us, particularly if we're going to show this thing off to anyone else, like for example with a project involving our work.

So let's take time here to consider the formatting of data. At the outset, I must say that this material has been gathered from many sources. Further, there must be better ways to do some of these things, or at least alternate methods which you may have. If so, please let us know about them.

The BASIC commands which play a part in formatting include:

PRINT
TAB
HTAB
SPC( )
FLASH
INVERSE
NORMAL
VTAB
INPUT
GET
SPEED
STR$(
LEN(
VAL(
ASC(
CHR$(
HOME (or CALL -936)
CALL -958
CALL -868
POKE

—and others.

The primary statement is, of course, PRINT. All else is formatting which characters we place on the screen, and where we place them. Within the limitations of the 40-character wide screen and the Apple character set, there is virtually nothing that can't be done. (There is both hardware and software available which bends these limitations, but let's stay with the unadorned Apple for now.)

The PRINT statement causes display of letters, numbers and all symbols except the double quotes ("). The reason you can't display double quotes is that this symbol is used to tell the Apple where material to be printed starts and ends. No problem; we have single quotes (') available to pinch hit. Example:

```
10 PRINT "'HELLO', SHE SAID AS WE"
```

Whatever you put between double quotes will be PRINTed. Some things can be PRINTed without quotes, namely numbers and previously identified strings. Examples:

```
PRINT 250
PRINT 4+6

30 X=5
40 PRINT X

50 X$="CIDER PRESS"
60 PRINT X$
```

In addition, Applesoft will allow the double quote *within* a PRINT statement, through the use of CHR$(34). To rewrite our example above:

```
10 PRINT CHR$(34);"HELLO";CHR$(34);", SHE SAID AS WE"
```

What about this one?

```
70 PRINT
```

Well, the PRINT command by itself brings about a line feed and [RETURN]; it's the equivalent of skipping a line.

How about this?

```
?
```

In Applesoft, a quick way to type [PRINT] is to type a question mark as a shorthand symbol.

```
80 ? "APPLE CORE"
```

will yield

```
80 PRINT "APPLE CORE"
```

when the line is LISTed. This not only saves typing strokes, but also allows you to edit a program line more quickly, by taking advantage of the four-character difference between [?] and [PRINT].

All of these PRINT commands so far will begin their display at the left side of the screen. Here's the first approach to making the right-hand phosphors carry their share of the load:

```
90 PRINT "          APPLE"
```

Sure; put in enough spaces to get you as far to the right as you want to be. Oh...if you put in more than 40 spaces, you'll continue the word on the next screen line.

Well, there are easier ways to break free of the left margin; [TAB] in Integer Basic and [HTAB] in Applesoft Basic. They work the same way in their respective dialects; let's use the Applesoft version:

```
100 HTAB 8
110 PRINT "APPLE CORE"
```

—and the first letter of the print statement will appear 8 spaces from the left margin. We risk duplicating the Apple manual here because there has been some confusion about [TAB], since it appears also in Applesoft, but as a slightly different function, and there is no "Integer/Applesoft comparative manual".

SIDE NOTE: THERE IS ONLY ONE "R" IN THE WORD "INTEGER!"

Regarding [TAB] and [HTAB], the lowest legal value you can use is 1; the highest is 255. (Well, you could use [HTAB 0], but only if you wanted to tab 256 spaces.)

Note too, that the [HTAB] command can cause the next item to be printed to the left or right of the current cursor position. Try this:

```
120 HTAB 20:PRINT "E";:HTAB
  10:PRINT "A"
```

Yep, the "E" was printed first, then the "A" to the left of the "E", on the same line. Now try it without the semicolon after the "E", and you see that the "A" is one line lower.

But before we get into the semicolon and other delights, let's clear up the Applesoft [TAB] first. Note that in Line 100 above, [HTAB] stood by itself as a command. So does [TAB] in Integer. [TAB] in Applesoft does not; it must be included in a [PRINT] statement, with the number in parentheses, like this:

```
130 PRINT TAB(8)"APPLE CORE"
```

Why do we bother with [TAB] in Applesoft at all? Frankly, I dunno. [HTAB] runs a bit faster than [TAB] and [TAB] cannot move the cursor to the left. Anybody have the answer?

Now try this:

```
140 A = 7
150 HTAB A: PRINT "APPLE"
```

Yes, you can [HTAB] to a variable, too, which leads to all kinds of possibilities. You may have noticed one application of this in an S. F. Apple Core Disk of the Month:

```
160 C$ = "APPLE CORE"
170 HTAB (20 - INT(LEN(C$)/2))
180 PRINT C$
```

Since 20 is halfway across the screen, and we're HTABbing 20 minus one-half the length of the string, the string will automatically be centered. If you want to print C$ so that it will be flush with the right-hand margin, make line 170 above read this way:

```
170 HTAB (41 - LEN(C$))
```

Yes, that's "41", not "40". The last character of the string will occupy the 40th space on the screen, which is the rightmost space.

We also lightly scratched the surface of ways to print more than one item on the same line. This is most often done with semicolons, although commas and the [SPC( )] command are also available. Try this:

```
190 PRINT "APPLE";" ";"CORE"
```

—and you get the space between the two words. The semicolons tie the elements of a PRINT statement together. Remember, a PRINT statement normally contains an automatic RETURN within it; the semicolon inhibits that RETURN. For example:

```
200 PRINT "APPLE";
210 PRINT "  ";
220 PRINT "CORE"
```

—and all is on the same line, even though three program lines were used.

The comma in a PRINT statement also inhibits the carriage return or line feed, but it does so in a very specialized way. The line

```
230 PRINT "A","B","C","D"
```

—causes the "A" to be printed at the left margin, the "B" in the 17th space, the "C" in the 33rd space, and the "D" in the first space on the next line. These locations are fixed; while you use the comma, you can't change them. It'll go 1, 17, 33 'till Kingdom come.

To print columns, you can use the comma, or you can use a formula which gives you better control of the columns. Let's assume a list of 30 items, each 6 characters wide. If you just [PRINT] that list, it'll scroll the first items on the list right off the screen. But if you split it into two columns, it'll all fit. Try this:

```
240 FOR I = 1 TO 15
242 PRINT S$(I);
244 HTAB 20: PRINT S$(I + 15)
246 NEXT I
```

For more than two columns, the general idea is the same; divide the total number of items in the list by the number of columns you want, and add 1. That becomes the top number in the for-next loop. As an example, for three columns, with 48 items, it's:

```
250 RN = 48
251 N = (RN/3) + 1
252 FOR I = 1 to N
253 PRINT S$(I);:HTAB 15
254 PRINT S$(N + I);:HTAB 30
255 PRINT S$(2*N) + I)
256 NEXT I
```

—Note the semicolons used in lines 253 and 254 to keep the items on the same line. For four columns, N would be set equal to ((RN/4)+1), and another PRINT statement would call for printing S$((3*N)+I). Obviously, you'd have to adjust the tabs accordingly.

Another command which helps to free us from the right margin is [SPC( )], which, as the name implies, involves spaces rather than tabulation. The TAB/HTAB family of commands uses the left margin as its baseline, measuring from there each time it is used. The SPC( ) command measures from the end of the previously printed statement. These next two statements will do precisely the same thing:

```
260 PRINT "A";"          ";"B"
270 PRINT "A"SPC(8)"B"
```

Note that the SPC( ) command must be in a PRINT statement, that it does not require semicolons, and that the number of spaces must be in parentheses. SPC( ) also requires less memory than a string of blank spaces in your program. And, to make this complete, you can also SPC( ) to a variable, which means that this next line will duplicate lines 260 and 270 above, putting eight spaces between the "A" and the "B":

```
280 X = 16
282 PRINT "A"SPC(X/2)"B"
```

Remember, the SPC( )command measures from the end of the item preceding it, rather than HTAB or TAB's reference to the left margin every time they are used.

The FLASH and INVERSE commands are used to highlight characters on the video screen (excuse me, the CRT display). They look good, if not overused. The most common programming error with them is forgetting to use NORMAL to return to good 'ol white on black, but this one manifests itself very quickly. Thanks to the semicolon, we can highlight part of the PRINT statement, like this:

```
290 PRINT "YOU ARE ";:
    INVERSE:PRINT "VERY WEAK";
    :NORMAL:PRINT " TODAY."
```

The INVERSE and NORMAL commands are set off by colons as separate statements in the program line. The PRINT command must be repeated after a colon, or else:

```
300 PRINT "YOU ARE ";:FLASH:
    PRINT "DEAD";:NORMAL:
    PRINT "TODAY."
```

There is one aesthetic problem with FLASH: if it's on the screen for more than a few seconds, it becomes more of a nuisance than a help. Clear it out when it has done its attention-getting job. One good use for FLASH is for showing that the computer is occupied by an internal activity for the moment, with a message like

```
310 FLASH: PRINT
    "WORKING. . .": NORMAL
```

INVERSE and FLASH affect only the Apple's output to the CRT display (excuse me, the video screen). Your input remains white on black.

VERTICAL SPACING

Remember, with the horizontal location process, that semicolons and [SPC))] measure from the previously printed character, while HTAB, TAB, and commas use fixed references measured from the left margin. Vertical location is also possible from a fixed reference (the top of the screen) or a relative distance from the last characters PRINTed.

We begin with the relative reference; the applicable command is:

```
320 PRINT
```

Yup; PRINT by itself allows you to skip a line. To skip three lines, type:

```
330 PRINT: PRINT: PRINT
```

or

```
335 ?:?:?:
```

—remembering that the question mark is a shorthand way of asking for [PRINT]. It's no more difficult than that. But sometimes, as when material scrolls upward off the 24-line screen, relative spacing doesn't quite do the job.

Hence the need for a means of positioning or formatting material vertically. The primary means of doing this is VTAB, or vertical tabulator, which will place the first character of your material a given number of lines down from the top of the screen. So,

```
340 VTAB 5: PRINT "APPLE"
```

will cause the word "APPLE" to be printed on the fifth line, at the left hand margin. Note, however, that whatever is on the screen from previous [PRINT] statements will not be disturbed. This is a situation which could very quickly lead to clutter, but which also has advantages.

Now, it is through the combination of vertical and horizontal formatting commands that we gain both flexibility and power. For example, given a blank screen, try this:

```
350 VTAB 5: HTAB 7:
    PRINT "COMPUTER"
```

—and, if you RUN lines 340 and 350 together, you get both words properly placed, but by independent commands. Aha! That should suggest all kinds of flexible formatting possibilities. The combination of VTAB and HTAB allows you to place any characters anywhere on the screen. One example is:

```
360 VTAB 5: PRINT "NUMBER OF
    MEMBERS: "
```

(lines calculating the number of members)

```
370 VTAB 5:HTAB 20:PRINT M:
    REM M = MEMBERS
```

What that means is that a data framework can be set up on the screen using commands like Line 360 above for the descriptions, and coming back later with commands like Line 370 to supply or to change the data. You can even do this:

```
380 VTAB 5:HTAB 20:INVERSE:
    PRINT "  ";M;"  " :NORMAL
```

—and you get the data in inverse, while the description is normal. (The spaces before and after the M are just to make the display look better). It's not uncommon to use the top 15 screen lines as a form framework, while the prompts for inputting data occupy lines 16 through 24. Part of the screen can be made to hold still, or be "frozen", by use of a [POKE] command.

A [POKE] command?? Yup. As with any POKE command, we enter a location and a value, lowering the top of the screen from line 0 to another line between 0 and 23. The memory location is 34, so if you want the screen fixed in place above line 10, the command is:

```
390 POKE 34,9
```

—and you can't print above line 10. Well, yes you can, by using VTAB to a line above line 10. But all of the text scrolling will take place below line 9 until you reset the window; try:

```
400 POKE 34,0 : HOME
```

and all is normal. In addition, location 34 (along with the other windows) can be cleared with:

```
405 TEXT
```

That HOME reminds me of three commands which can be used to "erase" material already on the screen prior to installing revised data at the same screen location. Example: if we set:

```
410 HOME:VTAB 4:HTAB 5:
    PRINT "CIDER SQUEEZINGS"
```

—we get the printed words on the fourth line down, covering the horizontal spaces from 6 through 21, as the string is 16 characters long. Now, we can try to replace the word "squeezings" with the word "press", thus:

```
420 VTAB 4:HTAB 11:PRINT
    "PRESS"
```

and we get—"CIDER PRESSZINGS". Oops! The new word didn't erase all of the longer word that was there first. Well, we couldn't use HOME or CALL -936, or we'd clear the whole screen.

Note: HOME is not available in Integer Basic; use CALL -936.

Note to Note: THERE IS ONLY ONE "R" IN "INTEGER"!

But we can use CALL -958, which clears only that part of the screen below and to the right of the cursor location. Or, if you have other material below the cursor, use CALL -868, which clears only the rest of the line on which the cursor is located. Like this:

```
430 VTAB 4:HTAB 11:CALL
    -868:PRINT "PRESS"
```

Note that the [HTAB] precedes the [CALL]; you thus clear from the 11th space to the right margin.

Now let's say you have a list of data to be printed out on the screen, and the list has more than 24 items. Whoops—there go the first few items off the top of the screen, while you scramble to find CTRL-S to stop the runaway. Well, there is a better way. At least two better ways, in fact.

Method 1:
Set your PRINT routine up like this:

```
800 REM PRINT ROUTINE
810 F = 0 : REM FLAG
820 FOR I= 1 TO N : REM N=
    NO. OF ITEMS
830 (the data print statement for
    each line)
840 F= F+ 1: IF F> 22 THEN
    GOSUB 1000: F= 0: HOME
850 NEXT I
```

```
1000 INVERSE:PRINT " PRESS
     [SPACE] TO CONTINUE...
     ";: GET A$: PRINT:
     NORMAL: RETURN
```

This method increments a flag with each line printed. At Line 23, the subroutine is called. Pressing the space bar (or any other key) causes the PRINT routine to continue in an orderly fashion, a screenful or "page" at a time.

Method 2

```
900 REM ROUTINE
910 FOR I = 1 TO N
920 (the data print statement for
    each line)
930 IF PEEK (37)> = 22 THEN
    GOSUB 1000: HOME
940 NEXT I
```

This one looks for the cursor location, at memory location 37. Finding the cursor at or below Line 22, it triggers the subroutine. You can, of course, set the values at other lines, to fit your format.

A semi-digression (which is nothing like a trailer truck running off the road): you can vary your displays by using the [SPEED] command to slow down the rate at which characters appear. SPEED= 255 is both the fastest speed and the default value. . . what you normally see on your Apple is the "255" rate. The slowest rate is 0. Like [FLASH], the slower display rates are useful in small quantities, but become irritating after prolonged exposure.

INPUTS AND GETS

Having covered the means whereby PRINTed characters output by the computer can be liberated from the left margin, or the top of the screen, let's consider the characters input by you in answer to requests for information necessary to the execution of your program. Like this:

```
440 PRINT "YOUR NAME";:
    INPUT A$
```

or

```
445 INPUT "YOUR NAME?";A$
```

which are the two variations of this command. Now, as shown above, these lines will begin at the left margin. You can move 'em wherever you want 'em.

Note another wrinkle here. In Line 440, we used a PRINT statement for the prompt material, followed by the INPUT request. In Line 445, the prompt material is included as part of the INPUT request. We didn't put the question mark in Line 440, because the system does it automatically, since it didn't find any prompt material in quotes. The INPUT request, when used as in Line 445, allows you to replace the question mark with any text you desire.

In a format situation, you quite often want a number printed in the same position on the screen, even if it may contain different numbers of digits at various times. *Don't input* a real or integer variable, INPUT a string, then use the VAL( ) function to derive the numerical value. If you hit a letter when the computer is expecting a number, it's error time.

Having INPUT a string, you can use one of my favorite subroutines, which equalizes string lengths:

```
450 FOR K = 1 to L% :IF LEN(A$)
    < L% THEN A$ = " "+ A$
451 RETURN
```

Of course, you have to set L% to the maximum string length for that application, but you can use different lengths during the program. It looks like this:

```
460 L% = 4: GOSUB 450: PRINT
    A$:
A% = VAL(A$)
```

The subroutine inserts blank spaces ahead of the INPUTed number, then creates the Integer variable after printing the formatted string. (Yes, it could be a real variable too).

The PRINT statement for a number calculated by the computer can also be converted to a formatted string before PRINTing; in fact, there's no law against using the A$ variable over and over, in conjunction with Subroutine 450. A$ needs to retain a value only long enough to PRINT it, after which it moves on to the next number.

```
470 A$ = STR$(TL):L% = 4:
    GOSUB 20: HTAB 4: PRINT
    A$;:A$ = STR$(TM):
    L% = 6:HTAB 18:PRINT A$
```

The INPUT statement requires a [RETURN] after the data is typed, and we all know that a GET statement does not. That is, as soon as the info is typed in response to a GET statement, the microprocessor is off and running with it:

```
480 PRINT "ENTER INITIAL: ";:
    GET A$
```

So a GET statement, while it is faster, is good for only a single-character response, right? Wrong. If you have a definite number of characters to be entered, try this:

```
490 PRINT "ENTER 3 INITIALS:
    ";: GET A$: GET B$: GET C$:
    D$ = A$+ B$+ C$: PRINT D$
```

As an alternative, you can GET in a loop, as in:

```
495 PRINT "ENTER 3 INITIALS: ";:
    FOR I = 1 TO 3: GET A$:
    B$ = B$+ A$: NEXT:PRINT B$
```

Notice that you can't do with GET what you can with INPUT as we showed in Line 445 above; you must use the PRINT statement. Note also, the absence of semicolons between the GETs. Either example will pick up three characters, no more, no less. It combines the three single-letter strings into one three-letter string, and heads for the hills. . . or at least for the next program line. For many reasons, including some caused by quirks in the machinery, you're best off acquiring numerical answers by GETting string variables and converting to real or integer using the VAL( ) function.

Finally, the most direct, and yet most sophisticated way to put a text character on the TV/CRT/VDU (Video Display Unit) screen is to POKE it there, directly into one of the 960 cells of the 40 by 24 "grid". Try this:

```
500 HOME: POKE 1335, 195:
POKE 1336, 201: POKE 1337,
196: POKE 1338, 197: POKE
1339, 210: POKE 1340, 160: 510
POKE 1341, 208: POKE 1342,
210: POKE 1343, 197: POKE
1344, 211: POKE 1344, 211
```

How 'bout that! What you've done, of course, is exactly what the Apple has been doing for you all through this article. You have first identified a location on the screen, and then placed a specific character in that location.

The first number in the POKE command is the location for a particular block on the screen, which is mapped out on Page 16 of the Apple Reference Manual. Each of the 960 blocks on the screen has its own address in the Apple's memory. We used the Figure to locate the address of the eleventh line for our example above (starting location in 1320 decimal or 528 in hexadecimal), and then moved 15 spaces in from the left margin by adding 15 to the 1320 line starting address, to begin our word at location 1335.

Look closely at that screen map; notice that the blocks are not numbered consecutively, but rather in three horizontal bands of eight lines each. You can easily demonstrate this phenomenon:

```
520 HOME: FOR I = 1024 TO
    2039:530 POKE I, 193: NEXT
```

Which character goes into that location is determined by a numerical code; the American Standard Code for Information Interchange, or ASCII. (To pronounce "ASCII", think of a donkey unlocking a door). The table on Page 15 of the Reference Manual shows the ASCII numbers for the characters available to you on the Apple. The ASCII code number is the second number in the POKE command.

Now try this:

```
540 HOME:POKE 1339, 220
```

That's a backward slash, and you won't find it on the Apple keyboard. The only way you'll get it on the screen is by POKEing it there, or by using the CHR$( ) function, which also involves the ASCII code number, in a PRINT statement. There are a couple of other characters you'll not find on the keyboard, including the underline (ASCII 223) and left bracket (ASCII 219). For all characters, whether normal, flashing, or inverse, the POKEs give you direct access to the screen.

AN INTRODUCTION TO LIGHT PENS

by Neil D. Lipson
I.A.C. Software Chairman

It has been brought to my attention that there are many programmers that do not know the operation of a light pen. The operation is relatively simple. There are a few different types of pens out on the market, each of which functions differently. Hopefully, this article will help you in your understanding of the operation of these pens.

There are basically three types of pens in use for the Apple. Those that use an on-off type of cell (typically a phototransistor), those that operate on resistance, and those that use an optodetector, sensitive to raster scan of a monitor. The most expensive (over $200) is the raster scan model. It picks up the actual scan on the monitor and uses this for determining the location of the point. Since this is far faster than the former two, it is fast enough to draw lines very quickly. However, to accomplish this, it must have some sophisticated electronics and a rather large barrel, which accounts for the very high price of the unit. It can pick out very small points (hi-res), among other things. This is due to the focusing nature of the optodetector. It must be very accurately adjusted along with the monitor, or it will not operate correctly.

The most popular light pens for the Apple, however, are those that use phototransistors, and cadmium sulfide and selenide cells. The phototransistor cell is somewhat faster than the cadmium cells, but does not do as much. It can detect only an on-off threshold. When using it with a monitor or TV, the brightness and contrast must be adjusted just right, or again, it will not always operate perfectly. This leads into another shortcoming of the cell, in that it cannot determine different variations of light. Therefore, you cannot use software to determine differences in light; only an on-off condition. This is fine if all you do is pick points from the screen with no change in light conditions. However, if the sun rises and shines through the window close to the monitor, you may end up readjusting the monitor. Also, suppose you want to use the pen to give variable readings instead of on and off?

This is the reason for going to a cadmium sulfide. They will do both of these functions. However, a pure cadmium sulfide cell is very slow. The solution for this is to go to a high speed cadmium selenide cell. You then have the speed of a phototransistor, and the ability to read different levels of light. These cells are quite expensive, and the manufacturer must choose very carefully a cell that fits all these parameters. What had to be done to satisfy these conditions, was to custom make the cell to operate under these parameters. While this added to the expense of the light pen, it gave the user the ability to do everything he could want with a light pen. Pens of this type can act as light meters, and other applications which the regular phototransistor pens cannot do.

The pen works differently for different applications. To spot a point on the screen, it will check every point very fast, and look for a possible match. When the point on the screen turns off, and the pen "sees" an off condition, it registers a possible hit. It then verifies the point by turning it on and off, checking it everytime it does this. If after a few checks, it verifies, and then uses this point in the program.

When the pen is used in drawing points, there are a few different methods used. One is to draw a "sheet" of light coming from the top down, and then from the left to right, and picking out the x and y coordinates. Another method is to scan the screen in bars, and localize which bar is being scanned. Another method is by using blocks of light. They are all satisfactory and give roughly the same end result.

For those pens that use cadmium selenide cells, different levels of light can be detected, and used accordingly. This type of pen is plugged into the game paddle into the pdl(0), pdl(1) or pdl(2) point. They simulate the paddle operation. The resistance (if the cell is properly designed) is from 0 to 150K ohms. As the amount of light rises, the resistance drops. Even if the operation of the resistance is not linear, software can correct it.

Hopefully, this will help you in your understanding of light pens and their operation.

Screen Formatting of Text

(continued from page 9)

But all that POKEing is a tedious process for large amounts of text, which is why we have all of the other PRINT and formatting commands. But with the POKEs, you've just stepped out onto the path that leads to (gasp!) machine language. I suggest that you review the Apple II Reference Manual (spiral bound, 1979), Pages 14-16, in which the Figure and Table appear. While you're in the Manual, you can decide how much closer to machine language you'd like to get, by translating the POKEs into "00"s through "FF"s.

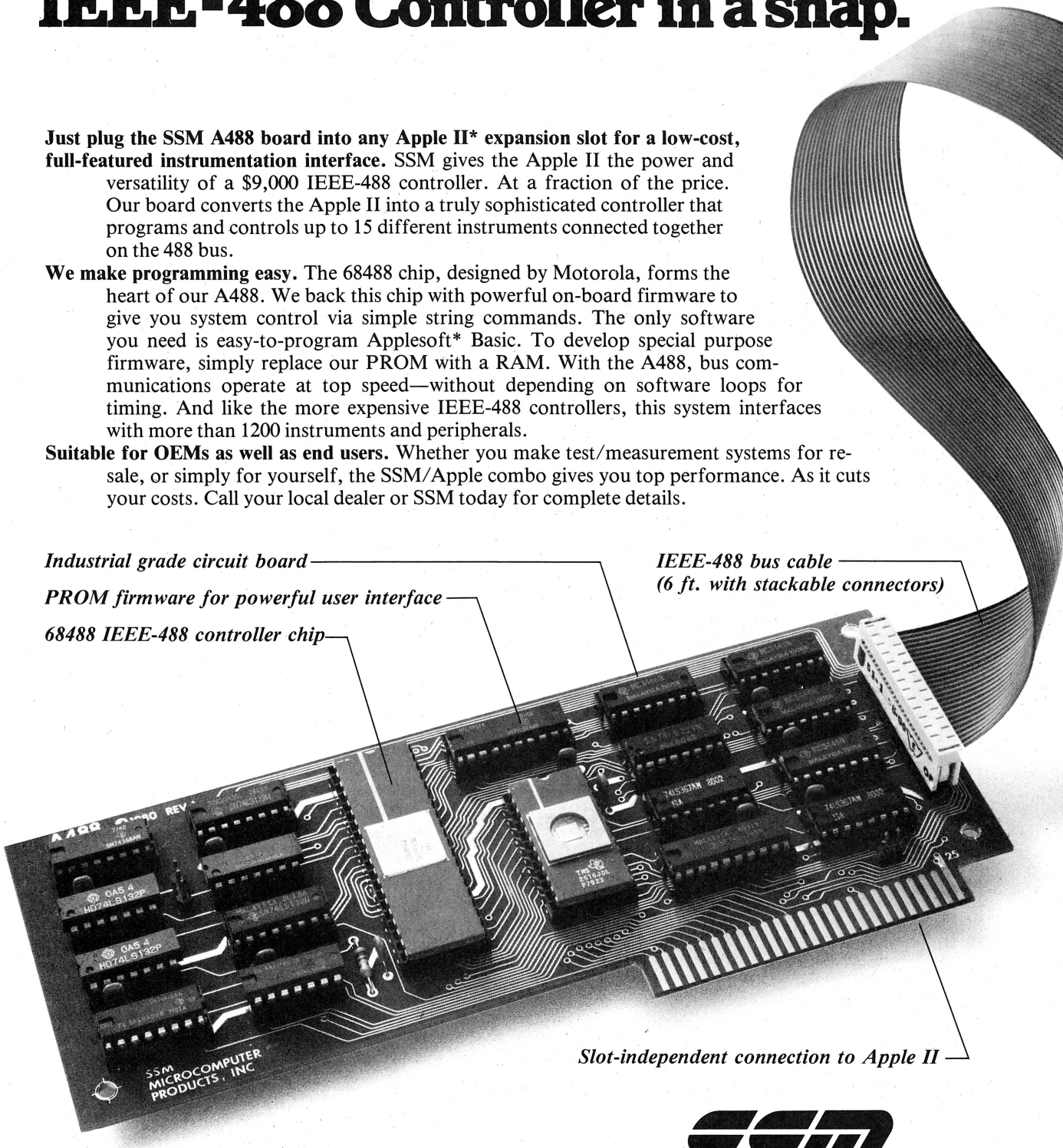
As noted, if you find better ways, please let us know; and do your part to beautify the world's CRT/TV/VDT screens.

Make the Apple II* a powerful IEEE-488 Controller in a snap.

Just plug the SSM A488 board into any Apple II* expansion slot for a low-cost, full-featured instrumentation interface. SSM gives the Apple II the power and versatility of a $9,000 IEEE-488 controller. At a fraction of the price. Our board converts the Apple II into a truly sophisticated controller that programs and controls up to 15 different instruments connected together on the 488 bus.

We make programming easy. The 68488 chip, designed by Motorola, forms the heart of our A488. We back this chip with powerful on-board firmware to give you system control via simple string commands. The only software you need is easy-to-program Applesoft* Basic. To develop special purpose firmware, simply replace our PROM with a RAM. With the A488, bus communications operate at top speed—without depending on software loops for timing. And like the more expensive IEEE-488 controllers, this system interfaces with more than 1200 instruments and peripherals.

Suitable for OEMs as well as end users. Whether you make test/measurement systems for resale, or simply for yourself, the SSM/Apple combo gives you top performance. As it cuts your costs. Call your local dealer or SSM today for complete details.



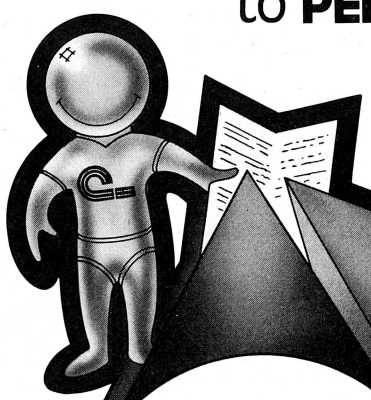
SSM's A488 board expands the Apple II to a high-performance IEEE-488 controller.

* Registered trademarks of Apple Computer Inc.

SSM Microcomputer Products, Inc.
2190 Paragon Drive
San Jose, CA 95131
(408) 946-7400 Telex: 171171
TWX: 910-338-2077

MR. RAINBOW

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



A STELLAR TREK
the definitive Hi-Res color version of the classic Startrek game. Three different Klingon opponents. Many command prerogatives from use of weapons to repair of damages. Needs 48K Applesoft ROM.
Disk...**$24.95**

VERSAWRITER II
A drawing tablet, simply plugs into your game I/O port. Trace, draw, design, or color any type of graphic. Adds words to pictures. Creates schematics. Computes Distance/Area of any figure. New - fill any area on the screen in seconds with over 100 different and distinct colors. Needs 32K Applesoft ROM and disk drive. A bargain at...
$249.95

BOWLING DATA SYSTEM
This data mangement program provides accurate record keeping and report generation for bowling leagues of up to 40 teams with 6 bowlers per team. Needs 80-column printer, 32K Applesoft ROM.
Disk...**$79.95**

SUPER SOUND
Musical rhythms, gunshots, sirens, laser blasts, explosions...add these and many more exciting sounds to your Apple. Use them in your programs, or create your own SUPER SOUNDS. Needs 16K Applesoft. Have a blast for only
$12.95...Tape
$16.95...Disk

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

Open Tues. - Sun.

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

Visa/Mastercharge welcome.

RAINBOW COMPUTING INC.

GARDEN PLAZA SHOPPING CENTER
9719 RESEDA BOULEVARD DEPT. 12AO
NORTHRIDGE, CALIFORNIA 91324
PHONE (213) 349-0300

NOTES ON HI-RES GRAPHICS ROUTINES IN APPLESOFT

by C. K. Mesztenyi
Washington Apple Pi

Checking out the entry points given by J. Crossley in the article "APPLESOFT INTERNAL ENTRYPOINTS" in the March/April 1980 Apple Orchard, I found the given entry points were 4 bytes off from the given ones in our APPLE II Plus. Furthermore, after checking out the routines in more detail, I thought to share my experiments with other APPLE II Plus owners interested in machine language programming. In the first section, I describe the essential data storage area, in the second I give the entry points of the subroutines somewhat more detailed than in the above article, and in the last section I give some listings of instructions following the entry points so that one could identify it for different versions of Applesoft.

1. DATA STRUCTURE

There are four data in five memory locations which specify a point on the high resolution screen (whether the screen is displayed or not, is irrelevant). I call these data collectively as external cursor data. The five memory locations, and their contents are as follows:

- $E0: Low order bits of the horizontal screen coordinate
- $E1: High order bit of the horizontal screen coordinate
- $E2: Vertical screen coordinate
- $E4: Color masking word from the color table ($F6F6-$F6FD)
- $E6: Page indicator ($20 for Page 1, $40 for Page 2)

I have called the above set of data as external cursor data since the actual point plot is performed by the following five instructions:

```
LDA  $1C
EOR  ($26),Y
AND  $30
EOR  ($26),Y
STA  ($26),Y
```

which uses data located at $1C, $26, $27, register Y and $30. The contents of register Y are always picked up from location $E5 prior to the above instructions, thus we may call the data in the following five locations as internal cursor data:

- $1C: The color masking byte shifted for odd address and none black or white, unchanged otherwise.
- $26, $27: (Low, high order) address of the byte corresponding to the page, vertical coordinate and leftmost seven points of the screen.
- $E5 (register Y): The integer part of the horizontal screen coordinate divided by 7.
- $30: The bit position taken from Bit Position Table corresponding to the remainder of the horizontal coordinate divided by 7.

These two cursor data (external and internal) are equivalent in the sense that given one, the other can be derived from it. There would be no need to make any distinction if they would correspond to each other all the time, but unfortunately, this is not always the case, e.g. the following sequence of BASIC instructions:

```
HCOLOR=1
HPLOT 0,0 TO 10,10
HCOLOR=2
HPLOT TO 10,50
```

plots two lines, (0,0) to (10,10) and (10,10) to (10,50), both with color 1, i.e. HCOLOR=2 has no effect. Actually it resets the color code in $E4 but it does not change $1C, and the statement HPLOT TO picks up whatever was left in $1C.

A machine language programmer can write his/her own graphics routines which takes time and uses sometimes much-needed memory space. Thus using the available programs in Applesoft ROM can be advantageous. If execution time is also important, as in the case of animation, then one should concentrate only on the internal cursor data, and modify the external cursor only when it is necessary. The entry points INTX and INTY, provide the basic routines for incremental plotting which are not available directly in BASIC. Also, modifying the external cursor coordinates allows the use of HLINE with off-set.

2. ENTRY POINTS IN APPLESOFT

Page and Color:

HGR2 ($F3D8): Displays page 2 with all graphics mode, sets $E6 to $40, clears page 2 (black) and sets $1C to zero (black I).

HGR ($F3E2): Displays page 1 in mixed mode, sets $E6 to $20, clears page 1 (black) and sets $1C to zero (black I).

BKGND ($F3F4): Clears the page defined by $E6 to the color defined by the contents of register A, which should be one from the Color Masking Table. Also stores register A in $1C.

HCOLOR ($F6F0): Assumes register X contains the color index (0 to 7). The routine picks up the appropriate color code from the Color Masking Table and stores it in $E4.

Positioning Entries:

HPOSN ($F411): Assumes the input upon entry in the registers as:

register X = low order bits of the horizontal screen coordinate,

register Y = high order bit of the horizontal screen coordinate,

register A = vertical screen coordinate.

The routine stores the registers in $E0, $E1 and $E2. Then, using $E6, sets $26, $27, $30 and $E5 together with register Y, and sets $1C to the contents of $E4. Thus this routine makes the internal cursor equivalent to the external one.

INTX ($F465): Modifies the internal cursor data in $1C, $E5, register Y and $30 so that it corresponds to incrementing/decrementing the horizontal screen coordinate X by one. Upon entry, if the N-flag is zero (positive) then it increments; if N is set (negative) then it decrements. The modification has a wrap around feature, i.e., incrementing/decrementing at the extreme sides of the screen defined by the internal cursor causes it to come back on the other side. The routine assumes that register Y corresponds to $E5 upon entry, and leaves the routine correctly modified if necessary.

Upon testing the N-flag the routine jumps to DECRX or INCRX.

DECRX ($F467): The routine modifies the internal cursor data by decrementing the horizontal screen coordinate by 1 (see INTX).

INCRX ($F48A): The routine modifies the internal cursor data by incrementing the horizontal screen coordinate by 1 (see INTX).

INTY ($F4D3): Modifies the internal cursor data in $26, $27, so that it corresponds to incrementing/decrementing the vertical screen coordinate by one. Upon entry, the N-flag is checked, and if it is set (negative) then goes to INCRY to increment by one, if it is not set (positive) then goes to DECRY to decrement by one. Note that the sign convention is used opposite of INTX. These entries also have the wraparound features, i.e. if the incrementation/decrementation causes the cursor to leave the screen on the bottom/top, then it comes back on the top/bottom.

DECRY ($F4D5): The routine modifies the internal cursor data by decrementing the vertical screen coordinate by 1 (see INTY).

INCRY ($F504): The routine modifies the internal cursor data by incrementing the vertical coordinate by 1 (see INTY).

IPOSN ($F5CB): Sets the external cursor data in $E0, $E1, $E2 equivalent to the internal cursor coordinate data.

Plotting Entries:

HPLOT ($F457): Assumes input data in the registers as HPOSN:

register X: low order bits of horizontal screen coordinate,

register Y: high order bit of horizontal screen coordinate,

register A: vertical screen coordinate.

The routine calls HPOSN with the above data, then goes to PLOT.

PLOT ($F45A): The routine executes the five instructions listed in the beginning of the article which plots a point using the internal cursor data. If this entry is used directly, then the user should make sure that register Y contains the data from $E5.

HLINE ($F53A): The routine assumes input in the registers:

register A: low order bits of horizontal screen coordinate,

register X: high order bit of horizontal screen coordinate,

register Y: vertical screen coordinate.

(Note that it is in different order than (HPOSN.)

The routine draws a line from the internal cursor position to the point defined by the input. Upon exit, it leaves the external cursor data corresponding to the input, the internal cursor data corresponding to the last plot point of the line. If the internal and external cursor data were not equivalent, then an off-set occurs. This can be visualized as follows:

Draw a line segment from the external cursor coordinates to the input coordinates. Now move this line segment parallel to itself so that the end-point at the external cursor position gets into the internal cursor position. This is the actual line segment which will be drawn. If it gets outside of the screen, then a wrap-around occurs, i.e. it comes back on the opposite side of the screen.

APPENDIX

The first few instructions are listed for each entry point so that one could identify them using the Monitor list feature.

Bit Position Table:

```
$F5B2: $81=10000001
$F5B3: $82=10000010
$F5B4: $84=10000100
$F5B5: $88=10001000
$F5B6: $90=10010000
$F5B7: $A0=10100000
$F5B8: $C0=11000000
```

Color Masking Table:

```
$F6F6: $00=00000000 (black I)
$F6F7: $2A=00101010
$F6F8: $55=01010101
$F6F9: $7F=01111111 (white I)
$F6FA: $80=10000000 (black II)
$F6FB: $AA=10101010
$F6FC: $D5=11010101
$F6FD: $FF=11111111 (white II)
```

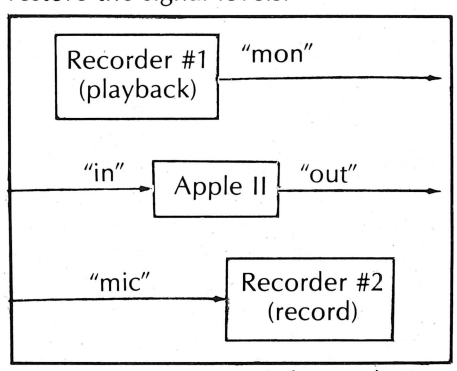
```
HGR2:    $F3D8:   BIT $C055
                  BIT $C052
                  LDA #$40
                  BNE $F3EA

HGR:     $F3E2:   LDA #$20
                  BIT $C054
                  BIT $C053
                  STA $E6
         ...
BKGND:   $F3F4:   STA $1C
                  LDA $E6
                  STA $1B
                  LDY $#00
         ...
HCOLOR:  $F6F0:   LDA $F6F6,X
                  STA $E4
                  RTS

HPOSN:   $F411:   STA $E2
                  STX $E0
                  STY $E1
                  PHA
                  AND #$C0
                  STA $26
         ...
IPOSN:   $F5CB:   LDA $26
                  ASL
                  LDA $27
                  AND #$03
                  ROL
                  ORA $26
         ...
INTX:    $F465:   BPL $F48A
DECRX:   $F467:   LDA $30
                  LSR
                  BCS $F471
                  EOR #$C0
         ...
INCRX:   $F48A:   LDA $30
                  ASL
                  EOR #$80
                  BMI $F46E

INTY:    $F4D3:   BMI $F505
DECRY:   $F4D5:   CLC
                  LDA $27
                  BIT $F5B9
                  PNE $F4FF
         ...
INCRY:   $F505:   CLC
                  LDA $27
                  ADC #$04
                  BIT $F5B9

HPLOT:   $F457:   JSR $F411
                  LDA $1C
                  EOR ($26),Y
                  AND $30
                  EOR ($26),Y
                  STA ($26),Y
                  RTS
HLINE:   $F53A:   PHA
                  SEC
                  SPC $E0
                  PHA
                  TXA
                  SPC $E1
```

AN APPLE II QUICKIE

by Gordon Stallings
Tulsa Computer Society

Here is a simple program which has been in use by the TCS Apple Users for the past year, and which should be more widely known:

To copy a cassette tape from one recorder to another, you can use the Apple II as an intermediary to restore the signal levels.



Put this program into the Apple II:

```
0000 20 FD FC JSR  $FCFD
0003 AD 20 C0 LDA  $C020
0006 4C 00 00 JMP  $0000
```

Start the program: *0G

As long as this program is running, any signal received from recorder #1 will be sent to recorder #2. For best results, the following tips should be observed:

1. Be sure that Recorder #1 is set up to reliably read the tape which you are wanting to copy—volume, tone, and head alignment must be set so that the Apple can read the tape.
2. Recorder #2 should have correct head alignment so that the new tape will be compatible with other machines.

HOW IT WORKS

The JSR calls subroutine $FCFD in the Monitor ROM, which watches the cassette input port waiting for a change of state, which indicates a zero-crossing on the playback tape. When the transition occurs, the subroutine RETURNS. The LDA $C020 toggles the cassette output port, recording a transition on the new tape. The JMP closes a program loop which can only be broken by RESET.

PRACTICAL SUPER HIRES GRAPHICS

by R. H. Good
California State University
Hayward, CA

Bob Bishop's article in the Fall 1980 Apple Orchard serves its didactic purposes admirably in showing that resolution of 560×192 is attainable with Apple II. But the program he presents in his Fig. 4 has two substantial defects: it is slow, and it leaves occasional dark segments (particularly noticeable on a color TV monitor).

Both problems can be alleviated with little trouble. The secret is to HPLOT successive vertical segments of steep lines as follows:

HCOLOR=3, HPLOT first segment;
HCOLOR=7, HPLOT second segment; increment x by 1;
HCOLOR=3, HPLOT third segment;
HCOLOR=7, HPLOT fourth segment; increment x by 1; and so on.

Note that HCOLOR=7 automatically plots things one-half x unit to the right of HCOLOR=3. A black line, HCOLOR=4, helps make the segments uniform.

Here is Bishop's example again, with its two nearly-vertical lines, to which has been added a third line drawn by the improved procedures outlined above.

```
]LIST

5  REM

LINE DEMO * R H GOOD

10  HGR
20  REM

HPLOT

40  PRINT  CHR$ (7): REM BELL
50  HCOLOR= 3
60  HPLOT 130,0 TO 140,156
70  PRINT  CHR$ (7): REM BELL

100  REM

BISHOP

110  FOR Y = 0 TO 156
120 XX = 280 + Y / 8
130 XXX = XX / 4:MX = XX - 4 * XX
     X
140  IF MX = 0 THEN  HCOLOR= 2
150  IF MX = 1 THEN  HCOLOR= 6
160  IF MX = 2 THEN  HCOLOR= 1
170  IF MX = 3 THEN  HCOLOR= 5
180  HPLOT XX / 2,Y
190  NEXT
200  PRINT  CHR$ (07): REM BELL
220  REM

RHGOOD

230 X = 150
240 D = 159 / 20
250  FOR Y = 0 TO 150 STEP D
260  HCOLOR= 4
270  HPLOT X + 3,Y TO X + 3,Y + D
      - 1
280  HCOLOR= 7
290  HPLOT X,Y TO X,Y + D - 1
300  HCOLOR= 3
310 X = X + 1
320 Y = Y + D
330  HPLOT X,Y TO X,Y + D
340  NEXT
350  PRINT  CHR$ (07): REM BELL
360  PRINT  TAB( 21)"HPLOT"
370  PRINT  TAB( 22)"BISHOP"
380  PRINT  TAB( 24)"R H GOOD"
390  CALL  - 756: TEXT : END
```

The Most Exciting Event of All Time for Apple* Computer Users

For the first time ever, a computer show devoted exclusively to the Apple computers. Applefest '81. The largest event in the world for Apple users.

You'll see it all at Applefest. All of the latest Apple software for home, business and education. New peripherals and accessories. Useful publications and support services. Over a hundred exhibits of products and applications from around the country.

Don't miss the Applefest '81 Seminar Program, a two day series with different topics running every hour. Learn about business software, making the Apple work for you at home, programming the Apple, Apples in education and more. Meet representatives from Apple Computer and other leading Apple support manufacturers. All of the seminars are free of charge to Applefest attendees.

Regular adult admission is just $3 per day. You can register at the door, or use the pre-registration form.

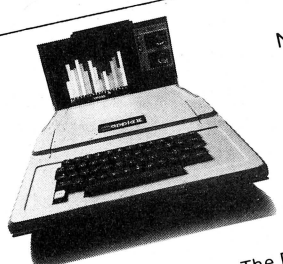
If you use the Apple, or are thinking about buying one, you won't want to miss a minute of Applefest '81.

Saturday, June 6 — Sunday, June 7, 1981

11:00 A.M. - 7:00 P.M. Saturday; 11:00 A.M. - 6:00 P.M. Sunday

The Plaza Castle

Arlington Street at Columbus Avenue, Boston, Massachusetts



Name _____

Street _____ State _____ Zip _____

City _____

Please send me _____ tickets. $_____ enclosed.

Detach and return to: The Boston Computer Society, Three Center Plaza, Boston, Massachusetts 02108

applefest '81

Sponsored by Apple/Boston
The Boston Computer Society's Apple User Group
Three Center Plaza, Boston, Mass. 02108
(617) 367-8080

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

COMPARING APPLESOFT PROGRAMS FOR DIFFERENCES

By Charles Boody, David W. Walker
and Val J. Golding
Call -A.P.P.L.E.

In the July/August 1980 Call -A.P.P.L.E., Charles Boody wrote:

> *"I often use programs that are published in various journals, and almost always find they lack some small (or sometimes some large) pieces that will make them more useful for me. Once I have made the changes or additions, I would often like to pass the changes on to others in the form of an article or list of changes. Unfortunately, it is difficult to remember what changes have been made in which lines of what parts of which subroutines, and how those changes affect other parts of the program. I also often find myself with more than one version of a program; my own or public domain ones, that seem to have small differences that are very hard to trace down. Sometimes too, I find that I have kept several versions of a program under development, but am not sure about the detailed differences in them."*

To alleviate some of those problems, Boody wrote the COMPARE program. Subsequent to the publication of the COMPARE program, we received a letter from David W. Walker who added some changes to the original program which he felt were significant changes, and we agree. In addition, Walker made several nice, but purely cosmetic changes. Interestingly enough, the form in which he submitted the changes to us was a printout produced by the COMPARE program, which for us made the work of updating the original a breeze!

```
10  REM

COMPARE CAPTURER BY VAL J GOLDING



100 D$ =  CHR$ (13) +  CHR$ (4):O
    P$ = "OPEN":WR$ = "WRITE":CL
    $ = "CLOSE": INPUT "  PROGRA
    M NAME TO CREATE TEXTFILE FR
    OM ?";NAME$:FILE$ = NAME$ +
    ".T": PRINT D$;OP$"LIST."NAM
    E$;D$;WR$"LIST."NAME$: PRINT
    "LOAD";NAME$
110 DL$ = "DELETELIST.":TXT$ = "
    TEXT:DEL0,0:END":CMD$ = ":HO
    ME:POKE33,33:LIST1-:":CH$ =
    "?CHR$(4)":PT$ = "?":C$ = "
    ":Q$ =  CHR$ (34):ZERO$ = "
    0 " + CH$ + Q$ + OP$ + FILE$
     + Q$ + C$ + CH$ + Q$ + WR$ +
    FILE$ + Q$ + CMD$ + CH$ + Q$
     + CL$ + Q$ + C$ + CH$ + Q$ +
    DL$ + NAME$ + Q$ + TXT$
120 PRINT ZERO$: PRINT "RUN": PRINT
    D$"CLOSE"
130 PRINT D$"EXECLIST."NAME$"
```

```
LIST

100  REM

COMPARE PROGRAMS ROUTINE

BY CHARLES G. BOODY

ADAPTED BY DAVID W WALKER

110 D$ =  CHR$ (13) +  CHR$ (4):
    PRINT D$"NOMONCIO"
120  GOTO 2000
150  ONERR  GOTO 810
160  GOTO 340: REM

200  REM

GETS STATEMENT FROM THE OLD FILE


210  LET OS$ = "": PRINT D$;"READ
     ";O$
220  GET A$: IF A$ <  > CHR$ (13
    ) OR  LEN (OS$) = 0 THEN  IF
     LEN (OS$) < 255 THEN OS$ =
    OS$ + A$: GOTO 220
230  RETURN
```

The COMPARE program requires that you first write your object programs to a text file which it can read. This is handled entirely automatically by our COMPARE CAPTURER, which requires only that you supply it with the name of the program you wish to create a text file from, and that this program has no line numbered zero.

The CAPTURER program will first create a temporary text file named "LIST. FILENAME", which it will later delete when it has served its purpose. CAPTURER then EXECs this file, which in turn lists the BASIC program to a text file named "FILE NAME.T". When the "old" and "new" files have been created, just run the COMPARE program, which will ask you for the names of the old and new programs. The suffix ".T" should not be added; COMPARE will do this itself.

It should be noted that program lines longer than 255 characters will be truncated by the COMPARE program, and that certain control characters may produce unexpected events. For example, a REM line which contains a Control M, [Carriage Return] (such as in the COMPARE program itself) will result in COMPARE believing that the data following the Control M is a separate program line, even though there is no line number.

Printer protocol is established in the group of lines commencing at 1000, and line 1110 assumes you have a printer card requiring a Control I. In both lines 1110 and 1120, the variable establishes a default of 70 characters per line for the printer. This may be changed to suit your specific needs.

That the program works is attested to by a page of sample output, which compares "CREATE3" (an early version of the EXEC file writer) and "COMPARE CAPTURER", the final version.

When entering COMPARE CAPTURER, it would be a good idea to not enter the extraneous spaces in the latter half of line 110, or a "TOO LONG ERROR" may result. The CAPTURER should also serve as a reasonably well written example of how to write a text file that writes a text file. Q$ is set to equal CHR$(34), the double quote ("), and is used when required to place a PRINT statement within a PRINT statement.

We have preferred to keep this type of shenanagin to an absolute minimum by almost entirely using strings instead of PRINT statements. The strings are defined in line 100 and in the first part of line 110, while the latter part of 110 is almost exclusively concatanenation. ZERO$ is the villain in this one, as it is later EXECed into the object program and run, causing the object program to list itself to the second text file.

```
250  REM

GETS STATEMENT FROM THE NEW FILE


260  LET NS$ = "": PRINT D$;"READ
     ";N$
270  GET A$: IF A$ <  > CHR$ (13
     ) OR  LEN (NS$) = 0 THEN  IF
      LEN (NS$) < 255 THEN NS$ =
     NS$ + A$: GOTO 270
280  RETURN
300  REM

PRINTS A STATEMENT WITH A
LINE LENGTH OF LL

310 II = 0
320 IO = II + 1:II = II + LL: POKE
     36,9: PRINT  MID$ (J$,IO,LL)
     : IF II <  LEN (J$) THEN 320


330  RETURN
340  PRINT D$"OPEN"O$;D$"OPEN"N$
350  GOSUB 210: GOSUB 260: REM


355  IF  LEFT$ (OS$,1) =  CHR$ (1
     3) THEN OS$ =  RIGHT$ (OS$, LEN
     (OS$) - 1)
360  IF  LEFT$ (NS$,1) =  CHR$ (1
     3) THEN NS$ =  RIGHT$ (NS$, LEN
     (NS$) - 1)
```

Computer Station
12 Crossroads Plaza
Granite City, IL 62040
(618) 452-1860

COMPUTER STATION proudly offers a high-speed binary video digitizer for the Apple II called the **DITHERTIZER II**. The peripheral board uses a video camera with external sync to load the hi-res page of the Apple with any image the camera can capture. The **DITHERTIZER II** is a frame grabber, DMA type digitizer requiring only $^1/_{60}$th of a second to capture a binary image. Software supplied with the board enables building dithered images and capturing image intensity contours. Intensity and contrast are user controllable via the game paddles. Matrix size for dithering changable with one keystroke. Requires video camera. With external sync; recommended model, Sanyo VC1610X.

DITHERTIZER II, $300; B/W SANYO VIDEO CAMERA, $410; PACKAGE OF DITHERTIZER II AND CAMERA, $650.

GRAPHIC DUMPS: COMPUTER STATION offers the highest degree of human engineering on the market for hard copy graphics from the hi-res pages of the Apple. The following machine language dump routines are available for BASIC:

IDS440G/445G*	**$44.95**
IDS460G*	**44.95**
ANADEX 9501	**44.95**
NEC SPINWRITER 5510	**44.95**
NEC SPINWRITER 5520	**44.95**

*** Also availabe for use with Pascal, $44.95.**

GRAPHICWRITER: Hard copy of character sets found in DOS Tool Kit for use with Applewriter or print statements in your own programs. Requires DOS 3.3, DOS Tool Kit, one of graphic printers below:

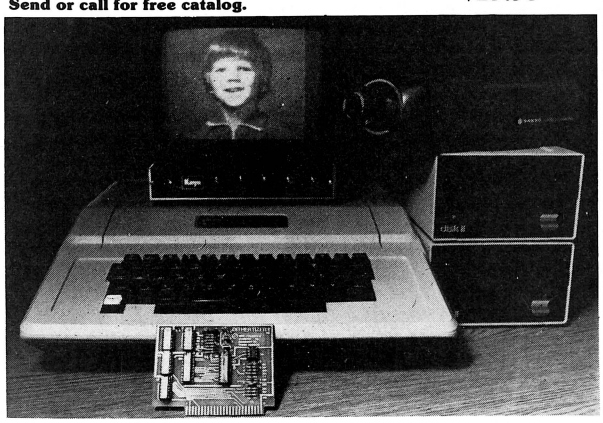
Silentype	**$34.95**
IDS440G/445G	**34.95**
IDS460G	**34.95**

VISILIST: Get hard copy of the FORMULAS used in VISICALC models. Prints grid location, contents (formulas or labels). and global parameters. Handy utility for all VISICALC users. **$24.95**

PROGRAMMER'S GUIDE TO THE APPLE II: Thick reference card (40 page booklet). **$4.95**

PROGRAMMERS HANDBOOK TO THE APPLE II: $29.95

Send or call for free catalog.



Apple II is a registered trademark of Apple Computer, Inc. VISICALC is a registered trademark of Personal Software, Inc. DITHERTIZER II is a registered trademark of Computer Stations, Inc.


```
400  REM

IF BOTH STATEMENT NUMBER AND
CONTENTS AGREE, THEN IGNORE

410  IF  VAL (OS$) =  VAL (NS$) AND
     OS$ = NS$ THEN 350: REM

500  REM

IF LINE NUMBERS ARE THE SAME BUT
CONTENTS DIFFER, THEN PRINT
"CHANGED" AND GET NEXT STATEMENT

510  IF  VAL (OS$) =  VAL (NS$) AND
     OS$ <  > NS$ THEN  PRINT : PRINT
     "CHANGED ";:J$ = OS$: GOSUB
     310: PRINT "TO ";:J$ = NS$: GOSUB
     310: GOTO 350: REM


600  REM

IF OLD LINE NUMBER < NEW LINE
NUMBER THEN OLD LINE IS DELETED;
FETCH NEXT OLD LINE


610  IF  VAL (OS$) <  VAL (NS$) THEN
      PRINT : PRINT "DELETED ";:J
     $ = OS$: GOSUB 310: GOSUB 21
     0: GOTO 410: REM


700  REM

IF NONE OF THE ABOVE, THEN THE
NEW LINE IS ADDED. PRINT "ADDED"
AND FETCH NEXT NEW LINE


710  PRINT " ADDED ";:J$ = NS$: GOSUB
     310: GOSUB 260: GOTO 410: REM



800  REM


WHEN ONE FILE IS EXHAUSTED,
DETERMINE WHICH AND INDICATE
REMAINDER OF OTHER "ADDED"
OR "DELETED", AS CASE MAY BE


810 X =  PEEK (218) +  PEEK (219)
     * 256: POKE 216,0: ONERR  GOTO
    910
820  IF X > 230 THEN 840
```

```
830   GOSUB 260: PRINT : PRINT "AD
      DED:";:J$ = NS$: GOSUB 310: GOTO
      830
840   IF  VAL (OS$) <  >  VAL (NS$
      ) THEN  PRINT : PRINT "DELET
      ED:";:J$ = OS$: GOSUB 310
850   GOSUB 210: PRINT : PRINT "DE
      LETED:";:J$ = OS$: GOSUB 310
      : GOTO 850: REM

900   REM

ASSUME END OF DATA ERROR IN
LAST FILE AND END PROGRAM

910   PRINT : PRINT "END OF COMPAR
      ISONS": PRINT D$"CLOSE"D$"PR
      #0": END : REM

1000   REM

PRINTER SUBROUTINES


1010   PRINT "PRINTER OR CRT? ";: GET
      PR$: PRINT PR$
1020   IF PR$ = "C" THEN PR = 0:LL
       = 30: RETURN
1030   INPUT "ENTER PRINTER SLOT O
      R CALL ";PR: RETURN
1100   IF  NOT PR THEN  RETURN
1110   IF PR < 8 THEN  PRINT D$"PR
      #";PR: PRINT  CHR$ (9); CHR$
      (26):LL = 70: RETURN
1120   CALL PR: PRINT  CHR$ (26):L
      L = 70: RETURN : REM

2000   TEXT : HOME : PRINT : PRINT


2010 TI$ = " PROGRAM COMPARER "
2020   HTAB 20 -  LEN (TI$) / 2:INVERSE
      : PRINT TI$: NORMAL
2030   PRINT : PRINT
2040   PRINT "     This program wi
      ll compare two ver- sions of
       a program and produce a lis
      t ofthe differences between
      them.  ";
2050   PRINT "Both     programs mu
      st be in textfiles; use  the
       program " CHR$ (34)"COMPARE
       CAPTURER." CHR$ (34)
2060   PRINT : PRINT "What is the
      name of the OLD version?"
2065   INPUT O1$:O$ = O1$ + ".T"
2070   PRINT : PRINT "What is the
      name of the NEW version?"
2080   INPUT N1$:N$ = N1$ + ".T"
2085   PRINT : INPUT "What is toda
      y's date? ";DT$
2090   PRINT : PRINT : GOSUB 1000
2100   HOME : VTAB 9
2110   IF PR THEN  PRINT "MAKE SUR
      E THE PRINTER IS READY, THEN
      "
2120   INPUT "         HIT [RETURN]
       TO RUN ";XX$
2130   TEXT : HOME
2140   GOSUB 1100
2150 XX$ = "**** COMPARISON OF **
      **"
2160   GOSUB 2500
2170   PRINT :XX$ = O1$: GOSUB 250
      0
2180   PRINT :XX$ = "and": GOSUB 2
      500
2190   PRINT :XX$ = N1$: GOSUB 250
      0
2200   PRINT : POKE 36,40 + 40 * (
      PR > 0) - (10 +  LEN (DT$)):
       PRINT "Prepared: "DT$: GOTO
      150
2500   POKE 36,(20 + 20 * (PR > 0)
      ) -  LEN (XX$) / 2: PRINT XX
      $: RETURN
```


KEYBOARD RECORDER

This program generates a keyboard file by monitoring and recording keyboard input. It can be used together with the System Monitor, BASIC or DOS, and is transparent to other programs receiving the input. The keyboard file can be selectively played back in a similar manner as the disk EXEC command to avoid repeated typing of the same input. A user may review the file on the screen, or save and load the file by using tape or disk. It saves time and storage space during program development and revision by keeping records of program changes or doing automatic program testing.

DISKETTE OR CASSETTE: $25.95 (with documentation; specify DOS 3.2 or DOS 3.3 for disk)

SINGLE DRIVE DISK COPY

This program makes disk copying easy and error free. It uses only one disk drive to copy some or all sectors on the disk.

DISKETTE ONLY: $11.95 (specify DOS 3.2 or DOS 3.3, APPLE II or II+)

Zeus Computing, A1
P.O. Box 712
Downey, Ca. 90241

****** COMPARISON OF ******

CREATE3

and

COMPARE CAPTURER

Prepared: 2/8/81

```
ADDED     10  REM


ADDED     COMPARE CAPTURER BY VAL J GOLDING


CHANGED   100 D$ =  CHR$ (13) +  CHR$ (0): INPUT "PROGRAM NAME TO CREATE TEXTFIL
          E FROM ";NAME$:FILE$ = NAME$ + ".T": PRINT D$"OPENLIST."NAME$;D$"WRITE
          LIST."NAME$: PRINT "LOAD";NAME$
TO        100 D$ =  CHR$ (13) +  CHR$ (4):OP$ = "OPEN":WR$ = "WRITE":CL$ = "CLOS
          E": INPUT "  PROGRAM NAME TO CREATE TEXTFILE FROM ?";NAME$:FILE$ = NAM
          E$ + ".T": PRINT D$;OP$"LIST."NAME$;D$;WR$"LIST."NAME$: PRINT "LOAD";N
          AME$


CHANGED   110 Q$ =  CHR$ (34):CMD$ = "OPEN":ZERO$ = " 0 D$=CHR$(13)+CHR$(4):PRIN
          TD$;" + Q$ + CMD$ + FILE$ + Q$:CMD$ = "WRITE":ZERO$ = ZERO$ + ":PRINT
          D$;" + Q$ + CMD$ + FILE$ + Q$:CMD$ = "CLOSE":PRNT$ = " PRINT " +  CHR$
           (34)
TO        110 DL$ = "DELETELIST.":TXT$ = ":TEXT:DEL0,0:END":CMD$ = ":HOME:POKE33
          ,33:LIST1-:":CH$ = "?CHR$(4)":PT$ = "?":C$ = ":":Q$ =  CHR$ (34):ZERO$
           = " 0 " + CH$ + Q$ + OP$ + FILE$ + Q$ + C$ + CH$ + Q$ + WR$ + FILE$ +
           Q$ + CMD$ + CH$ + Q$ + CL$ + Q$ + C$ + CH$ +


ADDED     Q$ + DL$ + NAME$ + Q$ + TXT$


CHANGED   120 ZERO$ = ZERO$ + ":HOME:POKE33,33:LIST:" + " PRINT D$" + Q$ + CMD$
          + Q$ + ":TEXT:PRINTD$" + Q$ + "DELETELIST." + NAME$ + Q$ + ":END"
TO        120  PRINT ZERO$: PRINT "RUN": PRINT D$"CLOSE"

CHANGED   130  PRINT ZERO$: PRINT "RUN": PRINT " 0"
TO        130  PRINT D$"EXECLIST."NAME$"


DELETED: 200  PRINT D$"CLOSE"

DELETED: 210  PRINT D$"EXECLIST."NAME$"

DELETED: 220  END


END OF COMPARISONS
```

2 SIDED DISKS

Edited by D. Buchler of *Mini'App'les* from material supplied by Dysan Corporation

Reversing Media on Single Head Flexible Disk Drive

Flexible Disk Drives (floppys) offer the end user low cost random access to data records. Prior to the introduction of the floppy, the only other alternatives were sequential tape cassettes, low cost one-half inch tape, or single cartridge hard media disk drives. The floppy disk has been an ideal peripheral for low cost systems.

There has been a tendency by some end users to economize by attempting to use the media on both sides in a single head drive. We most not lose sight of the fact that the value of the data stored on diskettes exceeds the cost of the media by a wide margin. Loss of data on either read or write means time delays, reconstruction of lost data, and customer dissatisfaction with the system, drive and/or manufacturer. All of this can be avoided in advance if the end user is made aware of the whys and why nots.

Head Shoe and Pad operation –

The relationship of the head to the media is such that when the jacket is properly inserted, and all interlocks are satisfied, the head is loaded on to the media on the recording side, and a felt loading pad is applied to the non-recorded side. In a Shugart disk drive this is the top of the disk. In normal operation, a gradual build up of oxide will accumulate on the pressure pad. There might even be some wear on the non-recorded side due to a scouring action of the oxide impregnated pad.

If the media is reversed, that is turned up side down, the scouring action will now occur on the prime recorded side, and the previously scoured side is now presented for recording. The recorded data is now subjected to an abrasive wearing by the contaminated load pad. Since this data is not being read, there is not any means of detecting the amount of wear or the loss of data. While a catastrophic failure might not occur, it is possible that some drop out or other read errors might go undetected. Worse yet, is the possibility that the error condition might be intermittent, which makes the entire operating system suspect.

Another adverse effect of reversing the media is caused by reversing the direction of rotation of the media against the jacket. This reversal of direction is apt to "break off" any build up of oxide particles. This presents a potential loose contaminant situation.

The net effect of this reversing (or flipping) action over a period of time, is to reduce performance and increase the probability of drop outs and errors.

Diskette tensioning

On most Floppy Disk Drives, when the diskette is properly inserted and operation has begun, pressure is applied to the jacket on both sides so that proper tension is created on the flexible media prior to the recording head. This also provides a wiping action of the liner material against the flexible media. When the jacket is reversed, or flipped, the direction of rotation is reversed, breaking loose any extraneous particles built up by prior wiping. Thus, reversing the media increases the probability of extraneous contamination and again increases the probability of errors.

Two head Drives

The above problem areas do not occur on two-head drives that are designed for two-sided applications. On a two-head drive, the pressure pad has been replaced by a second head mounted on a ceramic shoe. The operation now consists of a head-media-head relationship. The soft pressure pad with possible oxide build up has been eliminated.

The diskette tensioning apparatus is the same on one and two-head drives. Since media spin direction is not reversed by flipping, the oxide break off problem does not occur.

Summary

The foregoing summarizes the reasoning why Dysan and major OEM suppliers of diskette drives do not recommend two-sided media for one-head drive operation. Dysan feels the potential operating problems would make an unwarranted reflection on its reputation by using media in an unsuitable fashion. When IBM introduced the 3740 diskette, they intentionally interlocked reversal possibilities by offsetting the index hole from the centerline. IBM does not make a reversable diskette. DYSAN does test and supply two-sided media for operation in two-head (two sides) disk drives. Note that the standard diskettes are not certified for operation in two-head drives!

Some club members have reported problems in using both sides of a diskette. While those problems cannot be proven to be related to any of the above types of occurrences, it would seem wise to follow DYSAN's advice. Members have argued that the manufacturers are trying to sell more disks. However, not one of them has come out and recommended two side operation or even sold diskettes with two slots cut. Another argument heard frequently is that the member is using the second side for backup. Well you might get away with it, but I challenge anyone to religiously avoid using that reverse side except for the occasional update to create the backup. I myself plan to stick to one side from now on.

S. H. LAM ROUTINE UTILITY

by Lee Reynolds
Call -A.P.P.L.E. staff writer

S. H. Lam's Routine to set up machine language routines within a BASIC program has been used by various authors submitting to CALL -A.P.P.L.E. and other magazines. The usage of this routine is best covered in the article "Hiding Out in Basic", by Val Golding, in the June '79 issue of CALL -A.P.P.L.E.

I have often used this routine myself in my own programming. My usual method has been to: (1) create the machine code using my assembler, (2) assemble it into memory, and (3) go into the Monitor and laboriously copy down the hexadecimal values to be passed to Lam's routine in a string. Recently, however, I thought: why not let my computer save me some energy in step (3) above? This is my justification for the program whose listing accompanies this article. The program will automatically create a Text File containing the necessary statement(s) to call Lam's routine after setting up the string or strings it requires. The format of these statements in the Text File is:

(linum) A$ = (monitor string):
GOSUB (1st linum – 10)
CALL – 144

The value of (linum) above is asked for by the program, which will also ask the name of the binary file to BLOAD, BLOAD it, and assign a ".T" suffix to the name of the Text File to be created.

If more than one BASIC statement must be generated to process the machine language code, then subsequent line numbers increase by 10. (You could change this, if you wish, by modifying line 470.)

After you've used the program to create this Text File, you now need only EXEC it into your program that uses the machine language routine.

You will note that the name of the string variable passed as an argument to Lam's routine is A$. You could change this also, by modifying two lines in my program, namely 410, 430 and 470. (If you use Lam's routine in an Integer Basic program, remember that you must DIMension the string variable. As the program stands now, the maximum length of the string is 96 characters – not counting the extra 8 characters added on by Lam's routine.)

All functions are fully automatic, other than the specification of the line numbers to be assigned. The program will BLOAD the machine code, determine the starting address and length, write and save the Text File.

I hope that you will find this utility of value in your future programming.

```
10  REM

    S.H. LAM ROUTINE UTILITY

       BY LEE REYNOLDS
  CALL -A.P.P.L.E. STAFF WRITER

100  DIM HD$(16):D$ =  CHR$ (13) +
      CHR$ (4):Q$ =  CHR$ (34)
110  GOSUB 700: GOTO 200
120 J =  INT (BYTE / 16):HEX$ = H
     D$(J + 1):J = BYTE - 16 * J:
     HEX$ = HEX$ + HD$(J + 1)::::
      RETURN
130 BYTE =  INT (I / 256): GOSUB
     120: RETURN
140 BYTE = I - 256 * BYTE: GOSUB
     120: RETURN
150  PRINT : PRINT : INVERSE : HTAB
     8: PRINT " HIT ANY KEY TO CO
     NTINUE ": NORMAL : CALL  - 7
     56: HOME : RETURN
200  DATA  "0","1","2","3","4","5
     ","6","7","8","9","A","B","C
     ","D","E","F": FOR I = 1 TO
     16: READ HD$(I): NEXT I
210 MEMSIZ =  PEEK (978):MEMSIZ =
     (MESIZ + 13) * 256:MEMAD = M
     EMSIZ + 114:A1 =  PEEK (MEMA
     D) +  PEEK (MEMAD + 1) * 256
     :MEMAD = MEMAD - 18:A2 = A1 +
     ( PEEK (MEMAD) +  PEEK (MEMA
     D + 1) * 256)
220  VTAB 8: INPUT "BINARY FILE N
     AME? ";FILE$: VTAB 10: INPUT
     "BEGINNING LINE$? ";LINE$:SU
     B$ = LINE$:LINE =  VAL (LINE
     $) + 10:LINE$ =  STR$ (LINE)
230  PRINT D$"BLOAD"FILE$:FILE$ =
     FILE$ + ".T"
```

```
400 ADDR = A2:I = A1: GOSUB 130:A
    1$ = HEX$: GOSUB 140:A1$ = A
    1$ + HEX$
410 X$ = LINE$ + " A$= " + Q$ + A
    1$ + ":":L =  LEN (X$)
420  PRINT D$"OPEN"FILE$;D$"WRITE
    "FILE$
430 RT$ = Q$ + "ND823G" + Q$: PRINT
    " "SUB$;" A$=A$+"RT$": FOR I
     = 1 TO  LEN (A$): POKE 511
    + I, ASC ( MID$ (A$,I,1)) +
    128: NEXT : POKE 72,0: RETUR
    N "
440  FOR ADDR = A1 TO A2::BYTE =
     PEEK (ADDR): GOSUB 120:X$ =
    X$ + HEX$ + " ":L = L + 3: IF
    L < 100 THEN 490
450 X$ = X$ + Q$ + ":GOSUB" + SUB
    $ + ":CALL-144": PRINT X$
460  IF ADDR = A2 THEN 500
470 LINE = LINE + 10:I = ADDR + 1
    : GOSUB 130:X$ =  STR$ (LINE
    ) + " A$= " + Q$ + HEX$: GOSUB
    140
480 X$ = X$ + HEX$ + ":":L =  LEN
    (X$)
490  NEXT ADDR:X$ = X$ + Q$ + ":G
    OSUB" + SUB$ + ":CALL-144"::
     PRINT X$
500  PRINT D$;"CLOSE";FILE$: END
```

```
700  HOME : PRINT "S. H. LAM ROUT
     INE UTILITY": PRINT : PRINT
     "THE LAM ROUTINE IS A WAY OF
      STORING A   MACHINE LANGUAG
     E ROUTINE INTO MEMORY    FRO
     M INTEGER BASIC OR APPLESOFT
     .  SEE   THE ARTICLE "Q$"HID
     ING OUT IN BASIC"Q$" IN"
710  PRINT "THE JUNE '79 CALL -A.
     P.P.L.E.  YOU COULDWRITE A T
     EXT FILE CONTAINING THIS ROU
     T- INE, THEN EXEC THE FILE I
     NTO A PROGRAM.": PRINT : PRINT

720  PRINT "THE PRESENT PROGRAM W
     ILL LIKEWISE CREATEA TEXT FI
     LE CONTAINING THE LINES NEED
     ED TO EXECUTE THE LAM ROUTIN
     E, AS WELL AS  STORE THE STR
     INGS THAT IT REQUIRES.": GOSUB
     150

730  NORMAL : HOME : HTAB 6: PRINT
     "TO USE THIS UTILITY:": PRINT
     : PRINT " 1. BLOAD YOUR MACH
     INE LANGUAGE ROUTINE  INTO M
     EMORY BY ENTERING THE FILE N
     AME.": PRINT : PRINT " 2. EN
     TER THE STARTING  LINE NUMBE
     R TO   BE USED BY THE EXEC F
     ILE."
740  PRINT : PRINT " 3. EXEC THE
     NEWLY CREATED FILE INTO
     YOUR PROGRAM AND ENTER A LIN
     E THAT       WILL GOTO THE SE
     COND LINE OF THE LINES  CREA
     TED BY THE EXEC FILE."
750  PRINT : PRINT "NOTE:  A$ IS
     THE NAME OF THE STRING USEDI
     N THE EXEC FILE, AND MAY EAS
     ILY BE      CHANGED. LINE NUM
     BERS IN THE EXEC FILE  WILL
     BE INCREMENTED BY 10.": GOSUB
     150: RETURN
900  REM
```

BY LEE REYNOLDS & VAL J GOLDING

APPLE & PET

MAE

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

Now includes the Simplified Text Processor (STP)

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

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

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

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

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

SEND FOR FREE DETAILED SPEC SHEET

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

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

DOUBLE-SIZE GRAPHICS FOR THE SILENTYPE

by Bruce F. Field
Washington Apple Pi

Let me start by saying that I think the Silentype printer is potentially one of the most versatile printers on the market. Most conventional printers available for the APPLE control the printer operation via microprocessors in the printer with the printer functions coded in ROM. These instructions that tell the printer how to operate are hidden from the user, and in fact are unalterable by the user. You can only do as much as the manufacturer will let you do.

The Silentype on the other hand uses the microprocessor in the APPLE to control the printer functions with the instructions contained on a ROM on the printer interface card. What this means to the user is that the hardware functions of the printer (i.e. moving the printhead, printing a dot, and advancing the paper) are all controllable directly by the APPLE! We are no longer tied to the software provided by the manufacturer and are free to do anything we want within the mechanical constraints of the printer mechanism. WOOPIE!

Our only problem now is to find out how to control the functions of the printer. Well, APPLE "thoughtfully" didn't provide any information on the ROM software or the printer hardware to allow this. Letters to APPLE didn't produce any results either. Enter Sandy Greenfarb. Sandy also was trying to get information on the printer and obtained through IAC (International Apple Core) a 14 page writeup* on how to move the head and print a dot, and also a little on how the printer character set is defined and used. Just what I wanted. The result is the machine language routine presented here for printing either Hi-res graphics screen at double size.

*See p. 43 Winter Apple Orchard "Inside the Silentype Firmware".

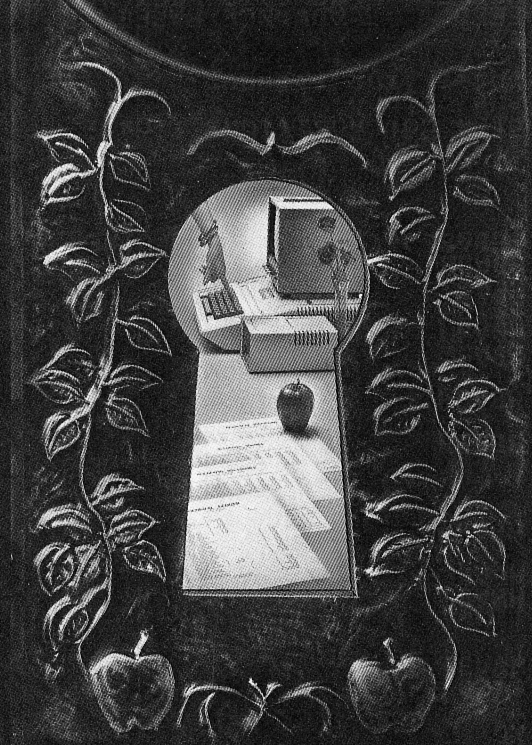
Using the routine is very simple; it uses the standard Silentype parameters to control the printing. You simply BLOAD the DBL GRAPHICS routine into memory (it resides from $300 to $3C7 and so won't conflict with Applesoft, Integer Basic, or DOS). You can generate or load your hi-resolution picture either before or after loading the DBL GRAPHICS routine. Then initialize the Silentype as you would normally (PR#1 from BASIC if your printer is in slot 1) and set up the printer parameters as desired. Instead of typing a control-Q to print the Hi-res screen, from BASIC you CALL 768 to activate this routine. The CALL can be placed inside a program or typed directly from the keyboard. If you are typing the CALL from the keyboard and don't want those words to appear on the printer, first reset the APPLE output to the screen (PR#0); the printer does not have to be "turned on" to print the graphics.

As it is written, the routine is slightly slot dependent and assumes your printer interface card is plugged into slot 1. It is very easy to change this if you use a different slot. The offending byte is at $30F hex or 783 decimal. This should be changed to $C0+printer slot (hex) or 192+printer slot (decimal). Once you do this, you can save it back to disk and not worry about it.

For those of you who are old hands at the APPLE and know how the Hi-res graphics screen works, you can go on to the next article; otherwise I'm going to try to describe how the routine works. First let me air a pet peeve. Most double size graphics printing routines I have seen (as for the Paper Tiger) chop off the sides of the screen when printed full double size because the picture is a little too wide for the printer. This comes about because the characters on the APPLE screen are 5 dots wide with a 2 dot space between them. Forty characters on the screen means 280 dot positions in the horizontal direction which is exactly the resolution of the Hi-res screen. Most dot matrix printers however, print characters 5 dots wide with a one dot space between. For 80 column wide printers this means 480 dot positions, not quite enough to print the graphics double sized. The obvious solution (at least for me) is to print the graphics screen sideways on the printer! The vertical resolution of the Hi-res screen is 192, times 2 is 384 which easily fits on a printer with 480 dot positions. So, guess which way my routine prints the screen?

Now that I have that off my chest, let's dig a little deeper into how the routine works. A rather poor map of the Hi-res screen is shown on page 21 of the new APPLE Reference Manual. (It would be handy to get it out now for reference.) This map shows an array of 40 boxes in the horizontal direction and 24 in the vertical direction. In the horizontal direction, each box controls 7 dots on the graphics screen. As explained on page 19 of the manual, bits 0 through 6 control the seven

It's Almost Obscene...



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

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

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

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

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

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

PROGRAMMA
PROGRAMMA INTERNATIONAL, INC.
2908 N. Naomi Street, Burbank, Ca 91504
(213) 954-0240
* Apple is a trademark of Apple Computer, Inc.

dots, with the least significant bit (bit 0) controlling the left-most dot of the group. In the vertical direction, things are not organized in an orderly fashion. Each box represents 8 bytes in memory, with each byte corresponding to a different vertical position on the screen, and the bits within the bytes controlling the 7 horizontal dots as explained above.

Now let's see if I can straighten all this out with an example. Suppose we want to know the address of the byte that controls the 87th dot from the left on the 23rd line from the top of the screen. Twenty-three divided by 8 is 2, with a remainder of 7, so we go down to the third vertical box from the top, which has an address of 8448 (decimal), and we want the 7th line in that box, which means we add 6144 to 8448. (We go to the third box, because box 1 controls lines 1-8, box 2 lines 9-16, etc., and we use the diagram in the lower right hand corner to find the number 6144 to be added to the base address.) Then we move over to the right to add in the contribution of the horizontal position. Eighty-seven divided by 7 is 12 with a remainder of 3. So we go over to the 13th box (which is labelled 12 because the numbering starts with 0) which adds 12 to the address. Now we total up these numbers to get 14604 which is the address of the byte we want. That wasn't so hard, was it? By the way, to get addresses for Hi-res page 2, we simply add 8192 to the addresses calculated above.

Since we now know how the screen works, let's print it. We are going to print it sideways, so we start at the lower left-hand corner, get a byte, print it, and work our way up; then move to the right one row and repeat. The way I'm doubling the graphics size is to print a cluster of four dots to represent one dot on the screen. This doesn't improve the resolution; it just makes the picture bigger.

The Silentype can print a line of 7 dots at one time. This is okay for regular size printing, but for double size we are going to have to make two passes to print what corresponds to one column of bytes on the screen. Each byte contains 7 bits of screen data which we are going to print as 14 dots wide. The first pass starts at the label NXROW1 in the assembly code, gets the byte from the screen, and looks up in a table what should be printed, based on the 4 least significant bits in the byte. This is printed twice, and the program loops around to get the byte from the row above. When this is completed, the paper is advanced, and the printer head returned to the left margin. The same process is repeated, except this time bits 4-6 of the byte are used. This completes the printing of the first seven dot columns of the screen; repeating this 40 times finishes the graphics dump.

There are just two other things to take care of, and actually these come first in the program. APPLE warns against running the printer and the disk at the same time as this overloads the APPLE power supply. To prevent this, there is a wait routine at the beginning of the graphics dump to allow the disk to timeout and stop running from any previous disk operation. Also, since the graphics dump doesn't respect the right margin, it is possible to set the left margin so that the print head is forced against the right side of the printer (bad, bad). Thus the left margin is checked and reset to a maximum of 18. Actually a left margin of 10 is my favorite, as it centers the graphics on the page.

```
                ********************************
3               *                              *
4               *  SILENTYPE  DOUBLE GRAPHICS  *
5               *                              *
6               *        BY  BRUCE FIELD       *
7               *     WASHINGTON  APPLE PI     *
8               *                              *
9               * Printer must be initialized  *
10              * with correct HI-RES page,    *
11              * inverse mode, left margin,   *
12              * and intensity.               *
13              *                              *
                ********************************
15              *
16              *

17  SLOT        EQU   $C100     PRINTER ADDRESS FOR SLOT 1
18  HBASL       EQU   26        FOR SCREEN ADDRESS
19  HBASH       EQU   27          COMPUTATION
20  XO          EQU   $2A       SCRATCH COLUMN VALUE
21  YO          EQU   $2B       SCRATCH ROW VALUE
22  LFMG        EQU   $CF11     PRINTER LEFT MARGIN
23  HPAG        EQU   $CF13     HI-RES SCREEN
24  DOTS        EQU   $CF2B     DOT IMAGE TO BE PRINTED
25  ROMS        EQU   $CFFF     CO-RESIDENT ROM SWITCH
26  INVRS       EQU   $CF14     PRINTER INVERSE FLAG
27  SFTLFT      EQU   $CD02     MOVE PRINTER HEAD TO LEFT MARGIN
28  PRNT        EQU   $CB0B     PRINT DOT IMAGE ROUTINE
29  FEED        EQU   $CCAB     ADVANCE PAPER
30  WAIT        EQU   $FCA8     MONITOR WAIT ROUTINE
31              *
32              *
```

```
                                ORG   $300
                                OBJ   $6300
                        *
                        *
                        * Wait 2 seconds for drive to turn off.
                        *
                        *
0300: A2 0D             DOUBLE  LDX   #13       COUNTER FOR WAIT LOOP
0302: A9 FF                     LDA   #255
0304: 20 A8 FC          LOOP    JSR   WAIT
0307: CA                        DEX            ;DECREMENT LOOP COUNTER
0308: D0 FA                     BNE   LOOP
                        *
030A: AD FF CF                  LDA   ROMS      SWITCH OUT CO-RES ROMS
030D: AD 00 C1                  LDA   SLOT      SWITH IN PRINTER ROMS
                        *
                        * If left margin >18, no room for graphics
                        * Reset margin to max of 18 if necessary
                        *
0310: A9 12                     LDA   #18
0312: CD 11 CF                  CMP   LFMG
0315: B0 03                     BCS   OK
0317: 8D 11 CF                  STA   LFMG      CORRECT LEFT MARGIN
031A: 20 02 CD          OK      JSR   SFTLFT    MOVE TO LEFT MARGIN


                        *
                        * Get each column of bytes on the screen
                        * and print them twice.
                        *


031D: A0 00                     LDY   #0
031F: 84 2A             NXCOL   STY   XO        XO IS COLUMN COUNTER
                        *
0321: A9 BF                     LDA   #191      INIT ROW
0323: 85 2B                     STA   YO
                        *
0325: 20 76 03          NXROW1  JSR   GETPT     GET SCREEN VALUE
0328: 29 0F                     AND   #$0F      MASK UPPER BITS
032A: AA                        TAX
032B: BD A8 03                  LDA   TBL1,X    LOOK UP DOT IMAGE IN TABLE
032E: 4D 14 CF                  EOR   INVRS     EXCLUSIVE-OR WITH INVERSE
0331: 8D 2B CF                  STA   DOTS
0334: 20 0B CB                  JSR   PRNT      PRINT DOT IMAGE
0337: 20 0B CB                  JSR   PRNT      TWICE FOR DOUBLE SIZE
                        *
033A: A5 2B                     LDA   YO        TEST IF ROW=0
033C: F0 05                     BEQ   NEXT
033E: C6 2B                     DEC   YO        DECREMENT AND CONTINUE
0340: 4C 25 03                  JMP   NXROW1
0343: 20 9F 03          NEXT    JSR   CRLF      DO RETURN AND LINE FEED


                        *
                        * Seven bits in each screen byte
                        * requires printing 14 dots on the
                        * printer, thus head makes two passes.
                        *
                        * This is the second pass
                        *


0346: A9 BF                     LDA   #191
0348: 85 2B                     STA   YO
034A: 20 76 03          NXROW2  JSR   GETPT     GET SCREEN BYTE
034D: 29 78                     AND   #$78      MASK LOWER BITS THIS TIME
034F: 4A                        LSR            ;SHIFT FOR TABLE OFFSET
0350: 4A                        LSR
0351: 4A                        LSR
0352: AA                        TAX
0353: BD B8 03                  LDA   TBL2,X    GET DOT IMAGE
0356: 4D 14 CF                  EOR   INVRS
0359: 8D 2B CF                  STA   DOTS
035C: 20 0B CB                  JSR   PRNT
035F: 20 0B CB                  JSR   PRNT
                        *
0362: A5 2B                     LDA   YO        TEST IF ROW=0
0364: F0 05                     BEQ   NEXT2
0366: C6 2B                     DEC   YO
0368: 4C 4A 03                  JMP   NXROW2
```

```
036B: 20 9F 03    106  NEXT2     JSR  CRLF
                  107  *
                  108  * One screen column completed
                  109  * Increment column and repeat
                  110  *

036E: A4 2A       111            LDY  XO
0370: C8          112            INY
0371: C0 28       113            CPY  #40        SEE IF ALL 40 COLUMNS DONE
0373: 90 AA       114            BCC  NXCOL      NO, CONTINUE
0375: 60          115            RTS            ;ALL DONE; EXIT HERE

                  116  *
                  117  * Compute memory address for HI-RES screen
                  118  * position; Y value in YO, X value in XO;
                  119  * leave with screen value in accumulator
                  120  *

0376: A5 2B       121  GETPT     LDA  YO
0378: A4 2A       122            LDY  XO
037A: 48          123            PHA
037B: 29 C0       124            AND  #$C0
037D: 85 1A       125            STA  HBASL
037F: 4A          126            LSR
0380: 4A          127            LSR
0381: 05 1A       128            ORA  HBASL
0383: 85 1A       129            STA  HBASL
0385: 68          130            PLA
0386: 85 1B       131            STA  HBASH
0388: 0A          132            ASL
0389: 0A          133            ASL
038A: 0A          134            ASL
038B: 26 1B       135            ROL  HBASH
038D: 0A          136            ASL
038E: 26 1B       137            ROL  HBASH
0390: 0A          138            ASL
0391: 66 1A       139            ROR  HBASL
0393: A5 1B       140            LDA  HBASH
0395: 29 1F       141            AND  #$1F
0397: 0D 13 CF    142            ORA  HPAG
039A: 85 1B       143            STA  HBASH
039C: B1 1A       144            LDA  (HBASL),Y GET SCREEN LINE
039E: 60          145            RTS

                  146  *
                  147  * Do line feed of 4 dot positions first,
                  148  * then do carriage return.
                  149  * This order reduces line stagger.
                  150  *
039F: A9 04       151  CRLF      LDA  #4
03A1: 20 AB CC    152            JSR  FEED       ADVANCE PAPER
03A4: 20 02 CD    153            JSR  SFTLFT     MOVE TO LEFT MARGIN
03A7: 60          154            RTS

                  155  *
                  156  * Tables for conversion of screen byte
                  157  * to dot image for Silentype.
                  158  *

03A8: 00 60 18    159  TBL1      HEX  006018
03AB: 78 06 66    160            HEX  780666
03AE: 1E 7E 01    161            HEX  1E7E01
03B1: 61 19 79    162            HEX  611979
03B4: 07 67 1F    163            HEX  07671F
03B7: 7F          164            HEX  7F
                  165  *
03B8: 00 40 30    166  TBL2      HEX  004030
03BB: 70 0C 4C    167            HEX  700C4C
03BE: 3C 7C 03    168            HEX  3C7C03
03C1: 43 33 73    169            HEX  433373
03C4: 0F 4F 3F    170            HEX  0F4F3F
03C7: 7F          171            HEX  7F


--- END ASSEMBLY ---
TOTAL ERRORS: 0
200 BYTES GENERATED THIS ASSEMBLY
```

Keep Ahead of Microcomputer Developments With America's Leading Authority

Interface Age is the most up-to-date source of microcomputer hardware and software advances. Whether you need to be informed for future purchases or to make comparisons, Interface Age should be #1 on your list.

- It has **more new product information** than any other small systems publication.
- Indepth hardware and software reviews
- Software and hardware applications
- Programming
- Robotics
- Book reviews
- Business applications
- Educational applications
- Latest technologies
- Tutorials

Take advantage of this no-risk trial subscription offer. If for any reason you are not completely delighted with the first issue, we will refund your payment in full. Order your subscription to INTERFACE AGE now with this convenient coupon.

NO-RISK TRIAL SUBSCRIPTION ORDER

12 issues for $18.00. That's a 40% savings off the single copy price. And if you are not delighted with the first issue, your payment will be refunded in full.

MAIL TODAY

- ☐ **One Year (12 issues) $18 (U.S. only)**
- ☐ **Two Years (24 issues) $30 (U.S. only)**
- ☐ **Canada/Mexico One Year $20**
- ☐ **Foreign Surface Mail One Year $35**
- ☐ **Foreign Air Mail One Year $50**

☐ **Check enclosed**
Must be in U.S. funds drawn on U.S. bank

Payment must accompany this order.

☐ VISA* ☐ master charge THE INTERBANK CARD ☐ AMERICAN EXPRESS Card

Card No. _____ Exp. Date _____

Signature _____

Name _____

Street _____

City _____ State _____ Zip _____

Make check or money order payable to: **INTERFACE AGE Magazine**
P.O. Box 1234, Dept. AO1, Cerritos, CA 90701
Allow 6-8 weeks for delivery.

MATRIX FUNCTIONS WITH THE APPLE

by Max J. Nareff
Cider Press

LOOPS AND TWO DIMENSIONAL ARRAYS

Some forms of BASIC provide the matrix commands (MAT READ, MAT PRINT, MAT INPUT, etc.) to facilitate display and manipulation of tables of numbers. Though the MAT commands are unavailable in Applesoft Basic, there is a substitute.

The term "matrix" is used here to denote a two-dimensional array of numbers consisting of horizontal rows and vertical columns. A matrix is a grid in which each of the elements (numbers) is given a reserved space or cell in the computer's memory.

To accomplish this, the matrix is first DIMensioned. Thus DIM A(12,12) allocates 144 cells in memory for the 144 (12*12) cells in the matrix. (Actually, since Apple begins dimensioning from zero, the number of elements in the variable is 13*13, or 169—but who's counting?)

Following is a simple program using double digits illustrating how a matrix can be constructed or read from data statements in Applesoft. For the sake of brevity, the number of rows is here limited to 3. Try more at your computer.

```
]RUN
10  20  60  70  90  11  88  42  32  77
41  66  19  20  91  72  72  18  36  24
11  30  93  46  66  33  14  98  16  32
```

```
0  DIM A(12,12)
10  DATA 10,20,60,70,90,11,88,42,
    32
20  DATA 77,41,66,19,20,91,72,72,
    18
30  DATA 36,24,11,30,93,46,66,33,
    14
40  DATA  98,16,32: REM

50  FOR I = 1 TO 3: REM
INDEX LOOP FOR ROWS

60  FOR J = 1 TO 10: REM
INDEX LOOP FOR COLUMNS

70  READ A(I,J): REM
READS DATA FOR MATRIX

80  PRINT A(I,J); SPC( 2);: REM

PRINTS, FORMATS MATRIX ELEMENTS

90  NEXT J
100  NEXT I
```

When the matrix is on screen, computations may be made in the immediate mode. For example:

PRINT A(1,5)*A(3,10)

With simple additions to the program all or selected matrix elements may be totaled, averaged, sorted, etc.

MANIPULATING MATRIX ELEMENTS

Above, MATRIX or table formation was demonstrated with the use of FOR-NEXT loops and double subscripted variables (also called two dimensional arrays). Now we deal with arithmetic manipulation of a MATRIX. Reference is made to the preceeding program which developed a table consisting of 3 rows and 10 columns of numbers. The following subprograms should be added to it.

1. SUM OF ALL MATRIX ELEMENTS

```
120 T = 0: REM
INITIALIZES THE TOTALER

130  FOR I = 1 TO 3: REM
LOOP FOR ROWS

140  FOR J = 1 TO 10: REM
LOOP FOR COLUMNS

150 T = T + A(I,J): REM
TOTALS ELEMENTS

160  NEXT J
170  NEXT I
180  PRINT : PRINT  TAB( 4);"SUM
     OF MAT ELEMENTS=";T
```

2. AVERAGE OF ALL MATRIX ELEMENTS

```
120 T = 0:K = 0: REM
INITIALIZES TOTAL AND COUNTER

145 K = K + 1: REM
COUNTS NBR OF ELEMENTS FOR AVGING

180  PRINT : PRINT  TAB( 9);"AVER
     AGE OF ELEMENTS=";T / K
```

3. SUM OF INDIVIDUAL ROWS & COLUMNS

```
200  FOR I = 1 TO 3: REM
INDEX LOOPS
210  FOR J = 1 TO 10
220  REM
CALCULATION OF ROW, COLUMN TOTALS

230 R(I) = R(I) + A(I,J)
240 C(J) = C(J) + A(I,J)
250  NEXT J
260  NEXT I
270  REM
LOOPS FOR PRINTOUTS

280  FOR I = 1 TO 3
290  PRINT  TAB( 14);"SUM OF ROW
     #";I;"=";R(I)
300  NEXT I
310  FOR J = 1 TO 10
320  PRINT  TAB( 14);"SUM OF COL
     #";J;"=";C(J)
330  NEXT J
```

The terms "MATRIX" and "TABLE" can be used interchangeably. In subprogram 2, line 145, the counter statement can also be expressed as K=I*J another form of iteration as the loop uncoils.

Any component of a MATRIX—whether single elements or rows or columns—can be manipulated as can several matrices. Try "PRINT" R(1)+R(2)"; "PRINT R(3)/10" and "PRINT C(2)+C(10)" or any combination required.

CROSSFOOTING

As defined by Logsdaon, (Programming in Basic, 1977) it is the mathematical process of summing the corresponding numbers (elements) of two arrays in pairs, one number from each array. The sum of each paired number appears in a third array. Originally used in computer card processing, the term has been broadened to include subtraction.

In the foregoing sections on Matrix simulation, the technique of CROSSFOOTING was demonstrated without being identified as such. Here now is a short application with a paired array.

Three service stations have each sold out their daily quota of two grades of gasoline (A & B). How many gallons has each outlet sold (C), and what is the grand total of company sales (T)?

```
5  HOME :T = 0: REM
INITIALIZES TOTALER

10  DIM A(3),B(3),C(3): REM
RESERVES MEMORY SPACE FOR
EACH ARRAY

20  PRINT "REGULAR","ETHYL","R+E
     SALE": PRINT
30  FOR X = 1 TO 3: REM
MACHINE LOOP FOR READING DATA

40  READ A(X),B(X): REM
READS DATA [A=REGULAR, B=ETHYL]

50 C(X) = A(X) + B(X): REM
CROSSFOOTING

60  PRINT A(X),B(X),C(X)
70 T = T + C(X): REM
COMPUTES TOTAL SALES 3 STATIONS

80  NEXT
90  PRINT : PRINT  TAB( 10);"TOTA
     L FUEL SALES= ";T
95  DATA 621.5,433.2,493.8,1217.5
     ,701.8,317.6: REM
FUEL SALES CAREFULLY ENTERED IN
PROPER SEQUENCE-- A,B,A,B, ETC.
```

MULTIPLYING MATRICES AND SCALARS

MATRIX statements, unavailable in Applesoft Basic, reduce the programming effort required to solve problems involving one or two matrices (tables) by eliminating the need for nested-loops. Large clusters of data can be managed quickly with single MAT statements. As noted previously, several of these statements can be simulated within Applesoft Basic, and while the effort is more tedious, selected mathematical operations can be performed using nested loops. Previously, the MAT READ and MAT PRINT statements were simulated, and a single matrix (table) generated. Several arithmetic operations were performed with its contents, both in the direct and indirect modes. Now, two small rectangular matrices, each composed of two rows and four columns of numbers, will be formed, and arithmetic interactions between them illustrated. MAT statements involving addition (or subtraction) and multiplication by a constant will be simulated.

Step 1 demonstrates formation of two (2×4) tables labeled MATRIX A & MATRIX B. Here again, the MAT READ & MAT PRINT statements are accommodated by use of nested-loops.

```
5  HOME
10  DIM A(2,4),B(2,4): REM
DIMENSIONS EACH MATRIX

50  PRINT  TAB( 15);"MATRIX A"
60  FOR R = 1 TO M: REM
ROW LOOP-- M=2

70  FOR C = 1 TO N: REM
COLUMN LOOP-- N=4

80  READ A(R,C): REM
FROM DATA STATEMENTS

90  PRINT A(R,C); SPC( 8);: REM
PRINTS AND FORMATS MATRIX

100  NEXT C
110  NEXT R
120  PRINT  TAB( 15);"MATRIX B"
130  FOR R = 1 TO M: REM (M=2)
140  FOR C = 1 TO N: REM (N=4)
150  READ B(R,C): REM
FROM DATA STATEMENTS

160  PRINT B(R,C); SPC( 8);
170  NEXT C
180  NEXT R
500  DATA 10,20,30,40,50,10,20,30

510  DATA 11,13,15,17,19,21,23,25
```

In step 2, a third MATRIX, C, will be formed by addition of A and B simulating MAT C=A+B.

```
20  DIM C(2,4)
200  PRINT  TAB( 8);"MATRIX C = M
     ATRICES A+B"
210  FOR R = 1 TO M
220  FOR C = 1 TO N
230 C(R,C) = A(R,C) + B(R,C): REM

MATRIX ADDITION

240  PRINT C(R,C); SPC( 8);
250  NEXT C
260  NEXT R
```

In step 3, scalar multiplication is demonstrated. In this type of operation, each element of matrix may be multiplied by either a constant, a variable or an expression. This step is an example of MAT D=(K)*A, where K is a constant.

```
30  DIM D(2,4
300  PRINT  TAB( 8);"MATRIX D = (
     K)*MATRIX A"
310  FOR R = 1 TO M
320  FOR C = 1 TO N
330 D(R,C) = (K) * A(R,C): REM
FOR EXAMPLE, LET K=5
```

The dimensions of the matrix on the left of the "equals" sign must be equal to or greater than those on the right side of the equation or else some of the elements may not appear; the shape of the new matrix may change, or more likely, the error sign will occur. Where the MAT capability exists, it is most useful and effective, but it is limited by its inability to add or subtract more than two matrices or to perform some other arithmetic.

SCORE ONE FOR THE APPLE: with the procedures outlined, many matrices can be added, subtracted, divided, etc. The interested reader is invited to try these and other options. Note what occurs where MATRIX A is multiplied by MATRIX B. The resultant matrix is produced by the multiplication of EACH element of A by the *corresponding* element of B [i.e. C(1,1)=A(1,1) * B(1,1), C(1,2)=A(1,2) * B(1,2) and so forth]. In a sense, this is a type of "scalar" multiplication where the multiplier varies from element to element. However, this is *not* true Matrix Multiplication in the language of computer science or matrix algebra, where the rows of one matrix are multiplied by the columns of the second and not by the corresponding elements.

Nailing Jelly to a Tree

Jerry Willis and Wm. Danley

Do you find programs in magazines, books and software houses that you would like to run on your computer, but they are written for another machine? This practical guide enables you to adapt the programs to your machine. It covers the ins and outs of computer math and logic, prepares you to understand and use programs written in machine and assembly languages and deals with converting existing BASIC programs.

ISBN 0-918398-42-8
$12.95

32 BASIC Programs for the Apple Computer

Tom Rugg and Phil Feldman

This new book is chock full of programs with practical applications, educational uses, games and graphics. Each of the 32 chapters fully documents a different program. Each chapter is made up of 8 sections that serve the following functions: purpose, how to use it, sample run, program listing, easy changes, main routines, main variables and suggested projects. If you enter the programs as they are shown, they will work. They are bug free!

ISBN 0-918398-34-7
$17.95

dilithium Press likes to make it easy

Our books are available from B. Daltons, Kroch's and Brentanos, computer stores or directly from us.

dilithium Press
P.O. Box 606
Beaverton, OR 97075

Write for free catalog!

PROGRAMMING THE APPLE TO COMMUNICATE BY TELEPHONE

by Terry Guerrant
First Software Co.

A MODEM is a device that allows one modem-equipped computer to "talk" to another modem-equipped computer over regular telephone lines, either locally or long-distance. The "talking" may be done with sounds (acoustic modems) or it may be done with pure electronic signals as with the Hayes Microcomputer Products, Inc. Micromodem II. You can communicate with another Apple computer or with a large, data-bank type computer.

The Micromodem II has been designed specifically for the Apple. It allows you to do more than just send and receive programs and data. The Micromodem II also allows you to control another computer from your own Apple over telephone lines.

Because you have control, you can: (1) operate a remote Apple without any help from a person at that remote Apple, just as if you were sitting at the remote Apple's keyboard; (2) use the computer after business hours to take advantage of the lowest-cost long distance telephone rates; (3) send or receive data, programs or messages when you are not physically present at your Apple; (4) enjoy the convenience and savings of not having to physically make delivery trips to transfer written materials or programs; (5) have branch business locations communicate freely with each other regarding inventory or other matters at any time over the telephone; (6) transmit material across the country by telephone and have a printer at the receiving end transfer to paper within minutes; (7) receive stock market prices with electronic accuracy whenever you wish.

The Micromodem II does not come ready to perform these tasks right out of the box. Programs must be written for it, and for the Apple, that dial the telephone, make a connection, control the flow of information, and provide for recovery from errors, if they should occur. I belong to a group of business users of the Micromodem II and the Apple. We have developed a group of 14 general purpose programs for the Micromodem II that cover most applications.

The experience of our group in developing our own Micromodem II programs indicates that if you are not already a proficient programmer in Applesoft, you should purchase a package of programs that already have the bugs worked out of them, rather than trying to write your own. If you decide to write your own programs, you will have to learn the Micromodem II's own language, create special sending and receiving programs that dynamically interface with each other, coordinate the activities of your Apple, your Micromodem II, the remote Apple, and the remote Micromodem II, and provide techniques for error recovery at both computers. Since you cannot physically see the remote Apple when you are operating it, getting the bugs out of programs can be difficult.

The members of our group found that the hardest problem for us to overcome was correcting errors at the remote Apple without losing control over it, and thus having to have a person at the remote Apple help in the error recovery procedure. The key to handling errors is to force the remote Apple to go into what is called "REMOTE CONSOLE MODE" whenever anything goes wrong. REMOTE CONSOLE MODE simply means that the remote Apple is the "slave" of your Apple and will respond to anything that you type on your own Apple's keyboard.

Sometimes a bad telephone connection will introduce static into your transmission which can confuse the remote Apple. Perhaps a file name is inadvertently misspelled, or a disk will fill up in mid-transmission. These and other types of errors make the error recovery routines of your programs their most important features. The actual transmission of programs or data can be relatively simple. It is how the errors are minimized and handled that requires real ingenuity.

As the main developer of the Micromodem II programs for our group, I soon realized that everyone in the group would have to be provided with complete instructions on the operation of the programs that were being written.

What started out as some sketchy instructions eventually evolved into a full-scale *Tutorial* complete with step-by-step instructions to follow. We began to get requests for copies of our programs from other persons with interests like our own, so a general purpose diskette was prepared which contained our 14 commonly used Micromodem II programs. To use these programs you must have a Hayes Microcomputer Products, Inc. Micromodem II, an Apple computer with firmware Applesoft language capability, 32K of memory, and at least one disk drive.

We now make the *Tutorial* and diskette available to people who need to make regular, practical use of the Micromodem II as we in the group do. For information you may write: First Software Co., 5622 E. Presidio Road, Scottsdale, AZ 85254, or telephone (602) 953-1214.

RESURRECTING KILLED PROGRAMS

by David G. McDonald
Apple Slice

A program is placed by APPLESOFT into memory starting with location $801. Each line of the program is stored in the following manner:

An address of 00 00 for the start of the next line denotes the end of the program and the beginning of the storage locations for numeric variables.

If a program is in memory and the "NEW" or "FP" command is typed, or APPLESOFT is entered in such a way as to reinitialize the program pointers (i.e. CTRL–B), the program remains generally intact, except you cannot "LIST" or "RUN" it. When specifying "NEW", or reentering APPLESOFT in such a way as to reinitialize it, the address locations, $801 and $802, which give the address of the second program line, are both set to zero, denoting the end of the program. Locations $69-6A, $6B-$6C, $6D-$6E, and $AF-$B0, which are the numeric storage pointers and the end of program pointer, respectively, are set up to give the address, $0803. Other than these changes, the program remains intact.

To restore the program to a useable form, we need to reverse these changes. Specifically, we need to:

1. Find the starting address of the second program line and store it in locations $801 (LS byte) and $802 (MS byte).
2. Find the address of the end of the program, add 1, and store it in locations $69-$6A, $6B-6C, $6D-$6E, and $AF-$B0.

Once this has been accomplished, the program is again in useable form.

The following assembly language program does just this (for a 4000 byte program, it takes 1-2 seconds). The main idea in using the program is to run it, then reenter APPLESOFT in such a way as to not clear the BASIC program again.

```
                ********************************
4               *                              *
5               * THIS PROGRAM WILL "RESTORE"  *
6               * BASIC PROGRAMS WHICH HAVE    *
7               * BEEN LOST DUE TO:            *
8               *                              *
9               *      (1) "NEW COMMAND        *
10              *      (2) RE-ENTERING APPLE-  *
11              *          SOFT WITH CTRL-B    *
12              *          OR, FOR DISK USERS: *
13              *          FP                  *
14              *                              *
15              *                              *
16              *  TO USE THE PROGRAM, WHEN IN *
17              *  APPLESOFT, TYPE "CALL 37632".*
18              *                              *
19              *  WHEN IN THE MONITOR, TYPE   *
20              *  "9300G" AND THEN A CONTROL- *
21              *  C TO REENTER APPLESOFT      *
22              *  WITHOUT REINITIALIZING      *
23              *  MEMORY.                     *
24              *                              *
25              *  WRITTEN BY:DAVID G MCDONALD *
26              *          MAY   1980          *
27              *                              *
                ********************************
29              *
30              *
31                        ORG   $9300
32                        OBJ   $6300
33              COUNT     EQU   $E
34              *
```

—— ——	—— ——	——.....——	00
Address of the Start of Next Line (2 bytes)	Line Number (2 bytes)	Program Line	End of Line

```
                        35  *
9300: A9 05             36  START   LDA  #05          INITIALIZE ADDRESS COUNTER
9302: 85 0E             37          STA  COUNT
9304: A9 08             38          LDA  #08
9306: 85 0F             39          STA  COUNT+1
9308: A0 00             40          LDY  #00
930A: B1 0E             41  SEARCH  LDA  (COUNT),Y   SEARCH FOR '00' BYTE
930C: F0 06             42          BEQ  REPLAC
930E: 20 58 93          43          JSR  INCR
9311: 4C 0A 93          44          JMP  SEARCH
9314: 20 58 93          45  REPLAC  JSR  INCR        RESTORE ADDRESS OF
9317: A5 0E             46  STORLS  LDA  COUNT       SECOND PROGRAM LINE
9319: 8D 01 08          47          STA  $801
931C: A5 0F             48  STORMS  LDA  COUNT+1
931E: 8D 02 08          49          STA  $802
9321: 20 58 93          50          JSR  INCR
9324: C8                51  TSTEOP  INY  TEST      FOR END OF PROGRAM
9325: B1 0E             52          LDA  (COUNT),Y
9327: D0 05             53          BNE  SLOOP
9329: C8                54          INY
932A: B1 0E             55          LDA  (COUNT),Y
932C: F0 0C             56          BEQ  EOP
932E: A0 00             57  SLOOP   LDY  #00
9330: 20 58 93          58  LOOP    JSR  INCR
9333: B1 0E             59  TEST    LDA  (COUNT),Y
9335: D0 F9             60          BNE  LOOP
9337: 4C 24 93          61          JMP  TSTEOP      GO TEST FOR END OF PROGRAM
933A: 20 58 93          62  EOP     JSR  INCR
933D: 20 58 93          63          JSR  INCR
9340: 20 58 93          64          JSR  INCR
                        65  *



9343: A5 0E             66  FINISH  LDA  COUNT       STORE LS BYTE OF:
9345: 85 69             67          STA  $69         BEGINNING OF SIMPLE VAR.
9347: 85 6B             68          STA  $6B         BEGINNING OF ARRAY VAR.
9349: 85 6D             69          STA  $6D         END OF NUMERIC STORAGE
934B: 85 AF             70          STA  $AF         END OF PROGRAM
934D: A5 0F             71          LDA  COUNT+1     STORE MS BYTE
934F: 85 6A             72          STA  $6A
9351: 85 6C             73          STA  $6C
9353: 85 6E             74          STA  $6E
9355: 85 B0             75          STA  $B0
9357: 60                76          RTS
9358: E6 0E             77  INCR    INC  COUNT       SUBROUTINE TO INCREMENT
935A: D0 02             78          BNE  RET         ADDRESS COUNTER
935C: E6 0F             79          INC  COUNT+1
935E: 60                80  RET     RTS



--- END ASSEMBLY ---

TOTAL ERRORS: 0

95 BYTES GENERATED THIS ASSEMBLY
```

LOOKIT

```
PROGRAM LOOKIT;

(* DISPLAYS COMPLETE 128 CHARACTER       *)
(*     TURTLEGRAPHICS CHARACTER SET      *)
(*                                       *)
(* FROM HARVEST VOL II, NO. 6, FEB '81   *)
(* NORTHWEST ILLINOIS APPLE USERS GROUP  *)
(*                                       *)
USES TURTLEGRAPHICS;

VAR I,J:  INTEGER
      A:  STRING[1];

BEGIN
  J:= 100;

(* INITIALIZE VERTICAL POSITION          *)
(* FOR STARTING LINE                     *)


  INITTURTLE;
  MOVETO(0,J);


(* PUT THE TURTLE 100 UNITS UP           *)
(* AND 0 UNITS LEFT ON HI-RES SCREEN     *)


 CHARTYPE(6);


(* NO MATTER WHAT YOU HAVE ON THE        *)
(* SCREEN, THE CHARACTER WILL SHOW UP.   *)


 FOR I:= 0 TO 127 DO
 BEGIN
  WCHAR(CHR(I));
  IF I MOD 40 = 0 THEN


(* DROP DOWN A LINE AFTER WRITING 40 CH *)


    BEGIN
     J:= J-10;
     MOVETO(0,J);
    END;
  END;
  READLN(A);

(* HOLD THE PAGE WHILE LOOKING AT THE    *)
(* CHARACTERS UNTIL YOU PRESS "RETURN"   *)

END
```

A NOTE ON CP/M

by Steve Jenkins

With the arrival of the Softcard there have been many questions concerning CP/M. CP/M stands for Control Program for Microprocessors and was written by Digital Research. If a program is in CP/M is it a BASIC program? (could be) CP/M differs from Apple's DOS in the fact that it is not language oriented. Microsoft's Softcard does more than just give you a Z-80 CPU. In providing the structure of CP/M, it allows programs the versatility to move from one CP/M system to another without any modifications. In my opinion this is the most important reason for CP/M. The capability of developing programs on the Apple for larger systems, and the inverse, adds new horizons to the software available for the Apple. And if languages are your cup of tea, most computer languages are available under CP/M.

At first you will find that CP/M is a little awkward, but remember your Apple was also difficult to operate when you first started. The best way to get acquainted with CP/M is to use it. Here is some help in learning the basic commands. First there are two types of commands—transient and resident. A resident command is in memory and an integrated part of the CP/M. A transient command must be loaded from the disk into memory, then executed (which means it must be on the disk). Some commands are:

RESIDENT COMMANDS

DIR: Gives Directory of Disk
Ex. DIR, DIR B:
REN: Renames a file Ex. REN NEW.NAM = OLD.NAM
ERA: Deletes a file Ex. ERA PROGRAM.NAM
TYPE: Copies a ASC file to screen
Ex. TYPE SAMPLE.TXT

TRANSIENT COMMANDS

STAT: Gives file and disk status
Ex. STAT, STAT *.*, STAT DSK:

PIP: Moves a file from one disk (or name) to another
Ex: PIP A:=B:*.* copy all of B to A
EX. PIP A:=B:*.COM copies all of the COM files on B to A

ED: CP/M editor (see manual for commands)
Ex. ED SAMPLE.TXT

FORMAT: Formats a blank disk
Ex. FORMAT B:

DUMP: Dumps a disk file to screen in hex
Ex. DUMP MBASIC.COM

Here are some more hints. Before you can write to a disk, the disk must be logged on (this is done by Ctrl-C). If you are copying a program from one disk to another, only the destination needs to be logged on (also remove write-protect tabs). Ctrl-P is a toggle to turn the printer on and off. CP/M operates with default drives; the command B: or A: changes the default drive but the command B:MBASIC runs the program MBASIC from drive B: and leaves A as the default drive. Only the programs that have the extent .COM can be executed from CP/M. These are just a few of the idiosyncrasies that need to be assimilated before you realize that CP/M is (in my opinion) the best operating system in micros today.

One book to read for a more complete list of commands and options is Rodney Zak's CP/M HANDBOOK. This book gives a complete list of commands in an easy to read format and is also good for the beginner.

Keep in mind that CP/M is not a language but an environment (operating system) where languages live (run), producing programs that meet needs.

```
>LIST
   10 REM

**************************
*   PROGRAMMER'S AID     *
* RAM TEST INITIALIZER   *
*  BY GARY M. COMEAU     *
**************************

   20 REM

THIS PROGRAM WILL INITIALIZE
THE PROGRAMMER'S AID RAM TEST,
ELIMINATING THE NEED TO ENTER
MEMORY ADDRESES MANUALLY


  100 VBL=0: DIM Y$(75)
  110 IN#0: PR#0: REM  KILL DOS HOOKS
  120 POKE 1010,105: POKE 1011,255
      : POKE 1012,90: REM
RESET VECTOR; DEFEAT AUTOSTART ROM
```

```
  200 CALL -10820: REM
INITIALIZE RAM TEST

  290 REM


ADJUST Y$ AS REQUIRED TO SUIT YOUR
SYSTEM MEMORY CONFIGURATION
SEE PROGRAMMER'S AID MANUAL PG. 31


  300 Y$="N400.4 800.8 1000.10 2000.20
       3000.20 4000.40 7000.20 8000.40
       34.00": REM 48K
  310 VBL= PEEK (218)+ PEEK (219)
      *256: REM
FIND Y$ IN MEMORY


  320 FOR I=VBL-5 TO (VBL-5)+ LEN(
      Y$)
  330 IF PEEK (I)=160 THEN POKE I,
      153: NEXT I: REM
REPLACE SPACES WITH CTRL Y'S


  400 FOR I=1 TO LEN(Y$): POKE 511
      +I, ASC(Y$(I)): NEXT I: POKE
      72,0: CALL -144: REM
POKE Y$ INTO KEYBOARD BUFFER
AND READ BUFFER


  500 REM

MEMORY TEST WILL RUN UNTIL YOU
HIT RESET


  510 REM

RAM TEST WILL OVERWRITE DOS;
REMEMBER TO REBOOT AFTER
```

TEXT TRAIN AND SUB SEARCH

IAC Software Reviews
by David B. Garson

Program: Text Train & Sub Search
Author: Bert Kersey
Distributor: Beagle Bros. Micro Software
4315 Sierra Vista
San Diego, CA 92103
Purpose: Entertainment
Language: Applesoft & Integer (32K)
Price: $16.00, Disk or Cassette, Manuals, and Catalog

RATINGS:
Speed: 85
Ease of Use: 90
Documentation: 75
Screen Display: 85
Reliability: 90
Overall Value: 95

In today's inflationary times, it is a joy to find a true software bargain. Imagine if you will, being able to get two pieces of software for only $16.00. In addition, a 36 page catalog/tip book. If you say impossible, then I direct you to Beagle Bros.'s Text Train and Sub Search.

Text Train, as the name implies, is a text-oriented race the clock game. It is your responsibility to control the train (actually a string of up to nine characters) by switching, coupling, and uncoupling. But be careful, if you hit an open switch you derail the train.

This game may bring back the good old days of playing with your HO train set. Your objective is to couple together a pre-defined series of freight cars (letters), and drive them to the check station in the fastest time.

The game begins with the Apple randomly placing letters both on the back of your train and on side tracks. It is then your job to get the freight cars in their proper order. All the while a clock is ticking away, making this game great for the individual (by beating their best time), or for a group at parties (competing against one another).

Text Train proves that a good game does not necessarily need graphics in order to be entertaining. Although the game could have been done with graphics, it may not have had the same appeal, since one does not always have text available to them in the graphic modes.

Sub Search is a variation on the old battleship game (remember 'B-5', 'Hit, you sunk my Battleship!). Done in lo-res graphics, the player must sink all the subs before his fuel and or oxygen runs out. By moving around the screen you are given a limited amount of information of your surroundings. As you home into your target, your sensors will begin to beep, telling you that you are getting close to a target. For a more detailed report you can use your scanners, but this costs fuel to use, so you can not use them too often.

Sub search is a fast paced game that will take some time before one can play it well. Sub Search will be liked by anyone who liked the original Battleship, although it was a little slow paced.

In addition to Text Train and Sub Search, there are two cute give-away programs on the disk. Another freebie that comes along with this package is the Beagle Bros. Catalog & Tip Book. This is a valuable book that contains quite a bit of useful information for any Apple owner. Look at the Nov.-Dec. 1980 Call-A.P.P.L.E. for a full review of the Tip Book.

And one can see, there is quite a bit to this software package. This is one of the few pieces of software that is actually under-valued for what you get (but don't tell Beagle Bros., or they might raise the price). The software is original and may teach you a few things that you may not know about your Apple. I don't think many people will be disappointed with this package.

WORD OF CAUTION*

by David E. Laden
Mini'app'les

Before you use a "CALL 62450" or a "CALL 62454" in your program, be sure you have initialized the contents of the high resolution pointers by executing HGR or HGR2. (CALL 62450 and CALL 62454, for those who don't know, clear the current high resolution screen to black and the most recently HPLOTted color, respectively.)

Depending upon the contents of the uninitialized pointer ($E6), a "CALL 62450" or a "CALL 62454" may erase all or part of the program in memory. This is especially true after a cold start (starting the Apple by turning it on) or making a disk-to-disk copy using Apple's copy program. The high resolution pointers do not seem to be initialized upon a cold start. Also, the copy program apparently uses some of the same zero page locations as Applesoft and does not return them to their original state.

NOTE: If you do not wish to initialize with HGR or HGR2 before you do a "CALL 62450" or a "CALL 62454" you can POKE a 32 ($20) into location 230 ($E6) for hi-res page 1, or for hi-res page 2 you can POKE a 64 ($40) into location 230.

*for those using Applesoft BASIC and high resolution graphics.

GETTING THERE FASTER IN APPLESOFT BASIC

by David Bartley
Apple-Dillo

I recently found an interesting quirk in Applesoft that explained why some of my programs took longer to run than I expected. By the time I worked it all out, I had come up with a few ways to speed up my programs and, surprisingly, found a reason to avoid resequencing their line numbers!

One of the more quoted clichés in computer programming is "a program spends 90% of its time in 10% of its code." Although the exact percentages vary, this is still a basic truth. Concentrate your efforts to speed up your program on the critical 10% where it spends its time.

It isn't always easy to find the critical sectors of code. Many hours have been wasted fine-tuning the wrong piece of code because a programmer "knew" where the time burner was. Things are especially hard in an interpretive language, where the cost of an operation is not always obvious.

You have to start somewhere, though, so an obvious candidate is an "inner loop", where the time to execute each statement is multiplied by the number of iterations through the loop. Nested loops are even more critical.

Observe the following boring but illustrative example of a loop:

```
100 FOR I = 1 to 10000
110 GOTO 120
120 NEXT I
```

Everything between the FOR and the NEXT is executed 10000 times. When preceeded by thirty dummy REM lines (which simulate a typical program), this simple loop consumes 38.1 seconds! Now look at this loop:

```
100 FOR I = 1 to 10000
200 GOTO 300
300 NEXT I
```

Although there is no apparent difference, this loop takes only 17.8 seconds (when preceded by thirty REM's). Why? I had to disassemble the GOTO handler in the Applesoft ROM to find out.

Applesoft, when executing a GOTO, first reads the line number and converts it to a two-byte binary value. It then performs a linear search to find the program line with that number. Generally it starts searching at the beginning of the program. This is why we are advised to put frequently used routines up front.

For forward jumps, however, it would be faster to start searching from the current position; this is what Applesoft does...almost. Here's the quirk I mentioned. *The interpreter looks at only the high byte of the line number. If that byte is greater than the high byte of the current line, then it commences the search from the current position. If it is equal or less, it starts the search from the beginning of the program.*

Since the low bytes are not compared, a forward jump to a line number whose high byte is the same will not execute as quickly as one where the high byte is greater. In our first example, the high bytes were INT (110/256) and INT (120/256), or 0 in both cases. In the second example, the high byte changed from 0 to 1, resulting in the difference in speed.

What did Apple (or Microsoft) save when they left out the comparison of the low-order line number bytes? Four machine language instructions, totalling eight bytes. It was not worth it! I've written a replacement GOTO handler that corrects that oversight, but before we get to that, let's examine some of our other options.

First, it's clear that we can work around the problem by choosing line numbers carefully. Any time we write a forward jumping GOTO, we should be sure that the line number we go to has a different upper byte. In practice, this is only important inside loops.

This strategy works, but it isn't easy. Furthermore, as soon as you resequence the line numbers using one of the utility programs floating around (such as RENUMBER), the problem may come back again.

An easier option, particularly for programs shorter than 300 or so lines, is to deliberately resequence with an increment that guarantees that all line numbers differ in their upper bytes. Any increment greater than 255 will do, but a multiple of 10 or 100 is more aesthetic. Don't use too large an increment. Divide 64000 by the number of lines in your program to determine the upper limit.

These options are unsuitable for large programs. It is almost imperative that you periodically resequence a large program as you write it, but eventually you have too many lines and too small an increment between them.

```
 1 (*****************************************)
 2 (*                                       *)
 3 (* Program to modify the BIOS modules *)
 4 (* to work with Dan Paymar's lower    *)
 5 (* case adapter for both input and    *)
 6 (* output.                            *)
 7 (*                                       *)
 8 (* Author    :Craig W. Vaughan           *)
 9 (* Date      :April 5, 1980              *)
10 (* Revision :1.0                         *)
11 (*                                       *)
12 (* REVISED   :LEE MEADOR                 *)
13 (* DATE      :DECEMBER 28, 1980          *)
14 (* REVISION :1.1 (FOR PASCAL 1.1)        *)
15 (* (NOTE: INPUT IS NOT SUPPORTED HERE *)
16 (*                                       *)
17 (* Copyright 1980 by Craig W. Vaughan *)
18 (* Permission is hereby granted for   *)
19 (* duplication for non-commercial     *)
20 (* purposes.                          *)
21 (* All other rights reserved.         *)
22 (*                                       *)
23 (*****************************************)
24
25 PROGRAM lcpatch;
26
27 VAR
28
29    blk: PACKED ARRAY[0..511] OF 0..255;
30    blt,bin:  INTEGER;
31    ptch : FILE;
32    S:STRING;
33
34 PROCEDURE block5;
35    BEGIN
36      bln := 5;
37      blt := BLOCKREAD(ptch,blk,1,bln);
38      blk[171] := 176;
39      blk[172] := 002;
40      blk[410] := 000;
41      blt := BLOCKWRITE(ptch,blk,1,bln);
42    END;
43
44
45 BEGIN
46    PAGE(OUTPUT);
47    GOTOXY(0,10);
48    WRITE('Name of disk? >> ');
49    READLN (S);
50    IF POS (':',S) > 0 THEN
51      S := CONCAT (S,'SYSTEM.APPLE');
52    ELSE
53      S := CONCAT (S,':SYSTEM.APPLE');
54    WRITELN;
55    WRITELN ('Updating ',S,'...');
56    RESET (ptch,S);
57    block5;
58    CLOSE (ptch,LOCK);
59 END.
```

MODIFYING PASCAL BIOS TO WORK WITH THE PAYMAR LC ADAPTER

by Craig Vaughan and Lee Meador

The following program is a lower case patch for Pascal version 1.1. Entry of lower case is as usual. E is used for a shift lock and W is used for a single shift. A keyboard which has been modified (in hardware) for lower case can enter lower case directly.

SIMPLE FUNCTIONS IN APPLESOFT

by D. B. Buchler
Mini'app'les

Few Applesoft programs seem to use the DEF capability. One neat application of DEF is to reduce the code needed to handle common usages of PEEKS and POKES. Two examples of this type of application are:

1. Simulate the INTEGER BASIC MOD function.

```
10 DEF FNM(X) = X – INT(X/256)
   *256
50 POKE 105, FNM(A) :REM
   PUTS LOW ORDER BYTE OF
   INTEGER PORTION OF A IN
   106
60 POKE 106, A/256 :REM HIGH
   ORDER BYTE OF INTEGER
   PART OF A IN 106
```

The above are equivalent to the INTEGER BASIC statments.

```
50 POKE 105, A MOD 256
60 POKE 106, A/256
```

2. Access 2 byte (16 bit) address or data in 2 memory locations.

```
20 DEF FNQ(Y) = PEEK (Y) + 256
   *PEEK (Y + 1)
50 PRINT FNQ(105) :REM
   PRINTS 16 BIT ADDRESS OR
   DATA STORED IN 105 & 106
```

The Newest In

Apple Fun

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

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

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

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

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

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

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

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

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

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

Order No. 0161AD $19.95

Solar Energy For The Home

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

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

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

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

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

Math Fun

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

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

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

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

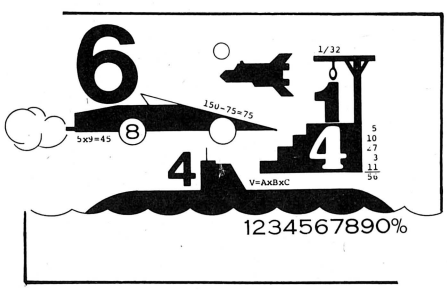
Car Jump—Make your stunt car jump the ramps. Each correct answer will increase the number of buses your car must jump over. These problems involve calculating the areas of different geometric figures.

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

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

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

Order No. 0160AD $19.95



Skybombers

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

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

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

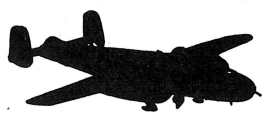
Flight personnel are sometimes forced to parachute from badly damaged aircraft. As they float helplessly to earth, they become targets for enemy missiles.

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

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

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

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



Paddle Fun

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

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

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

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

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

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

Order No. 0163AD $19.95

Instant Software™

*A trademark of Apple Computer Inc.

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

Apple* Software

From Instant Software

Santa Paravia and Fiumaccio

Buon giorno, signore!

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

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

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

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

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

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

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

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

TO ORDER

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

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

Apple Cassettes

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

We Guarantee It!

Instant Software Guarantee

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

10

Name _____

Address _____

City _____ State _____ Zip _____

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

Card No. _____ Exp. Date _____

Signed _____ Date _____

Order your Instant Software today!

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

Instant Software™ Inc.

Peterborough, N.H. 03458

CRAE 2.0

A FAST CO-RESIDENT APPLESOFT EDITOR FOR APPLESOFT PROGRAMERS, NOW PERFORM GLOBAL **CHANGES/FINDS** TO ANYTHING IN YOUR PROGRAM. OTHER COMMANDS INCLUDE: RENUMBER, APPEND, QUOTE (COPY), LIST (OPTIMIZED), MODIFY (SINGLE LINE), AUTO LINE NUMBERING, FORMATTED MEMORY DUMP, HEX/DEC CONVERSION, FREE SPACE, AND ABILITY TO ENTER MONITOR COMMANDS. CRAE 2.0 IS COMPATIBLE WITH NEIL KONZEN'S PROGRAM LINE EDITOR, REQUIRES 48K RAM APPLESOFT ROM AND DISK.

MCAT 2.0

MCAT 2.0 IS A FAST BINARY UTILITY WHICH CREATES A SORTED MASTER CATALOG WHICH IS SAVED ON DISK AS A BINARY FILE (FAST). THE MASTER CATALOG CAN BE EASILY UPDATED A WHOLE DISKETTE AT A TIME (ADD, DELETE, REPLACE). LIST/PRINT HAVE GLOBAL SEARCH CAPABILITY AND ONE OR TWO COLUMNS, PROVISIONS FOR DUPLICATE VOLUME NUMBERS. APPROXIMATELY 1200 FILE NAMES, 48K OR 32K, 13 OR 16 SECTORS DOS SUPPORTED.

CRAE on disk with 20 page manual
$24.95

MCAT on disk with 10 page manual
$19.95

CRAE and MCAT on one disk
$39.95 with manuals

EROM #1

CRAE's powerful Global Change/Find, optimized List Command, Hex to Decimal and Decimal to Hex conversion now available on a 2716 EPROM.

EROM #1 w/manual

$69.95

EROM #2

CRAE's Autoline numbering, formatted Memory Dump, Append, number conversion (Hex/Dec) on one 2716 EPROM.

EROM #2 w/manual

$49.95

EROM #3

CRAE's powerful renumber and quote function now on two 2716 EPROMS.

EROM #3 w/manual

$49.95

EROM 1, 2, 3
$149.95

ALL EROMS REQUIRE APPLESOFT ROM AND ROMPLUS+ AND ARE COMPATIBLE WITH NEIL KONZEN'S PROGRAM LINE EDITOR.

OLDORFS REVENGE	THE TARTURIAN	CREATURE VENTURE
48K, Applesoft ROM	48K, Applesoft ROM.	48K, Applesoft ROM.
ON DISK $19.95	**ON DISK $24.95**	**ON DISK $24.95**

SEE YOUR LOCAL DEALER OR SEND CHECKS TO
HIGHLANDS COMPUTER SERVICES

14422 S.E. 132nd
Renton, Washington 98055
(206) 228-6691

Washington residents add 5.3% sales tax. Applesoft and Apple are registered trademarks of Apple Computers, Inc.

Romplus is a registered trademark of Mountain Computers, Inc.

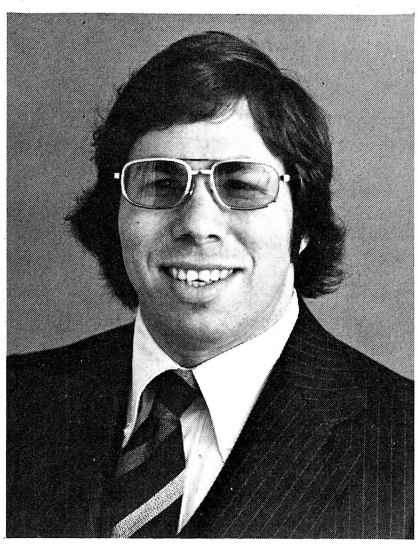
contact

apple computer inc.

the user group newsletter

HOMEBREW TO CHAMPAGNE

by Steve Wozniak
Apple Computer, Inc.



Steve Wozniak, *Co-founder Apple Computer, Inc.*

In a recent speech before the International Apple Core, Steve Wozniak, co-founder of Apple Computer, described how his involvement with a local computer club helped give birth to Apple.

I'd like to compliment the organizers from the International Apple Core for bringing together so many Apple fans. I hope you all love your Apple II's as much as I love mine.

Those of you who read the magazines are aware that the Homebrew Computer Club was formed back in spring of '75. It was one of the first computer clubs in the country. The phases it went through, from formation to maturity, parallel what I was doing at that time, which also led to the birth of the Apple I and Apple II. Both products to a large degree were born out of the club, rather than from an existing company or product philosophy.

At the time, I was working at Hewlett-Packard designing calculator-type circuits. I knew how to design and write compilers and such, but I'd kind of let that slip for a few years. I was more interested in calculator chips and driving around the world and just doing all the things you do at that age.

Well, I got involved in designing video games. I was in a bowling alley and saw a PONG game and thought, "Hey, I can design one of those." So I went out and designed my own version. It was famous around H-P because when you missed a shot, a random message would flash on the screen and say something like "Oh Shit!" or "Darn it".

I also got interested in video terminals, because I watched some friends using Teletype 33s, which were popular low-cost terminals in those days. I could talk to timesharing systems and play Lunar Lander and look at all sorts of interesting files.

I wanted a terminal, but couldn't afford one. At Hewlett-Packard they had a flexible policy which said, basically, that any engineer is welcome to parts for projects of his own design. So I took advantage of

that policy and designed a video terminal and a modem to get on various networks.

This was about the time the Homebrew Computer Club was formed. I heard from a friend that they were interested in microprocessors. I didn't really know what microprocessors were, but I decided to go, since they were also interested in video terminals.

We got together in Gordon French's garage. It was raining outside, but he had his garage door open and 40 of us were crowded inside. We introduced ourselves around the room. A good many of those people later started their own personal computer companies—companies like Processor Technology, Cromemco, and some of the other peripheral designing houses.

I learned a lot that night. I was a little bit behind these goings on. There was some discussion about whether we should start an 8008 portion of the club because Gordon French had one. He had a bunch of software he was writing for it and he passed out 8008 instruction manuals at the meeting. It's probably a good thing we never went in that direction, but focused more on the 8080, which was pretty expensive back in those days, about $270.

That was the formation of the club. Exciting things were happening and we would continue to meet.

At about that time, I had a video terminal and some video games. My terminal was almost complete. Lee Felsenstein, who later became our chairperson, also had a video terminal—it later turned into a Processor Technology board.

I was amazed by the high school kids at our meetings. Everyone knew the buzzwords, knew what was going on and which chips had what kind of instructions, saw all this home/hobby computer business through the Mark 8 to the ALTAIR. I didn't know anything. I had kind of drifted out of my early high school and college experiences in minicomputers, and I felt like a dummy. All these other people knew what was going on. I was certainly interested in what they were interested in. It sparked my interest in small computers, minicomputer-type instruction sets, and microprocessors.

Forming a newsletter is critical to the survival of a club, because that's how you get the word out and keep the interest up from month to month. The first couple newsletters of the Homebrew Computer Club were typed on one sheet of paper, just a list of names and addresses. We also set up a library. Gordon French offered to maintain a repository of software and articles and whatever else. He also offered his phone number between certain hours on certain nights. We could call him up and say: "What's going on? Do you have one of these?" It was really a large task for one person, so it kept him busy for much of the next year.

One of the early topics of discussion, while the club was being formed, was how home computers were about to happen. Everyone in the hobby clubs was saying it, but no one anywhere else was. At work, I would walk around and talk to guys in the lab, and everyone there was the way I had been, isolated from what was about to happen. Possibly some of the concepts of the hobbyists in those days were a little off base from what really happened, but it was obvious even then that the hobbyists were going in the right direction.

Byte magazine was the first to hit the stands. I was fortunate enough to have been one of the first people to buy it. People at Homebrew were talking about Altair Computers—who had one, who had ordered one, and when everyone was going to get one. Very few people in the club at that time had their own computers or could afford to build one.

The only way to have a computer was to build your own, which a few people had done. I was putting some of the finishing touches on my video terminal, and starting a file of all sorts of microprocessor-related information. There were only two microprocessors at that time, the 6800 and the 8080. Since I was familiar with minicomputers, the 6800 struck me as a more intelligent machine to go with.

I started collecting thousands of data sheets on terminals, printers, breadboarding accessories, and all the things that fit into the microprocessor world. Then the Homebrew Club went into a phase of very rapid growth. There's probably a few of you in this room who were there and remember it. It was extremely exciting. Each two-week meeting was the most important thing in your life. Rumors were flying. People were throwing ideas up in the air and guessing what was going to happen next.

Interface Age magazine was introduced about this time. It had started as a southern California computer club magazine. Magazines like Byte and Interface Age really helped make the whole thing happen on a national level, because everyone was getting informed. Without those magazines, it might have taken years for this thing to have become a full-fledged major industry.

A few other clubs were being formed around the country, and the magazines would list them. Now there are far more Apple Clubs alone than the total number of clubs in those days.

Very few companies were making hobby computers back then. MITS and SPHERE and a couple of others. These companies were all small enough so that they could afford to spend time with us. And since some of them were local, we were able to see some of their very first demos even before the equipment was in the stores. We would see some of the early equipment, like the first IMSAI.

A lot of the club members were interesting personalities. We had fellows who would ask around the room, "Is there anyone here from Intel?" If there wasn't, they would donate a bunch of Intel microprocessors to a raffle. We had our Rona Barrett types, too, who managed to keep the rumor mills spinning. I would say the rumors were 90 percent inaccurate and 10 percent accurate, but it was always interesting because they were digging up information.

Meetings were like a free forum; anyone could stand up and say what was going on. We had a few used chip salesmen, who were hocking chips at low prices and various other pieces of equipment like motherboards and extender parts. We'd have people who would get up and say, "I'd like to help anyone with hardware or software." And they usually got latched onto by someone who really needed a program written. There were people who would say, "Hey,

I need help in this area, can anyone help me or locate a part for me?"

If you think about it, the purpose of the club, more than anything else, was to facilitate communication. Lee Felsenstein, our chairman, organized the club so that we had what he called a "mapping period." Those who wanted something and those who had would stand up and say what their need or offering was. They would sort of "map" together and spot who they needed to meet with during our "random access period," which was a get-around to talk with people who had the same interests.

The Club grew quite quickly until we had about 550 members meeting twice a week. This was an extremely large number of people meeting for something that wasn't even making news.

People also set up demos of equipment they were working on. That was an opportunity for me to show off my video terminal. During this period I was designing a 6800 system, not having any hardware but knowing Hewlett-Packard had decided on the 6800 as their "standard" processor.

At the time, I could get some at a reasonably low cost—like 40 bucks per chip. There was no way to get an 8080 that cheap in those days. So I designed a system around the 6800, based on timing diagrams. When you're a computer designer, you have to look at timing diagrams and become intimately familiar with how many microseconds it takes to get from this phase to that event and such.

About the same time, which was late summer of 1975, a company called MOS Technology began making news by claiming they had a better processor than the 6800, that had been developed by the same people that designed the 6800, was hardware compatible with it, and was going to be sold over the counter (which was absurd) for $20. It was amazing, because in those days you always had to deal with major companies. Go through reps, stocking distributors, and the like. And here MOS was, selling microprocessors over the counter. But a lot of people were getting interested in this kind of thing.

I and a couple of other Hewlett-Packard people who were interested in getting microprocessor chips took this opportunity to visit the MOS Technology booth at a trade show and buy their new microprocessor—the 6502.

I can still remember the bus ride home. We were looking through the manual and discovering what the 6502 did and didn't have. That night, I went to a meeting of the computer club. The people from SPHERE came by and told us they were about to introduce the Micro-SPHERE, which they never really did. They also demonstrated a larger computer which was like a Nova. It had a hard disk with some color graphics running to a color TV display. In those days, on a hobby level, this was unheard of. I was very impressed by a color clock on the screen, and all the different colors, so I got interested in color graphics. At the time, it was more like, "Hey, this is really neat. But it's impossible to do because I know about TV, and it's difficult to just go out and design a color circuit."

Once I had the 6502, I started thinking about what I could do with it. I could have gone ahead and built a machine, but I was more interested in going through the manual and writing some code in the native machine language of the processor. I was used to assemblers but, unfortunately, there were no assemblers available. So I learned a technique that I didn't even think was possible: I just wrote the code down on paper, filling in the op codes and the displacements.

The first thing I wanted to write was a BASIC interpreter because it was a hot, catchy language and I wanted to do the first one for the 6502. It seemed like an interesting, worthwhile project, but it wasn't like it was going to be a product or something, just something to show off to the club.

I had never programmed in BASIC, although I knew it was so close to FORTRAN, which I could use, that I'd have no problem with it. BASIC seemed to be the favored language of the club people.

I'd have loved to write a compiler, but it was more important to get it written than to have a full language. I made tradeoffs to keep the code very small and went with an integer language. Because I had no assembler, I was also forced to have a loosely bound program with all the modules intercommunicating. With that kind of code, you need make only one change and then, everything has to be reassembled. I just could not take the time to reassemble the code myself by hand. That forced me into a structure where every new feature in the langauge was added on as a module, which actually made the development go very quickly and smoothly.

I got the interpreter developed and running on some simulators at H-P. Then I thought, "Hmmm-I don't have any hardware." As far as 6502s were concerned, nobody had anything in their hands except a chip. So I pulled out my own 6800 design, made the necessary changes to accomodate the 6502, and came up with a reasonable system that would get me to a point where I could connect the keyboard and the video display.

There were few processors in those days that had keyboard entry. This is what people were beginning to do with their ALTAIRS. I had already been through the front-panel switch stage and decided from day one that you really want a keyboard and video display.

The computer outfits back then did not have such things as monitor ROMs. It's a crucial thing to have, at least for an Apple II type machine. So I wrote a monitor. I kept my design small, so it only took me a couple of nights to get it wired together by hand and tested. I used my existing video terminal, which was a slow speed terminal but compact enough to stick on the same PC card as the 6502 processor.

So I got the computer together, and borrowed a four-case static card from a friend who had wired his own card together out of 2102s. And I managed to get it to play and got my compiler up, so I was able to sit down at the keyboard and test things out with a miniscule monitor. The circuit eventually became the Apple I.

What helped speed up the development was that every two weeks I could go back to the club and talk to people and say: "This is what I'm doing," and they would listen with interest.

In November, 1975, I took it down to the Homebrew Computer Club and ran BASIC for the first time. Of course, there were no cassette or even teletype loaders, so I

had to type in the text by hand. After about an hour, I had enough of BASIC working that I could run some interesting programs. Some high school kids who later became part of the early Apple crew would sit down and type out demo programs. A lot of people were interested because the computer was small and compact.

Shortly after this, I decided this 4K RAM board was not the way to go because everyone was starting to introduce 4K dynamic RAM chips. They were much more dense and compact than the 4K RAM board. So I got some AMI, three-transistor, cell design parts, surplus at a good price. The part was doomed not to be made very long. It was a 22-pin package, which meant you could connect all your address lines to the RAM, but you had to design special transistor circuits. You know, I'm a digital person. I like digital chips, but on these you had to design special clock drivers.

I brought the card up on those dynamic RAMs and it was very impressive to take it down to the club and show people I was using the new 4K chips. In the hobby world, you just didn't do anything like that when they were brand new.

Steve Jobs got me some and said to me "Why don't you use the 16-pin chips?" I looked at all the data sheets and found it was easy to interface because you didn't have to design an expensive clock driver and other types of logic to support the dynamic RAM.

I put these dynamic RAMS on the same card with the video terminal and processor. It was really quite a thrill to look at this entire board and know it was a usable computer. A lot of people were still talking about waiting to get their ALTAIRS and switches and maybe buy some peripherals someday.

About this time, those of us who were interested in the home computer revolution approached management in our lab at H-P and said: "Hey, look—with so much RAM, so much microprocessor, so much video, and a case, you can build a computer that talks BASIC and sell it for a thousand dollars." H-P had a lot of reasons why it couldn't be an H-P product. The lab manager said it was a great idea for a start-up company but wouldn't work for H-P. Later, I went through the legal department to get a release for the Apple I, which meant that no H-P division was interested in it.

Meanwhile, back at the Homebrew Computer Club, we had moved into a big auditorium. My favorite place was in the back row so I could set up my Apple I stuff while the show was going on. Everybody there was into this idea of the computer being a motherboard, a processor card, and a bunch of 4K RAM cards, and maybe a video card and an interface card. It was just not integrated enough and they looked just like all the minicomputers, with lights and HEX and switches on the front.

It was a big advantage to have a small, low-cost, integrated computer on one PC board, because people got a machine where they could talk to the screen and really do useful things in BASIC.

I started passing out schematics of the thing, since we didn't have a product or a company at the time. I'd go to people's houses and help them wire up their own. Steve Jobs said, "Why don't we just form a company and sell PC boards? You know, people really want these things, because hand writing is a major operation and a PC board is very inexpensive." A lot of people were offering PC board level computers in those days, so I decided we'd do it.

I figured we wouldn't make any money at it, but I sold my H-P65 calculator (partly because I knew they were coming out with the H-P67) and Steve sold his van, and that gave us our starting capital to pay a PC designer to design it. All the user had to do, once he got the board assembled, was plug in transformers, a video monitor, and a keyboard. In those days, it was unheard of for a hobby-type product to be that completely assembled.

Steve managed to sell some completely loaded boards, and all of a sudden we were in the computer business. The ALTAIR bus was still the big thing in the hobby clubs: People would look at Apple and say, "They're just a garage shop."

If this thing had been designed originally as a product, a lot of different decisions might have been made, such as to use the 8080 or go with the S100 bus, and a certain type of product that now makes sense in retrospect wouldn't have happened.

By the time we made our first delivery, we were in desperate need of a cassette interface. We ended up designing our own, and that pretty much gave us the Apple I system. If anyone has one, I wouldn't trade it in on an Apple II because it's a rare item now. We only built about 200 of them, and we only shipped about 175.

The Apple II came out shortly after. I had been thinking about how to put color on the Apple I and about carrying the designs and the concept a bit further. By the time I had color graphics up and running, I would take it down to the club every two weeks and show it to people.

We managed to get hold of some of the first 16K RAMs coming out, because of Steve Jobs's connections, and we shipped a couple of 32K Apple I systems using 16K RAMS. I believe they were the first computers ever shipped with 16K RAMS, because the manufacturers couldn't supply any of the big companies yet.

A lot of Apple I and Apple II enthusiasm was spurred by the positive feedback I'd gotten at the club. It was an interesting atmosphere in which to develop a product.

The main function of the clubs is to facilitate communication on various levels, and the International Apple Core appears to be a very good structure to disseminate information to everyone.

Question:

Why didn't you go with the S100 bus on the Apple?

Answer:

If the decision had been made: "Let's start a company and sell some products into this market," we would have gone with the S100 bus. What happened was that I was trying to build a small computer for myself. And when you keep it small and on one PC board, it just led to a more natural: "The microprocessor bus has enough information, why go with the 'standard' bus?"

It would also have meant going with an 8080 processor, which I didn't have and couldn't afford. If there had really been an S100 bus on the Apple computer it probably wouldn't have destroyed the Apple

concept. It just wasn't necessary, and there are enhancements to our buss, such as pre-decoded I/O, that turned out to be much more favorable. So I'm kind of glad we didn't, considering that of all the S100 companies around, just a few emerged as solid companies.

Question:

One of the things that makes a company really great is allowing someone with an idea, regardless of how off the wall it is, to play with it. Does that kind of thing exist at Apple or is the pressure reaching a point where you're on a project and that's it?

Answer:

That kind of development is very much encouraged. A lot of people at Apple like myself, Bob Bishop and some of the others, have just come up with off the wall concepts overnight or a weekend idea, a weekend project, and there's absolutely no problem unless it gets out of hand. I would encourage it as long as the normal work gets done. We have several specific examples where that does happen. A lot of our engineers come out of a club environment where they did things on their own anyway to begin with, which was also good.

Question:

Remember, that in the design days memory was really expensive and more characters meant more money. A 4K machine was a big machine and we couldn't afford to supply a video terminal with each Apple. We wanted to cut that cost down so we decided, hey, we'll just put video out and modulate it and go into a TV set and it's a lot less expensive besides, everyone has a home TV. You can't put more than 40 characters reliably on a TV, it's that simple, the band width of the RF units won't cut it. That's how the decision was made. If we knew we were going to do a system that was going to be used for small business, word processing and whatnot, the decision might have gone a different way.

Question:

Is it time to think about the fact that everyone's getting a VCR?

Answer:

It's funny, but the VCR market isn't growing as rapidly as anticipated, although it seems around here that it is. But you read the marketing reports and they're not selling many. Maybe in other parts of the country. Almost everyone I know has a VCR. I think that the VCR is a great mechanism. I think that there's room for some people to design some very simple circuits that can transfer data at very high rates with sufficient redundancy to store large data bases on a VCR and even randomly access it. VCRs all have the circuitry so you can rewind it, etc., so you might think of a VCR operating system. It's possible. It's feasible. One of the problems you get into is that, whenever you have that kind of operating system, you have to have gaps between data fields. Let's say you have a VCR you can start and stop. Well, the stopping has to be short relative to the data time or your tapes going to only be ten percent full of data. It turns out that in a few inches on video tape you can store far more than the amount of memory that your system contains. There are some problems but I wouldn't be surprised if it happens anyway. I don't know if you're aware of what CORVUS has done with their MIRROR system, but they back up their hard disk on video tape in 10 minutes. That's a $600 backup scheme so you don't have to use 100 floppies or buy another CORVUS drive to backup the one you have.

For Subscription Information see page 95

ROMPLUS* NEEDS ROMS FOR APPLES

RESIDENT FIRMWARE UTILITY PROGRAMS WILL EXPAND THE CAPABILITIES AND INCREASE THE USEFULNESS OF YOUR APPLE II OR APPLE II PLUS! *

***APPLESOFT EDITROM**

Global search, change, or remove any string, variable, literal, constant, or basic command word that appears in your Applesoft Programs. **EDITROM** uses no ram space that will interfere with your program. It does not reset any system pointers to protect itself and will operate with any size system - 16K, 32K, or 48K. After **EDITROM** has been initialized, the ampersand (&) command can then be used to call the **EDITROM** back for repeated use without readdressing the ROM + BOARD. Completely compatible with Konzen's Program Line Editor. If **PLE** is up, EDITROM will keep PLE up and allow joint operation.
Will operate with any version of **DOS** and requires FP in Rom. **35.95**

COMMANDROM

COMMANDROM Is like having a resident **'FID'** but with more operating features and conveniences. **COMMANDROM** will read a disks (13 or 16 Sector) File Directory Listing and display the following: A Command Menu, current drive number, number of sectors used and left, number of pages set up to hold all file names, the first page of file names, and an indentification letter next to each file name on display. Pressing any one of the command keys will load or run any file (A 'B' file load will display start and length addresses.), lock or unlock a file or all files, delete a file - with verify before deletion, change from one drive to the other, read a new disk, display a Track/Sector Map, change page numbers to view all file listings without recataloging, or exit to current language or monitor. No system pointers are reset and no RAM is reserved for **COMMANDROM.** Requires 48K, 3.2 or 3.3 **DOS** and **INT** or **FP** in **ROM** **35.95**

BASICSROM

Will boot a 13 sector disk on a system configured for 16 sector operation. The **BASICSROM** can be addressed on coldstart (without **Auto Start ROM** or warmstart (with **Auto Start ROM**) at any time **35.95**

DISK COPY/SPACE ROM

Duplicates a disk, from either **SINGLE** or **DUAL** drive, single or dual controller, 13 or 16 sector and with or without **VTOC.** Options include - Gross copy, active sectors only copy, **DOS** overwrite, auto boot of copy disk, free space on disk in sectors and kilobytes, and Init and volume number are selectable. Requires a minimum of 32K **35.95**

***APPLESOFT RENUMBER/ MERGEROM**

Made famous by Apple Computer Inc., this powerful utility will not disturb any part of a program in memory, when it is activated. Requires 48K, with or without Disk II. **35.95**

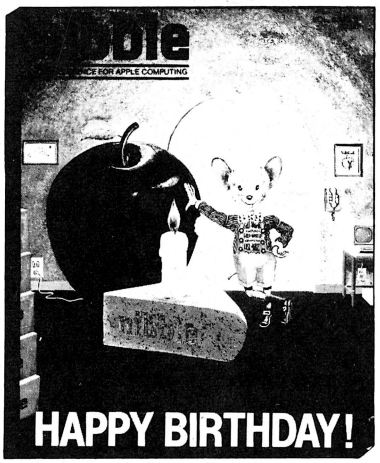
'YOUR' PLE ROM

Now you can put your Program Line Editor in **ROM.** Write for details. **$45.95**

*REGISTERED TRADEMARKS

BOX 599
WEST MILFORD, N.J. 07480
SOFT CTRL SYSTEMS

"NIBBLE® IS TERRIFIC" (For Your Apple)



NIBBLE IS: *The Reference for Apple computing!*

NIBBLE IS: One of the Fastest Growing new Magazines in the Personal Computing Field.

NIBBLE IS: Providing Comprehensive, Useful and Instructive Programs for the Home, Small Business, and Entertainment.

NIBBLE IS: A Reference to Graphics, Games, Systems Programming Tips, Product News and Reviews, Hardware Construction Projects, and a host of other features.

NIBBLE IS: A magazine suitable for both the Beginner and the Advanced Programmer.

Each issue of NIBBLE features significant new Programs of Commercial Quality. Here's what some of our Readers say:

– *"Certainly the best magazine on the Apple II"*
– *"Programs remarkably easy to enter"*
– *"Stimulating and Informative; So much so that this is the first computer magazine I've subscribed to!"*
– *"Impressed with the quality and content."*
– *"NIBBLE IS TERRIFIC!"*

In coming issues, look for:
☐ Stocks and Commodities Charting ☐ Assembly Language Programming Column
☐ Pascal Programming Column ☐ Data Base Programs for Home and Business
☐ Personal Investment Analysis ☐ Electronic Secretary for Time Management
☐ The GIZMO Business Simulation Game

And many many more!

NIBBLE is focused completely on the Apple Computer systems.

Buy NIBBLE through your local Apple Dealer or subscribe now with the coupon below.

Try a NIBBLE!

nibble
Box 325, Lincoln, MA. 01773 (617) 259-9710
I'll try nibble!
Enclosed is my $17.50 (for one year).
(Outside U.S., see special rates on this page.)
☐ **check** ☐ **money order**
Your subscription will begin with the next issue published after receipt of your check/money order.

Name _____
Address _____
City _____
State _____ Zip _____

NOTE
First Class or Air Mail is required for all APO, FPO and all foreign addresses with the following additional amounts

Air Mail Postage Rates 12-14 oz. x 8
Europe $32.00
Mexico and Central America $21.00
South America $32.00
Middle East $35.00
Africa: North $32.00
Central $43.00
South $43.00
Far East, Australia $43.00
Canada $18.00
APO FPO $7.50

THE APPLE II CASSETTE INTERFACE

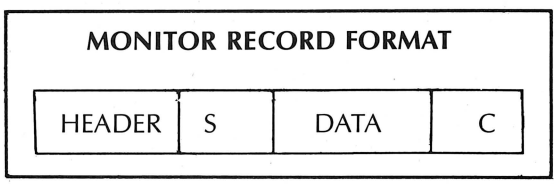
an IAC APNOTE
furnished by Apple Computer, Inc.

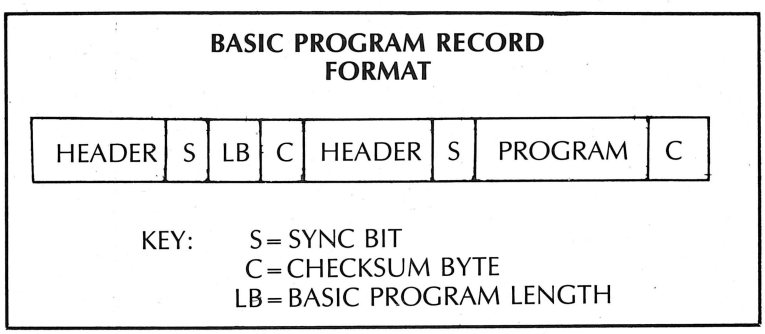
INTRODUCTION:

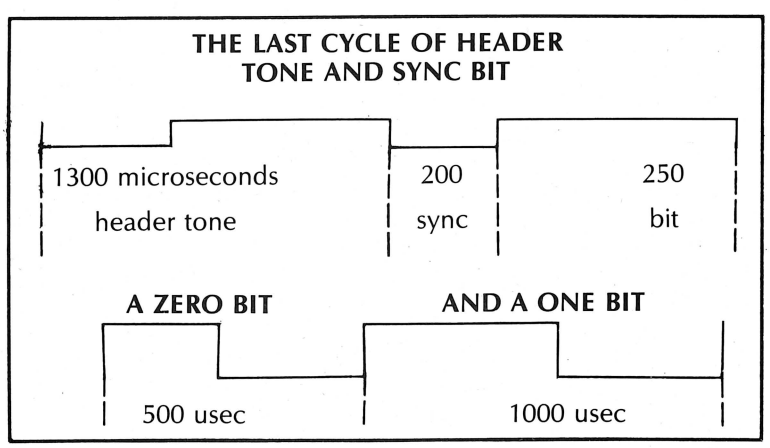
This note is to explain the cassette interface built into the Apple II and Apple II plus. Included is information on the format and use of the read and write subroutines. It is assumed in this note that the cassette recorder is in the proper mode, play or record, when the read and write routines are executed. Also note that the timing is approximate and may vary from one Apple to another.

RECORDS:

A record is a block of binary data. This data may be a BASIC or APPLESOFT program, a machine language program, or just binary data. Records representing BASIC or APPLESOFT programs are really two records, the length of the program and the actual program. A record consists of a header, sync bit, the actual data, and a checksum byte for error detection.







THE HEADER:

The header consists of 10 seconds of 770 Hz tone, (1 cycle equals 1300 microseconds). This is enough time for the cassette motor to get up to speed and the plastic tape leader to go by. There is also a shortened header between the BASIC length bytes and the BASIC program itself. This header is generated by a subroutine called HEADR. The value of the accumulator on entry controls the length of the header tone. This can vary from 0.2 seconds to 40 seconds. On entry, the X register should be 0, and the carry flag should be set. HEADR also generates a sync bit at the end of the tone. HEADR resides at hexadecimal address $FCC9, or decimal address -882.

THE SYNC BIT:

The sync bit is one half cycle of 2500 Hz, (200 microseconds) and one half cycle of 2000 Hz, (250 microseconds). It is used to signal the end of the header tone and the start of the data. It is generated by HEADR.

THE DATA:

The data is recorded on the tape with a low starting address and a high ending address. Each byte of data is shifted out most significant bit first, least significant bit last. A zero bit is made up of one cycle of 2 kHz, (250 microseconds per half cycle) and a one bit is one cycle of 1 kHz, (500 microseconds per half cycle). This works out to 2000 baud for zeros only and 1000 baud for ones, an average of 1500 baud.

THE CHECKSUM:

All during reading or writing, each data byte is EXCLUSIVE OR-ed with a checksum byte. This byte is written on the tape at the end of the data block. If the checksum computed during a read agrees with the checksum that was written out, then the data is probably good. This method will detect an odd number of errors for any of the eight bits of the byte.

WRITING DATA:

The cassette output circuitry is quite simple. It is a flip-flop connected through a voltage divider to the jack on the back panel of the Apple. Any time the address $C020 is accessed, this flip-flop changes state. Accessing the flip-flop once every 500 microseconds generates a 1000 Hz tone.

READING DATA:

The cassette input circuit is more complicated. It consists of a 741 operational amplifier configured as a zero crossing detector. That means that whenever the voltage at the input jack goes from positive to negative, (or negative to positive) the output of the amplifier switches from a 1 to a 0 (or 0 to 1). The detector is accessed by any read to address $C060. The sign bit (most significant bit) reflects the detector status. The read routines continually EXCLUSIVE ORs this bit with the value most recently read to detect a change in state. The amount of time required to change state indicates the incoming frequency, which then is used to determine if a one or a zero has been received. After detecting the first zero crossing at the start of a read, the read routine uses HEADR to generate a 3.5 delay then waits for the sync bit. It then reads the data and puts it in the specified memory range.

USING THE CASSETTE INTERFACE:

To either read or write all you need do is specify an address range and execute the read or write subroutine. The address range is stored in four bytes, two for the start and two for the end. In both cases the least significant byte is first.

FROM THE MONITOR:

If the start is $800 and the end is $9FF then

800.9FFW will write the data to the cassette and
800.9FFR will read it.

FROM MACHINE LANGUAGE:

Again if the start is $800 and the end is $9FF then store the address range.

```
LDA #$00
STA $3C     starting address low
LDA #$08
STA $3D     starting address high
LDA #$FF
STA $3E     ending address low
LDA #$09
STA $3F     ending address high
```

then JSR $FEDC to write the data to the cassette or JSR $FEFD to read from the cassette.

FROM BASIC:

First set up the address range. If S= the start and E=the end, then from Integer Basic,

```
POKE 60,S MOD 256
POKE 61,S / 256
POKE 62,E MOD 256
POKE 63,E / 256
```

or from APPLESOFT,

```
POKE 60,S – INT(S / 256) * 256
POKE 61,S / 256
POKE 62,E – INT(E / 256) * 256
POKE 63,E / 256
```

Then to write out to cassette CALL -307 or to read in from the cassette CALL -259.

INTERNATIONAL APPLE CORE™

APPLE ORCHARD BACK ISSUES

Back issues of Apple Orchard are available, while supplies last, as follows:

Volume 1, Number 1 — $5.00 each
All other issues — $3.50 each
(No. 2 is no longer available)

Please send your name, address, and issue number(s), along with a check, money order, or your VISA or MasterCard number and expiration date to:

Apple Orchard Subscriptions
P.O. Box 1493
Beaverton, Oregon 97075

***SYNTAX ERR

Several questions have arisen in connection with Applesoft Program Listing Formatter in the Winter, 1980-81 Apple Orchard. The EXEC file creator, which we apologize for omitting, appears below.

We have just finished re-testing the formatter on *our* system, and it works as published in the Orchard. Unfortunately, what's fair isn't always fair, meaning that it apparently does not work properly with *all* printer configurations, notably the Silentype. A number of modifications have been suggested, some of which seem to work, to one degree or another.

The problem lies in lines 6 and 28, with the CALL 65171 and CALL 65161, which do the monitor equivalent of a PR# 0 and IN# 0 respectively. The purpose of the IN# 0 is to disable Program Line Editor, should it happen to be in memory, so that the ESCape function of the list formatter would operate, and the PR# 0 was to return control of the output routine to the screen without printing a linefeed on the printer, which would occur if the PR# 0 was used, since it must be in the form of a PRINT statement, and DOS will not accept a PRINT statement ending in a semicolon. So the irresistable force has met the immovable object, almost.

Some possible solutions:

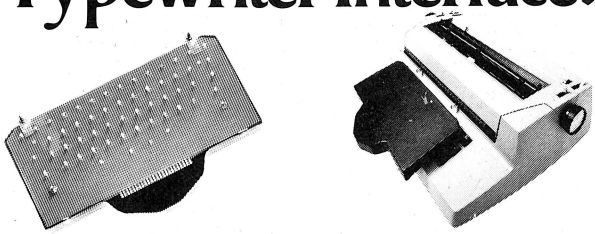
Remove the CALL 65161 outright (but make sure PLE is not in memory).

Change the CALL 65171 to PRINT D$"PR# 0" or

Follow the CALL 65171 with a CALL 976 (to restore DOS)

```
5  REM

EXEC FILE CREATOR FOR
PROGRAM LISTING FORMATTER

APPLE ORCHARD 3 * WINTER 1980-81

10 D$ =  CHR$ (13) +  CHR$ (4): PRINT
     D$"OPENFP LIST"D$"WRITEFP LI
     ST"
15   PRINT "POKE 0,PEEK(103):POKE1
     ,PEEK(104):POKE2,PEEK(175):P
     OKE3,PEEK(176)"
20   PRINT "POKE104,PEEK(176)"
30   PRINT "IF PEEK(175)=255 THEN
     POKE 104,PEEK(104)+1"
40   PRINT "POKE103,(PEEK(175)+1)-
     INT((PEEK(175)+1)/256)*256"
50   PRINT "POKE PEEK(103)+PEEK(10
     4)*256-1,0"
60   PRINT "POKE PEEK(103)+PEEK(10
     4)*256,0"
70   PRINT "POKE PEEK(103)+PEEK(10
     4)*256+1,0"
80   PRINT "RUN PROGRAM LISTING FO
     RMATTER"
90   PRINT D$"CLOSE"
100  END
```


At last...the Typewriter Interface!



Turn your electric typewriter into a low cost, high quality hard copy printer. 1 Year Warranty

Dynatyper—the patented* RDI—I/O Pak is fast becoming the industry standard for typewriter output. Why? Because:

1. It takes 2 minutes to initially install and 5 seconds to remove or replace.
2. You *do not* have to modify your typewriter. All factory warranties and maintenance agreements on your typewriter will be honored.
3. You can use it with *all* powered carriage return typewriters that have U.S. keyboard. Our Model I works with all *non* Selectrics and our Model II works with Selectrics. Conversion between models takes 2 minutes and the kit (26 plungers) is available for a nominal charge.
4. You don't have to lug around a bulky printer when you travel. If there is a typewriter at your destination, you can install the light (3 lbs.) I/O Pak in just 2 minutes.
5. Same interface for TRS-80, Apple and GPIB. Centronics and Pet compatible interfaces are available in third quarter 1980. Electric pencil available.
6. Delivery: Stock to two weeks. Price: $499. for the complete system, FOB Rochester, Domestic.

Over 1000 in operation today. VISA and MasterCard accepted. Call Ken Yanicky at 716-442-7804.

*Patent Pending

ROCHESTER DATA

3000 Winton Road South, Rochester, New York 14623 incorporated

THE MAILING LABEL AND FILING SYSTEM

From Avant-Garde Creations **only $24.95 ppd.**

This unique system will handle both your filing needs and your mailing label needs.

It's uniqueness starts with user-determined variables (up to 10 options) and continues with a special COUNT/SORT routine that allows the user to sort up to 9 VALUES for each of any 9 (out of 18) variables. It will print mailing labels, do a regular print-out or just display the criteria-meeting records while it counts them. It will also range-sort for 3 particular variables.

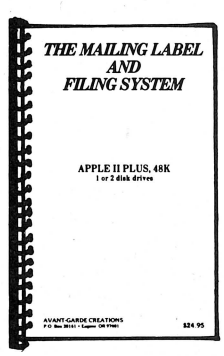
It makes an alphabetized directory of names and record numbers. You can find records by name or by numbers in seconds. If you don't know the exact spelling there's a quick-find option for directory-reading.

You can customize your labels and print up to 6 lines of your variables on them.

It includes special quick-copy and backup programs.

An easy to use system, brimming with options and dynamics, which ends the need for separate filing and mailing label programs.

Includes 40-page program manual and disk. APPLE II PLUS, 48K, one or two disk drives.



$24.95 ppd.
We accept VISA/Mastercharge

Write for our detailed brochure and more information:

Avant-Garde Creations
P.O. Box 30161
Eugene, OR 97403
Dept. ao
503-345-3403
Noon to 6 pm.

INTERNATIONAL APPLE CORE™

THE I.A.C. IS COMING TO CHICAGO

FOR THE SECOND ANNUAL GENERAL MEETING AND SEMINARS . . . SATURDAY AND SUNDAY MAY 2nd & 3rd

AT THE CHICAGO O'HARE MARRIOTT HOTEL IN CHICAGO

SATURDAY MAY 2nd
9 A.M. to 1 P.M.
GENERAL BUSINESS

Saturday meetings will cover general business and give the attendee the opportunity to become acquainted with the I.A.C. as well as the opportunity to participate in policy making decisions. This meeting is open to the public at no charge.

All member clubs are invited to send a representative to this meeting. Each club has a vote as to the future direction of the I.A.C.

SUNDAY MAY 3rd
9 A.M. to 5 P.M.
SEMINARS

Some of the speakers will be:

- **Steven Wozniak,** V.P. & Founder, Apple Computer
- **Phil Roybal,** V.P. Marketing, Apple Computer
- **Steven Jobs,** V.P. & Founder, Apple Computer
- **Mark Pump,** Northern Illinois Apple Users Group.
- **Vern Rayburn,** Microsoft, Inc.
- **Berry Yarkoni,** Apple III Product Marketing Manager.
- **Tom Woteki,** Washington Apple PI.

Tickets in advance — $5
At the door — $7.50
Limit first 1200 tickets.

REQUEST FOR TICKETS

NAME _____

ADDRESS _____

CITY/STATE/ZIP _____

___NO. OF TICKETS/$5 EA. TOTAL$ _____

MAKE CHECK PAYABLE TO:
INTERNATIONAL APPLE CORE
P.O. BOX 976
DALY CITY, CALIFORNIA 94017
PLEASE WRITE "TICKET ORDER" ON LOWER LEFT SIDE OF ENVELOPE

Following the I.A.C. Meeting
Monday May 4 thru Thur. May 7
THE NATIONAL COMPUTER CONFERENCE AND PERSONAL COMPUTING FESTIVAL AT McCORMICK PLACE, CHICAGO, ILLINOIS

Make plans to attend both the I.A.C. meetings and the N.C.C. Conference!

NEW PRODUCT PARADE

edited by Mark L. Crosby
I.A.C. New Products Editor

Hardware

REAL-TIME CLOCK – BSR CONTROL

THUNDERCLOCK PLUS is two systems on one peripheral card. It provides month, date, day-of-week, hour, minute and second. Selectable time formats: 24 hour or AM/PM ASCII string or numeric values. Software controlled interrupts are included. Battery provides backup power when the Apple II is turned off. It is also an interface for the popular BSR X-10 Home Control System. This board, with the Ultrasonic Interface Option, controls AC outlets through your BSR Command Console and Apple II. All 22 BSR commands are available for your program's use including 128 brightness levels – $139 with manual. Ultrasonic Interface Option with software disk and manual – $49.

Contact: Thunderware Incorporated, P.O. Box 13322, Oakland, California 94661 (800) 227-6204 ext. 307 – in California (800) 632-2131 ext. 307.

REAL TIME CLOCK

SUPERCLOCK II is a new Real Time Clock for the Apple II computer. This compact board plugs into any slot of the computer and provides time and calendar information as well as advanced interrupt capabilities. Other features include: timing from milliseconds to years, control firmware in PROM, write protect switch, and 12/24 hour format. The SUPERCLOCK is easily read from BASIC, Applesoft, and PASCAL. Will work with existing software written for the Mountain Hardware clock. Up to four software controlled interrupts can be used to allow foreground/background programming. Automatic updating of the Filer and each booted disk in PASCAL. Includes a long-life rechargable battery. Several utility programs are included. For example, one automatically maintains a date for each file in the Catalog. $159.

Contact: West Side Electronics, P.O. Box 636, Chatsworth, CA 91311.

SELECTRIC CONVERSION SYSTEM

This system permits you to connect your computer to any Selectric I, II, or III. Designed for both business and personal use for word processing, accounting and programming. Manual operation of the typewriter is not affected. A single-board computer gives precise control of functions and timing. Internal firmware features include form-feed, buffer hold, bell output, backspace and tab control. Parallel and RS-232C versions are standard. An adapter to IEEE-488 is available. Baud rates are switch-selectable from 110 to 9600. Prices range from $575 to $675, with OEM and dealer quantity discounts available.

Contact: ESCON Products, Inc., 12919 Alcosta Boulevard, San Ramon, CA 94583 (415) 820-1256.

CENTRONICS PARALLEL PRINTER INTERFACE

California Computer Systems has released the Model 7728 for compatability with printers using centronics-type interfaces (IDS Paper Tiger, Microtek MT-80P and MPI 88T, Okidata Microline 80 and Centronics printers). The 256 byte ROM driver responds to standard Apple II printer commands. Supports interrupt daisy chain with arbitration logic including IRQ generation and DMA daisy chain passthrough. 8-bit data output bus, four status inputs, data strobe and acknowledge handshake signals and a printer reset signal. $119.95.

Contact: California Computer Systems, 250 Caribbean Drive, Sunnyvale, CA 94086.

MOBILE APPLE POWER SYSTEM

The Mobile Apple System is a power supply designed to allow an Apple computer to operate from an automobile's 12 volt battery. The system is available for $110.00 with discounts available for larger quantities.

Contact: Dr. George Grinstein, CPU, Inc., 5161 Atlanta Highway, Montgomery, AL 36109.

80-COLUMN APPLE II CARD

The Full-View 80 is an 80×24 upper/lowercase plug-in card that provides 80-column capability while retaining the standard 40-character and graphics operating modes. You can select the different displays via keystroke or under program control. Two models provide either a 7×9 or 5×7 dot matrix (the latter suitable for low-performance monitors). Custom-designed character sets are available via EPROM. Full keyboard editing, complete cursor control and tabbing. Incorporates PASCAL and BASIC protocols. $395.

Contact: Bit 3 Computer Corp., 1890 Huron Street, St. Paul, MN 55113.

80-COLUMN CARD

SUP'R'TERMINAL is an 80 column by 24 line plug-in compatible board that includes lower case – all 128 ASCII characters. It is compatible with all Apple II peripherals. This card's features include a 5×8 dot matrix with inverse characters, shift lock, upper/lower case data entry at the keyboard, a user definable character set, compatibility with the Apple II Communication Interface, and fast scrolling and clearing. Sup'r'terminal's display is suitable for an inexpensive (8 MHz) monitor. All monitor-type escapes are valid – follows protocols of PASCAL and BASIC operating systems and incorporates control characters for both languages.

Contact: M & R Enterprises, P.O. Box 61011, Sunnyvale, California 94088.

DOUBLE VISION

Includes a full 128 character set with a fully programmable cursor, built-in light pen capability, inverse video, an 80 column by 24 line display with upper and lower case. This card permits you to have graphics on the Apple video output. Can use either the ESCape key or a one-wire modification to the shift key for upper/lower case entry. It is compatible with various word processing systems including Apple-Pie 2.0, EasyWriter Professional System, Text Tditor/Formatter. Transparent for use with BASIC and PASCAL.

Contact: The Computer Stop, 16919 Hawthorne Blvd., Lawndale, California 90260 (213) 371-4010.

Producers of hardware and software for the Apple line of computers should send news releases 3 months in advance to NEW PRODUCTS EDITOR, Apple Orchard, P.O. Box 976, Daly City, California 94017. The IAC cannot be held responsible for claims made by manufacturers.

SWITCHABLE ISOLATOR/INTERRUPTER

The ISOLATOR features input Spike/Surge Suppression together with three individually switched and filtered sockets. Total isolator load capability is 1875 watts maximum, with each switched, filtered socket capable of handling a 1KW load. Each switch has an associated pilot light. MODEL ISO-6 $128.95

The POWER INTERRUPTER disconnects power from controlled apparatus if AC line voltage is disrupted or exceeds user selectable limits. Front panel controls provide UNDER/OVER voltage interrupt level selection and Power Reset. Other features include integral Spike/Surge Supression and response delay to prevent false interrupts. The interrupter can accommodate a 15 amp resistive load or a 10 amp inductive load. MODEL PI-15-O/U OVER AND UNDER-VOLTAGE $142.95. MODEL PI-15-U UNDER-VOLTAGE ONLY $127.95.

Contact: Electronic Specialists, Inc., 171 South Main Street, Natick, MA 01760 (617) 655-1532.

SURGE PROTECTION

Various models of surge protectors are designed to protect equipment using standard 3-prong, NEMA 5-15P plugs. Nominal response time of one picosecond gives fast, transient surge suppression. Clamps any voltage rise of more than 10 above normal. Its rated dissipation is 600,000 watts at 100 microseconds. Prices range from $89.50 for standard duplex (120 or 220 volt) to $132.50 for a surge & dropout model. International, Terminal Strip, and Systems models available.

Contact: Advance Products, Manufacturers Representative for RKS Enterprises, Inc., 1086 Bush Street, Suite 302, San Francisco, California 94109 (415) 771-6953.

16K MEMORY CARD FOR THE APPLE II

RAMCard is a circuit card for the Apple II computer that provides an additional 16K of RAM to Apple users, converting a 48K to a 64K system. It is compatible with Microsoft's SoftCard peripheral and can be used with all software available for the SoftCard. When used with SoftCard, RAMCard provides a full 56K CP/M environment expanding the utility of Microsoft's CP/M languages for the Apple: FORTRAN-80, COBOL-80 and BASIC-80. It can also be used to provide additional memory space for other Apple software packages. Installation takes just a few minutes using step-by-step instructions. RAMCard requires an Apple II or Apple II Plus with 48K RAM. It cannot be used in addition to the Apple Language Card. $195.00

Contact: Microsoft Consumer Products, 400 108th Ave. NE, Suite 200, Bellevue, WA 98004 (206) 454-1315.

BAR CODE READER

The ABT BarWand, a modified Hewlett-Packard HEDS 3000 reader, simply plugs into the Apple II or III game I/O. This wand allows the familiar bar patterns to be read by the computer. Software packages are available to read Universal Product Code (UPC), to print and read our own Label Code and Applesoft programs in Paperbyte Code. The latter are two forms of bar code which can be printed with an inexpensive matrix printer. Sounds a scan tone when date has been read correctly. Warrantied for one year, includes a demonstration diskette. $195.00

Contact: Advanced Business Technology, Inc., 12333 Saratoga-Sunnyvale Road, Saratoga, CA 95070 (408) 446-2013.

ROM SIMULATOR FOR THE APPLE II

This simulator takes the place of a ROM while developing a program for a real ROM. It holds a program in the exact location the ROM version will ultimately reside and plugs into the destination ROM socket. The simulator can be used to increase the RAM available and replace one of the present system ROMs with RAM when used in an Apple II. The card contains the logic necessary to automatically switch control of the address and data buss from the Apple II to the Lamar Instruments Superkim (target) ROM sockets. $395.00

Contact: Lamar Instruments, 2107 Artesia Blvd., Redondo Beach, CA 90278 (213) 374-1673.

NEW DISK DRIVES FOR APPLE II

Micro-Sci has introduced two new Apple-compatible disk sub systems. The A-40 is a 40 track drive which provides 12.5 percent increase in capacity and 3 times the speed in track-to-track access. The A-70 has 70 tracks effectively doubling the capacity of a standard Apple II drive. The disk controller provides compatibilty with DOS 3.2, DOS 3.3 and Pascal systems. This is achieved with a jumper selectable boot prom. Disks written with the Micro-Sci drive cannot be read with Apple II drives, but it can read diskettes generated by the standard Apple II drives. Uses band positioning and standby power down circuitry yet draws no more current than an Apple II disk system. A-40 w/controller $495; wo/controller $395; A-70 $675 & $575, respectively.

Contact: Micro-Sci, Division of Standun Controls, Inc., 1405 E. Chapman Avenue, Suite E, Orange, CA 92666 (714) 997-9260.

INTERACTIVE VTR BOARD

The C.A.V.I. permits control of an industrial video recorder/player for random access of video tape segments. Software is provided to control all VTR functions, including audio/video switching. Optional software is available as a lesson authoring system. The system is designed to permit creating your own Computer Assisted Video Instruction program. BCD Associates will provide custom video tape production services. Requires Apple II with ROM Applesoft and 48K of memory, Disk II with controller, Color TV monitor, Industrial video Tape Recorder or Player. Includes interface card, control cables, operating software on disk, and instructions – $495.00 "The Instructor" Courseware Authoring System – $295.00

Contact: BCD Associates, 1216 n. Blackwelder Avenue., Oklahoma City, OK 73106 (405) 524-7403.

APPLE VIDEO RECORDING BOARD

The Adwar Apple Proc Mod is a circuit board which plugs into Slot #7 inside the Apple computer and processes the sync information. This permits ½ and ¾ inch video tape recording of the Apple signal. You can duplicate and edit such a tape without the usual loss of color. The Proc Mod brings the Apple signal closer to standard video tolerances so that video equipment can use the signal properly. This board stores an entire Apple video frame in solid state memory at the Apple scanning rate and then reads from that memory at the standard broadcast rate.

Contact: Adwar Video, 100 Fifth Avenue, New York, NY 10011 (212) 691-0876.

VIDEO DIGITIZER

The Dithertizer II is a peripheral board for the Apple II which utilizes a video camera with external sync to load the high resolution page of the Apple II with any image that can be captured with the camera. It requires only one frame (1/60th of a second) to capture a binary image. Software is included to build dithered (psuedo gray scale via half tones) images from multiple binary images and to capture image intensity contours using image subtraction. Contrast and density of the image may be varied with the game paddles and viewed on a monitor. Requires a video camera with external sync. Dithertizer II $300.00, B&W video Camera (Sanyo VC 1610X) $410.00, complete package $650.00.

Contact: Computer Station, 12 Crossroads Plaza, Granite City, IL 62040 (618) 452-1860, or, Peripherals Plus, 119 Maple Avenue, Morristown, NJ 07960 (201) 538-3385.

GRAPHIC TERMINAL

Fantastick-I is a multifunctional terminal that lets you draw and move patterns on the screen, and is also capable of analyzing drawn patterns. It is plug-compatible with the Apple II, consisting of a joystick, SW1-SW3 buttons, tenkey, I/O expansion connector, pilot lamp, and changeover switch on the reverse side. Software is included for various modes of drawing/data entry, subroutines provide slide, revolve, color control, count, painting, disk commands. $125.00

Contact: Hypersoft International, Inc., 3928 S. Sepulveda Blvd. #9, Culver City, California 90230 (213) 397-2274.

16 CHANNEL VARIABLE A-D BOARD

This board allows you to digitally store, analyze, display and print out your measurements. Can be used for measuring position, pressure, light, temperature. Could be used as a computerized volt-ohm meter. Has variable gain allowing increased measurements from 5 to 100 volts. Includes a complete software package with test kit, calibration method, and various applications. $179.95

Contact: Computer Technology Associates, 5812 Cromo Drive, Suite 102, El Paso, Texas 79912 (800) 854-2003 ext 815 – in California (800) 522-1500 ext. 815.

SPEECH SYNTHESIZER

The ECHO II speech synthesizer for the Apple II allows the creation of your own vocabulary with phonemes (word sounds) while using very little RAM memory (approx. 800 bytes + 20 bytes/word). Enhanced operating systems and vocabulary ROMs will be offered as they become available. This peripheral card fits into an available slot within the Apple II. Comes complete with speaker, instruction, manual, and a disk containing a speech editor, sample programs, and a sample vocabulary – $225.

Contact: Street Electronics Corporation, 3152 E. La Palma Avenue, Suite C, Anaheim, California 92806 (714) 632-9950.

Software

MICRO-PAINTER

This software package uses high-resolution graphics to "paint" pictures in 21 different colors on the Apple II. Micro-Painter includes dot-by-dot and inverse coloring. Pictures can be saved or displayed in any combination of colors or in an unpainted state. Pictures can also be repainted at any time. Micro-Painter is written in both BASIC and Machine Language for all Apple II computers. $34.95

Contact: Datasoft, 16606 Schoenborn St., Sepulveda, CA 91343 (800) 423-5630; in California (213) 894-9154.

INSTRUCTOR GRADEBOOK FOR THE APPLE II

Serendipity Systems is offering this tool for educators at every grade level. Records and reports individual and class performance for classes of up to 400 students, and statistically meaures the effectiveness of teaching and evaluation techniques. Offers various grading categories, e.g., test, homework, quiz which are then "weighed" by the user to produce final scores. Marks can be entered either as numbers or as letter grades. Reports include alphabetized class lists, blind grade listings by student I.D. for posting, individual performance reports and permanent class records. A 30 page self-teaching manual is included that is designed for the computer novice – $169.

Contact: Serendipity Systems Inc., 225 Elima Road, Ithaca, New York 14850 (607) 277-4889.

SCHOOL ADMINISTRATIVE PACKAGES

This system contains four modules, all available separately, which allow for teacher compilation of grades from class assignments and tests, the input and preparation of grades to print report cards and maintain student records, the preparation of reports to guidance counsellors for class scheduling purposes and the preparation of master school schedules and individual student schedules. Module 1, The Electric Gradebook, maintains assignment-by-assignment records of student progress – $49.95. Module 2, The Grade Program, allows the inputting of grades and test scores in order to prepare report cards and such sorted lists as honor rolls – $259.95. Module 3, the Counsellor Element, manages students records to prepare summary grade reports, file copies of grades and file folder labels – $89.95. Module 4, The Schedule Component, prints student course request forms, allows request form entry, accepts proposed course offerings, and prepares master school class schedules and individual student schedules – $259.95.

Contact: CompuSoCo, 26251 Via Roble, P.O. Box 2325, Mission Viejo, CA 92690

The "Assistant Principal" is a complete administrative package for high school and junior high schools. The package provides total control of class rosters, student master records, student schedules, teacher assignments, and grade reporting. Allows design of student input documents, student entry, class scheduling, accepts grades and test scores, prints report cards, file folder labels and prepares student master records. The system automatically prints ranked class lists and records attendance information. Requires two disk drives and Applesoft ROM. Includes seven diskettes with a 2-volume operating manual for $500.00. Operating manuals are available separately for $50.00

Contact: Monument Computer Service, Village Data Center, P.O. Box 603, Joshua Tree, CA 92284 (800) 854-0561. In California (800) 432-7257.

GENERAL PURPOSE PLOTTING PACKAGE

AppleGraph is a high-quality software package for plotting in a variety of formats for use by the business, professional, and research decision maker. Yields high-resolution, multicolor graphics for video display and hardcopy output. Featured are pie charts generated separately or bar graphs, area plots, points, and solid or dashed lines produced in any combination of overlays. Uses English language commands which may be entered interactively or in advance for automatic presentation of an entire data analysis complete with mathematical manipulation, curvefitting, smoothing, and simple statistics. Supports hardcopy on Silentype, Qume, Paper Tiger, and HIPLOT.

Contact: Business & Professional Software, Inc., P.O. Box 11 Kendall Square Branch, Cambridge, Massachusetts 02142. (617) 491-3377.

HI-RES GRAPHICS TEXT WRITER

With Applewriter Graphics, Apple users may now obtain hard copy of the character sets available with the DOS 3.3 Tool Kit. The graphics driver is transparent when used in conjunction with Applewriter; all the menus and options for editing and printing are still there, but now a variety of print "fonts" are available for the final draft. Requires 3.3 DOS, DOS Tool Kit, Apple parallel or Centronics interface card with other drivers to be available soon. May be used as a stand alone package for use with the print statements in programs. $34.95.

Contact: Computer Station, 12 Crossroads Plaza, Granite City, Illinois 62040 (618) 452-1860.

WORD PROCESSING SOFTWARE

SUPER TEXT II is an advanced word processing system for the Apple II. Contains new features like preview mode and SHIFTkey modification. Supplied with a backup copy of the program disk and newly rewritten documentation. This system will operate with the Apple II Plus with 48K and a disk drive. Offers compatibility with several 80×24 video boards, single key cursor control, automatic word overflow, character, word and line insertion and deletion scrolling, automatic paragraph indentation, ditto key, "the" key, block copy, save and delete, decimal alignment, justification, page numbering, centering, super/subscripting, on screen lower case with Paymar and Muse adapters, split screen, on-screen floating point calculations. Auto-Link feature provides unlimited file size when doing find, find and replace, preview or printing operations. – $150 (trade-ins up to $100 for old SuperText).

Contact: MUSE, 330 N. Charles Street, Baltimore, Maryland 21201 (301) 659-7212.

FINANCIAL ANALYSIS

Desktop PLAN—A Programming Language for ANalysis is a flexible tool which will allow you to customize your own financial reports. In a step-by-step process you develop report design and data values interactively. You then enter in calculation rules (e.g. "Multiply line 22 by line 77 save in line 23, columns 1-12"). The model you have then created may be re-executed as many times as necessary to test varying assumptions by changing input values. $99.95.

Contact: Personal Software Inc., 1330 Bordeaux Drive, Sunnyvale, CA 94086 (408) 745-7841.

DATA MANAGEMENT SYSTEM

The CC Data Management System stores and retrieves information and lends itself to "row and column" information. Files are easily created with varying types of fields which may also be computed fields (add, subtract, multiply, divide, exponentiate). A scan feature allows searching for records that have a field over, below or between a range of values. Update and addition of records is accomplished by prompts. Multi-diskette capability allows up to 85,000 characters per file. Sorting is permitted on up to 10 fields as keys. Will print reports, mailing labels, etc. $99.95.

Contact: Personal Software Inc., 1330 Bordeaux Drive, Sunnyvale, CA 94086 (408) 745-7841.

MULTIFIELD INFORMATION SYSTEM/UTILITIES DISKS

This data-base system is entirely menu driven. Includes user defined defaults, user defined restraints on fields, fast search and sort capabilities, easy editing features. Also permits combining a data file with a letter or document to allow creation of "personalized" letters. Requires an Apple II with 32 or 48K with Applesoft ROM and at least one disk drive. Compatible with DOS 3.3 Includes diskette and manual. $79.95

The MFI utility diskette extends the capabilities of the MFI described above. Allows you to recover accidentally deleted files, sort files that are too massive to load into memory, read diskette catalogs in an MFI file, perform user-defined statistical functions on MFI files, search and replace MFI files, and do formula calculations on numeric fields within an MFI file. $29.95.

BROWNPACK 1 is a utility package for use with Applesoft BASIC in ROM. These routines include several machine-language programs which give your BASIC programs such things as PRINT USING capabilities, superfast machine language sorting (1000 string items and a numeric array in 5 seconds), automatic diskette menu, packing & unpacking of numbers, HI-RES shape utility programs, and much more. Fully compatible with DOS 3.3, includes eleven utility routines on diskette, instruction manual and quick-reference chart. $39.95.

Contact: The Computer Emporium, 3711 Douglas, Des Moines, IA 50310 (515) 279-8861.

PAYROLL PACKAGE

The PAKRE Payroll Package consists of five integrated programs: Create/Update Employee File; Enter/Write Checks; Examine Payroll Information; Examine Payroll Checks, and Print Tax Forms. It will store employment records for approximately 70 employees and payroll information for approximately 935 checks per year (weekly payroll for 18 current employees) when used with DOS 3.2. compatibility with DOS 3.3 allows a still larger capacity. Permits entry of previously paid payroll, prints several summary reports, and information for 941-A and W-2 forms—$150.

Contact: Computerized Service Station Systems, 5230 Clark Avenue, Suite 12, Lakewood, California 90712 (213) 866-2581.

PAYROLL SYSTEM

The LMA payroll system, which was field tested at small businesses in California, New York City, Indiana and Florida, comes in two configurations—single and dual disk drive, and effectively performs full payroll functions for up to 45 and 150 employees, respectively. Features user-changeable tax rates, complete source listings in BASIC, fully documented file layouts for customization and a menu-driven control system. Handles payrolls in which employees have different frequencies of pay, multiple-states and complex city taxes, full personnel records, time card entries with overtime, 941A Federal listings, W-2 forms, paychecks and paylistings, and summary reports. Requires an Apple II Plus. $349.00.

Contact: Your local computer store—or—Lenz, Masterson & Associates, Inc., 684 Haddon Avenue, Collingswood, New Jersey 08108 (609) 854-1333.

COMPUTERIZED JOB CONTROL SYSTEM

This system offers job costing and reporting to provide management with reliable measures of productivity. Furnishes instant job status checks for determining exact work-in-progress figures. Combines information on job orders, estimates, labor hours, material costs and service costs to produce several valuable reports. Profit/loss values and variances are given to allow fine-tuning. Can be customized for individual businesses and allows for as many as 400 jobs in progress. Includes users manual and tutorial program. JCS is written in PASCAL and requires a 48K Apple II with three disk drives and a 132 column printer. $750.00

Contact: High Technology, Inc., 8001 N. Classen Blvd., P.O. Box 14665, Oklahoma City, OK 73113 (405) 840-9900.

PRACTICAL BASIC PROGRAMS

An entire collection of the 40 programs featured in the book, PRACTICAL BASIC PROGRAMS. Available on diskette for the Apple II computer, the programs are taken from common applications in four general categories: Business, Statistics, Mathematics, and Miscellaneous. They deal with such subjects as decision analysis, checkbook reconciliation, statistical techniques, and Federal Tax form preparation. A book is included which presents each program with a description, sample run, practical problems and a BASIC source listing. Requires a 32K Apple II with one disk drive. $40.00

Contact: High Technology, Inc., 8001 N. Classen Blvd., P.O. Box 14665, Oklahoma City, OK 73113 (405) 840-9900.

Miscellaneous

DIRECTORY OF APPLE SOFTWARE

Included in this volume are more than 800 Apple programs and many peripherals collected from over 100 vendors. There are 364 pages. Each listing includes program title, publisher, memory requirements, program description and price. Language and hardware requirements are often included as well. Also included are a summary of Apple control and editing characters and disk commands and a glossary of computer terms. Listings are arranged alphabetically—an index lists by category. Skarbeks Software Directory, 364 pp; $11.95.

Contact: Skarbeks Software Directory, 11990 Doresett Road, St. Louis, MO 63043 (314) 567-3292.

HEAD CLEANING DISKETTES

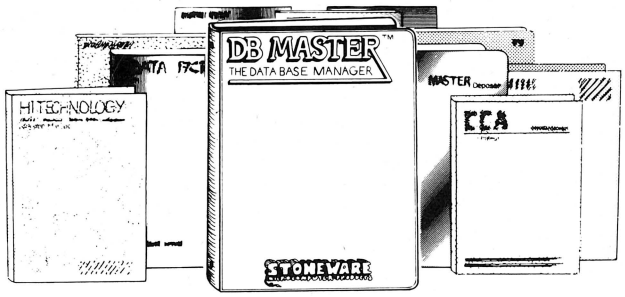
During normal usage of your disk drive particles of dirt, dust or oxide build up on the read/write head(s). This can cause lost data, I/O errors and even destroy a diskette. You can either take your drive apart and clean it by hand (which may void warrantee) or do it the easy way.

This kit includes a bottle of cleaning solution and a head cleaning diskette to keep your drive heads contaminant-free. Saturate the disk with solution, insert in drive and turn it on for 30 seconds.

Contact: Chuck Hastings, Data Recording Products Division/3M, 223-5N 3M Center, St. Paul, Minnesota 55144.

UNTIL TODAY THERE WERE MORE THAN 20 DATA BASE MANAGERS FOR THE APPLE II.
NOW THERE'S ONLY ONE!

DB MASTER



THE APPLE DATA BASE MANAGER YOU'VE BEEN WAITING FOR!

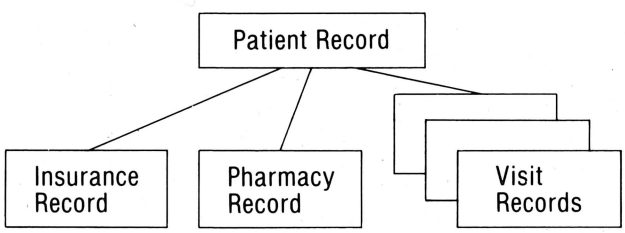
If you want an easy-to-use, flexible, and versatile data base manager, you have a choice of one. DB MASTER from Stoneware Microcomputer Products - soon to become the standard by which all others will be judged.

But don't just take our word for it. Compare the many advanced features of DB MASTER with the data base manager you're now using. Or for that matter, compare it with any data base manager on the market. No one will even come close.

FEATURES	DB MASTER	OTHER DBMS
FILING SYSTEM:		
Maximum search time to find any record by its primary key	UNDER 3 SECS	_____
True ISAM file system with multi-field primary keys	YES	_____
Multiple secondary keys for rapid access (5-7 seconds) to records by any field	YES	_____
Primary & Secondary keys maintained automatically—no need to rebuild keys after adding records	YES	_____
Maximum record size (bytes)	1020	_____
Maximum number of fields/record ..	100	_____
Handles files with more than one diskette of data	YES	_____
Custom disk operating system (DOS) for faster data retrieval and program chaining	YES	_____
User-designed screen formats	YES	_____
Up to 9 screen "pages" per record...	YES	_____
Ten field types, including dollar/cents, phone & social security number, date, etc.	YES	_____
Automatic data compaction for increased disk storage capacity.	YES	_____
Wild card, partial string, range and Boolean search capabilities...	YES	_____
Dynamic prompting (tm) - lists all available functions on screen—no need for quick reference card.	YES	_____
Password file protection	YES	_____
Four function calculator mode	YES	_____
Daily update lists for printout of all records added/edited on any day or range of dates	YES	_____
REPORT GENERATOR:		
Send reports to screen or printer	YES	_____
Sort on up to 6 fields at a time	YES	_____
Column subtotals and totals	YES	_____
Subtotal and page breaks	YES	_____
Up to 24 computed fields per report .	YES	_____
Up to 9 lines of column titles	YES	_____
Up to 9 lines for each record	YES	_____
Maximum number of fields per report	100	_____
Code fields - store short codes, print long descriptions	YES	_____
Comment lines and footnotes	YES	_____
Comment fields for printing labels or headers within each record ...	YES	_____
Summary only reports	YES	_____

Have a more complex application? DB MASTER can be used to emulate the hierarchical data base manager s used with larger computer systems!

A typical Hierarchical File Structure:



(Learn more about emulating a hierarchical data base in our 140 page user's manual.)

Coming soon:
DB MASTER UTILITY PAK #1: Add, drop or change fields in existing files without re-entering data! Interchange DB MASTER files with VisiCalc* and other programs!
DB MASTER FOR HARD DISK SYSTEMS
DB MASTER FOR THE APPLE III

STONEWARE™
MICROCOMPUTER PRODUCTS
50 Belvedere Street, San Rafael, CA 94901 (415) 454-6500

Apple, Apple II & Apple III are trademarks of Apple Computer, Inc. VisiCalc is a trademark of Personal Software, Inc.

APPLICATION FOR MEMBERSHIP

Name of Organization _____

Mailing Address

Street: _____

City: _____ State: _____ Zip: _____

Country: _____

(If the above is a post office box, please supply a street address below where parcels may be sent.)

_____

OFFICERS OR OFFICIALS

NAME PHONE (area code & number)

___ President: _____

___ Treasurer: _____

___ Editor: _____

___ Other: _____

(Please check above the name of the IAC contact person).

For Clubs:

Terms Expire: _____

Number of Members: _____

Make check or money order payable in U.S. dollars to "International Apple Core".

Remittance enclosed: $_____

(Return application and remittance to the International Apple Core, P.O. Box 976, Daly City, California 94017, USA.)

Check appropriate categories below:

_____ FULL MEMBERSHIP is available only to Apple User's groups. A $50.00 membership fee must accompany this form. Please indicate the time and place of your regular meetings: _____

_____

_____ ASSOCIATE MEMBERSHIP applicants must provide evidence that they are nonprofit institutions. There is no membership fee.

_____ SPONSORS: Please indicate the name, position, and telephone number of the person in your organization responsible for liaison with the IAC. The Sponsoring Membership fee is $200.00.

_____ Please add 15% to your fees if your organization is overseas and you would like all material sent International Air Mail.

P. O. BOX 976, DALY CITY, CALIFORNIA 94017 USA

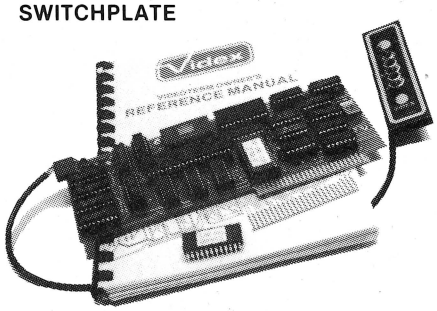
"APPLE" is the registered trademark of Apple Computer, Inc.
INTERNATIONAL APPLE CORE is licensed by Apple Computer, Inc. to use certain of the latter's trademarks.

The Text Solution for APPLE II®

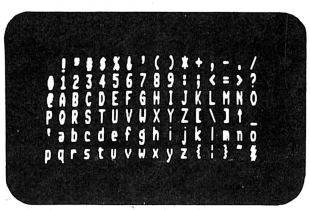
Now APPLE II® Owners Can Solve Text Problems With VIDEOTERM 80 Column by 24 Line Video Display Utilizing 7 X 9 Dot Character Matrix

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

VIDEOTERM, MANUAL, SWITCHPLATE



7X12 MATRIX
18X80 OPTIONAL



7X9 MATRIX
24X80 STANDARD

VIDEOTERM

BASICs VIDEOTERM lists BASIC programs, both Integer and Applesoft, using the entire 80 columns. Without splitting keywords. Full editing capabilities are offered using the ESCape key sequences for cursor movement. With provision for stop/start text scrolling utilizing the standard Control-S entry. And simultaneous on-screen display of text being printed.

Pascal Installation of VIDEOTERM in slot 3 provides Pascal immediate control of the display since Pascal recognizes the board as a standard video display terminal and treats it as such. No changes are needed to Pascal's MISC.INFO or GOTOXY files, although customization directions are provided. All cursor control characters are identical to standard Pascal defaults.

Other Boards The new Microsoft Softcard* is supported. So is the popular D. C. Hayes Micromodem II*, utilizing customized PROM firmware available from VIDEX. The powerful EasyWriter* Professional Word Processing System and other word processors are now compatible with VIDEOTERM. Or use the Mountain Hardware ROMWriter* (or other PROM programmer) to generate your own custom character sets. Naturally, VIDEOTERM conforms to all Apple OEM guidelines, assurance that you will have no conflicts with current or future Apple II* expansion boards.

Advanced Hardware Design VIDEOTERM's on-board asynchronous crystal clock ensures flicker-free character display. Only the size of the Pascal Language card, VIDEOTERM utilizes CMOS and low power consumption ICs, ensuring cool, reliable operation. All ICs are fully socketed for easy maintenance. Add to that 2K of on-board RAM, 50 or 60 Hz operation, and provision of power and input connectors for a light pen. Problems are designed out, not in.

Available Options The entire display may be altered to inverse video, displaying black characters on a white field. PROMs containing alternate character sets and graphic symbols are available from Videx. A switchplate option allows you to use the same video monitor for either the VIDEOTERM or the standard Apple II* display, instantly changing displays by flipping a single toggle switch. The switchplate assembly inserts into one of the rear cut-outs in the Apple II* case so that the toggle switch is readily accessible. And the Videx KEYBOARD ENHANCER can be installed, allowing upper and lower case character entry directly from your Apple II* keyboard.

Firmware 1K of on-board ROM firmware controls all operation of the VIDEOTERM. No machine language patches are needed for normal VIDEOTERM use.

Firmware Version 2.0

Characters	7 x 9 matrix	Display	24 x 80 (full descenders)
Options	7 x 12 matrix option; Alternate user definable character set option; Inverse video option.		18 x 80 (7 x 12 matrix with full descenders)

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

Apple II* is a trademark of Apple Computer Inc.
ROMWriter* is a trademark of Mountain Hardware Inc.
Micromodem II* is a trademark of D. C. Hayes Associates Inc.
Softcard* is a trademark of Microsoft
EasyWriter* is a trademark of Information Unlimited Software Inc.

PRICE:
- **VIDEOTERM includes manual $345**
- **SWITCHPLATE . . . . . $ 19**
- **MANUAL refund with purchase . . . . $ 19**
- **7 x 12 CHARACTER SET . . . . . $ 39**
- **MICROMODEM FIRMWARE . . . . . $ 25**

APPLE II® OWNERS!

introducing the

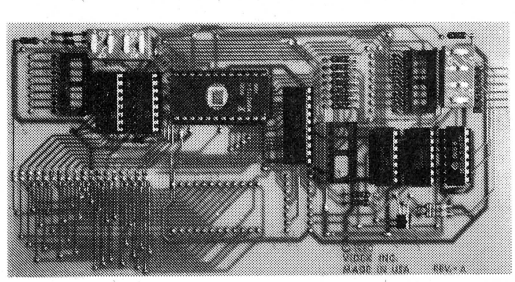
KEYBOARD & DISPLAY ENHANCER

- ***PUT THE SHIFT AND SHIFT LOCK BACK WHERE IT BELONGS***
- ***SEE REAL UPPER AND lower CASE ON THE SCREEN***
- ***ACCESS ALL YOUR KEYBOARD ASCII CHARACTERS***

Videx has the perfect companion for your word processor software: the **KEYBOARD AND DISPLAY ENHANCER**. Install the enhancer in your APPLE II and be typing in lower case just like a typewriter. If you want an upper case character, use the SHIFT key or the CTRL key for shift lock. Not only that, but you see upper and lower case on the screen as you type. Perfectly compatible with Apple Writer and other word processors like, for example, Super-Text.

If you want to program in BASIC, just put it back into the alpha lock mode; and you have the original keyboard back with a few improvements. Now you can enter those elusive 9 characters directly from the keyboard, or require the Control key to be pressed with the RESET to prevent accidental resets.

KEYBOARD AND DISPLAY ENHANCER is recommended for use with all revisions of the APPLE II. It includes 6 ICs, and EPROM and dip-switches mounted on a PC board, and a jumper cable. Easy installation, meaning no soldering or cutting traces. Alternate default modes are dip-switch selectable. You can even remap the keyboard, selecting an alternate character set, for custom applications.



PRICE • KDE-700 (REV. 7 or greater) . . . . . $129.
• KDE-000 (REV. 6 or less) . . . . . $129.

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

Videx INC

VIDEX
897 N.W. Grant Avenue
Corvallis, Oregon 97330
Phone (503) 758-0521

SORTING IN BASIC

by Art Mack
Cider Press

One of the things that every programmer has to do eventually is to sort some data. Described here is one way to write a sort routine in BASIC as follows.

Starting with the first element of an array, (the sort shown here is for an array of real numbers, but applies to Integer and String arrays as well) the array is searched to find the smallest element, (assuming we want to sort into ascending order) and that element is placed in position 1 of the array. Next, we start with element number 2 and find the smallest element left in the array and place it in position number 2. We continue this way with positions 3, 4, 5, etc., until we reach the last element in the array. Thus when we are finished, the smallest element in the array is in position 1, the second smallest in position 2, and so on.

The following example shows how to sort an array named "D" of dimension "N" (assume that all the elements of "D" have already been put into the array).

```
800 FOR I = 1 TO N-1: REM LOOP
    FOR N-1 TIMES, THE LAST
    ELEMENT WILL AUTOMATI-
    CALLY BE IN THE RIGHT
    PLACE.
801 FOR J = I + 1 TO N: REM LOOP
    AND CHECK THE I'TH ELE-
    MENT WITH ALL REMAINING
    ELEMENTS IN THE ARRAY.
802 IF D(I) < D(J) THEN 806: REM
    IF THE I'TH ELEMENT IS THE
    SMALLEST TRY THE NEXT RE-
    MAINING ELEMENT.
803 A = D(I): REM OTHERWISE
    SWITCH THE ELEMENTS
    (LINES 803-805).
804 D(I) = D(J)
805 D(J) = A
806 NEXT J: REM CONTINUE FOR
    THE NEXT J
807 NEXT I: REM CONTINUE FOR
    THE NEXT I
```

Note that the "inner" loop (lines 801-806) is executed from N-1 to 1 times depending on the value of I, and that the "outer" loop (lines 800-807) is executed N-1 times. Also note that the routine can easily be changed to sort the array into descending order (highest to lowest values) simply by changing line 802 to:

```
802 IF D(I) > D(J) THEN 806
```

IAC Sponsoring Members *(continued from page 3)*

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

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

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

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

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

dilithium Press
P.O. Box 606
Beaverton, OR 97075
(503) 646-2713

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

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

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

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

Mountain Computer, Inc.
300 Harvey West Blvd.
Santa Cruz, CA 95060

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

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

Peachtree Software
3 Corporate Square, Suite 700
Atlanta, GA 30329
(404) 325-8533

Peripherals Unlimited
Craig Vaughan
2105 Sheriff Ct.
Vienna, VA 22180

Programma International, Inc.
2908 N. Naomi St.
Burbank, CA 91504
(213) 954-0240

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

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

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

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

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

As an IAC sponsor, your name could be listed here in each issue of the Apple Orchard.

apples apples apples apples apples

WE HAVE EVERYTHING YOU NEED FOR YOUR APPLE COMPUTER

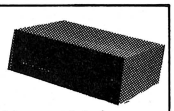
APPLE CLOCK

Real-time and date information. Interrupts permit. Foreground/Background operation of two programs simultaneously. Battery back-up. Crystal-controlled for .001% accuracy. Onboard ROM for easy access from BASICs. Supports PASCAL. Time from one millisecond to one year. **$279.00**

ROMWRITER

Program your own EPROMs. Create your own firmware. Programs 2K, 2716 5V EPROMs. Disk software package provides easy EPROM programming. EPROMs are verified after BURN. RUN your programs from on-board socket or install them on ROMPLUS. **$175.00**

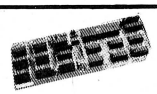
EXPANSION CHASSIS



By popular demand! Eight more slots for your Apple. Attractive sturdy enclosure. Its own heavy duty power supply. Easy to use. Address cards in Expansion Chassis the same way as in your Apply. Only one additional command to specify in Apple or in Expansion Chassis. **$649.00**

Z80 MICROSOFT Z80 SOFTCARD

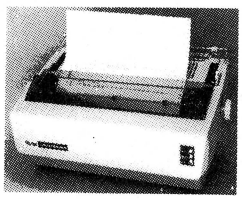
- Z80 CPU on Apple Card
- CP/M 2.2 by Digital Research
- Microsoft BASIC MBASIC 5.0
- GBASIC 5.0 includes Apple Graphics
- File transfer functions for reading 13 or 16 sector Apple diskettes
- Will use 80 x 24 cards & terminals
- Can use Language Card for 56K CP/M



COBOL & FORTRAN-80 NOW IN STOCK!

$349.00 List **$299**

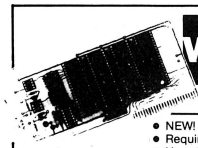
The Vista V300 Printer



- 25 CPS printspeed
- Static print impact
- 136 printable columns
- 1/120 inch min. char. spacing
- 1/48 inch min. line spacing
- 1000 msec. return time
- 40 msec., line feed time
- 381mm (15") max. paper width
- Multistrike fabric black ribbon
- 96 print characters
- Standard 96 character wheel
- Standard parallel or RS232-C compatible
- 115V±10%, 50/60 Hz, 70 W power requirements

$1895.00

Vista™ MUSIC MACHINE 9© WITH 9 VOICES!



ASSEMBLED AND TESTED **$129.95**

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

apples NEW CALIFORNIA MICROPRODUCTS KEYPAD

- Has ← → and "space" keys for VisiCalc™ compatibility!
- One version for new and old Apple keyboards.
- Easily installed. No soldering.
- Metal case, heavy duty.
- Does not req I/O slot.

List $199.00 **$174.95**

MICROSOFT 16K ON A PLUG-IN CARD. $175

Microsoft's new RAMCard simply plugs into your Apple II,* and adds 16K bytes of dependable, buffered read/write storage.

Until now, the only way to get this much memory was to have an Apple Language Card installed. VisiCalc™ and other Apple software packages can take advantage of RAMCard. Even with the RAMCard in place, you can still access your ROM BASIC and monitor routines.

AVAILABLE SOON! VISTA 80-TRACK APPLE DISK DRIVE

- Can boot & run standard Apple disks.
- Can boot & run 80-track diskettes for over 300 Kbytes of storage.
- Based on MPI B91 Drive.
- Uses Apple Disk Interface.
- Uses Apple DOS 3.2 or 3.3 (3.3 required for 80-track operation).

List $499.00 **$459.00**

SPECIAL Visi-Calc

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

$125.00*

*SUBJECT TO QUANTITY ON HAND

AVAILABLE SOON CRYPTEXT for Apple II data encryption for disk files. $449

apples

BIT 3® $349.95

80 BY 24 VIDEO BOARD
- Fully compatible w/Apple II
- Has video input from Apple II
- Can switch between Apple video and 80 by 24 video under software control
- Fully follows Pascal protocols, no system reconfiguration needed
- Comes with 7 x 9 character generator, full upper/lowercase
- Light pen input supported in firmware (light pen not included)

NEW M & R ENTERPRISES

"Apple Fan" cure your thermal problem in an over-crowded machine designed to fit inside a case and operate noiselessly.

List $54.95 **$49.95**

NEW apple /// NOW IN STOCK! $3995.00

The Most Powerful Professional Computer In Its Class!

FEATURES:

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

- Apple III Option A: Apple II 96K, Information Analyst Package, 12" B/W Monitor..... $3995.00
- Apple III Option B: Same as Option A Plus: Disk II for Apple III..... $4395.00
- Apple III Option C: Same as Option A Plus: Disk II for Apple III, Silentype Thermal Printer $4785.00
- To substitute 128K Apple III for 96K Apple III..... Add $450.00

DISK II FLOPPY DISK SUBSYSTEM

Disk II disk w/controller	$569.00
Disk II additional disk	$450.00

Includes powerful disk operating system, and adds high speed random access storage to your system.

APPLE II PLUS

Apple II + w/16K	$1150.00
Apple II + w/32K	1199.00
Apple II + w/48K	1249.00

Apple II Plus systems come with floating point Basic Card in ROM. Order with integer ROM Card or Language Card for multi-language operation.

16K Upgrade Kit..... $44.95

APPLE II

Apple II w/16K	$1150.00
Apple II w/32K	1199.00
Apple II w/48K	1249.00

Apple II systems come with integer BASIC in ROM. Order with Applesoft ROM Card or Language system for Multi-Language operation.

SEND FOR OUR 1981 CATALOG, ONLY $2.00, FREE WITH ANY ORDER

ADVANCED COMPUTER PRODUCTS

RETAIL STORES OPEN MON.-SAT.

STORE #1: 1310 "B" E. Edinger, Santa Ana, CA 92705 Showrooms, Retail, Warehouse
STORE #2: 674 El Camino Real, Tustin, CA 92660 Specializing in Systems
Coming Soon – STORE #3: 542 W. Trimble Road, San Jose, CA 95131 • (408) 946-7010

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

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

IAC MEMBER CLUB ADDITIONS

Abilene Apple Club
925 N. Judge Ely Blvd.
Abilene, TX 79601

Aerospace Apple User Group
28901 Lotusgarden Dr.
Canyon County, CA 91351

Apple Pi of Brevard
P.O. Box 327
Melbourne, FL 32901

Apple Polishers
1112 Glacial Dr.
Minot, ND 58701

Apple Tech
412 West Third
Mishawaka, IN 46544

Apple Users-Nu
Northwestern University
Evanston, IL 60201

Apple Users Group Sweden
Norra Vallvagen 24
Kristianstad 291 32 Sweden

Apple Canada
P.O. Box 696, Station B
Toronto, Ont. Canada M2K 2P9

Apple Boston
3 Center Plaza
Boston, MA 02108

Atlanta Society of Professional Apple Users Group
6600 Power Ferry Rd.
Suite 220
Atlanta, GA 30339

Blossum Valley Apple Club
5821 Cottle Rd.
San Jose, CA 95123

British Apple Systems User Group
P.O. Box 174
Watford, England WD2 6NF

Catalunya Apple Club
Fabra y Puig 389 E/4
Barcelona-31 Spain

Caug
2805 Chestnut Ct.
Columbus, IN 47201

CIA (Central Illinois Apple)
1023 W. Hudson
Peoria, IL 61604

Emu
Box 3143
GPO
Sydney 2001 New Australia

Fetch
Naval Air Facility Box 13
FPO Seattle, WA 98767

Fox Valley Crab Apples
Univ. of Wisconsin–Oshkosh
Oshkosh, WI 54901

G. F. Apple S.A.U.C.E.
25000 B Southe Columbia Rd.
Grand Forks, ND 58201

Gerold V. Van Der Vlugt MD
Department of State
Washington, D.C. 20520

Gila Valley Apple Growers Assn.
Box 1077
Thatcher, AZ 85552

Gauge
16-B S
Oceanview
Finegayah, GUAM 96912

H.A.U.S.
P.O. Box 30262
Honolulu, HI 96820

H.O.T. Apple-Pie
2321 Lee St.
Waco, TX 76711

HISD/Basic Curriculum Dept.
3830 Richmond Ave.
Houston, TX 77027

Jakarta International School
P.O. Box 791KBT
Jakarta, Indonesia

Little Rock Apple Addicts
P.O. Box 55215
Hillcrest Sta.
Little Rock, AR 72205

Monmouth Apple Corps
P.O. Box 333
West Long Beach, NJ 07764

Mountain View Apple Users Group
1923 Viola Dr.
Sierra Vista, AZ 85635

NovApple
8108 Adair Ln.
Springfield, VA 22151

NW Suburban Apple Users
1300 S. Elmhurst Rd.
Mt. Prospect, IL 60056

PenCom
J 303 Waverly Dr.
Frederick, MD 21701

Peninsula Apple Core
1419 Todds Lane
Hampton, VA 23666

Placer County
1228 High St.
Auburn, CA 95603

Plane Apple Club
Box 12013
Wichita, KS 67277

Ridgecrest Apple Group
Star Route Box 109E
Inyokern, CA 93527

S Australian Apple Users Club
c/o Computerland
125 Pirie St.
Adelaide-South Australia 5000

SEA
3258 Powers Ferry Rd.
Marietta, GA 30067

Short Hills Apple Pits
29 Clive Hills Rd.
Short Hills, NJ 07078

Slacc
2445 Cleveland
Granite City, IL 62040

SMAUG
10201 Fontainebleau Blvd.
#206
Miami, FL 33172

South Bay Apples Computer Club
P.O. Box 5201
Torrance, CA 90510

South Colorado Apple Users
1635 S. Prairie
Pueblo, CO 81005

Space Coast Apple User Group
P.O. Box 4332
Patrick AFB, FL 32925

Tidewater Apple Organization
1021 Tivoli Crescent, Apt. 102
Virginai Beach, VA 23456

WAUG
P.O. Box 19
Wondai, Old Australia 4606

Washington Apple Pi
P.O. Box 34511
Washington, D.C. 20034

Wisconsin Apple Users
P.O. Box 11463
Milwaukee, WI 53211

CONTRIBUTING PUBLICATIONS AND CLUBS

FOR THIS ISSUE OF *APPLE ORCHARD*

Apple-Dillo–River City Apple Corps
2015 Ford St., Austin, TX 78704

Apple Slice
P.O. Box 536, Bountiful, UT 84010

Call -A.P.P.L.E.–Apple Pugetsound Program Library Exchange
304 Main Ave. S., Ste. 300, Renton, WA 98055

Cider Press–S. F. Apple Core
1515 Sloat Blvd., Ste. 2–San Francisco, CA 94132

Contact–Apple Computer, Inc.
10260 Bandley Drive, Cupertino, CA 95014

Dallas Apple Corps
P.O. Box 5537, Richardson, TX 75080

Green Apple Bits–No. Central Iowa Apple Users Group
4417-129 No. Zircon Lane, Cedar Falls, IA 50613

Harvest–Northwestern Illinois Apple Users Group
1300 S. Elmhurst Rd., Mt. Prospect, IL 60056

Mini' App'les
13516 Grand Ave. S., Burnsville, MN 55337

Original Apple Corps
12804 Magnolia, Chino, CA 91710

Tulsa Computer Society–Apple Users
P.O. Box 1133, Tulsa, OK 74101

Washington Apple Pi
P.O. Box 34511, Washington, D.C. 20034

LETTER PERFECT T.M. LJK

WORD PROCESSOR Apple II, and Apple II Plus*, DOS 3.3., 40/80 Columns

MAIN - MENU
CURRENT DRIVE
NUMBER #1
SLOT #6

→ Editor ←
Change Drive #, Slot
Load
Save
Merge
Screen Format
Printer
Lock
Unlock
Delete
Format Disk
Reconfigure
Data Base Merge
Quit

Press ← or → to move cursor
Press (Return) for selection

40 Column
80 Super-R-Term
Smarterm
Videx
Bit 3

USE: HAYES MICROMODEM II T.M. **TO SEND FILES**

EASY TO USE: LETTER PERFECT is a character orientated word processor with the user in mind. Fast action (machine language), menu driven, single load program. 34K Free. Requires Paymar LCA, one disk drive, and 32K memory. One time configuration for your system, printer type etc., can be reconfigured at any time. Right hand justification. Supports incremental spacing, underlining, boldface with NEC or Qume/Diablo. Will use any of the special print characters of your specific printer. Key codes make mnemonic sense for easy use. All text packed during saving for greater disk storage capacity. Full typewritten page of buffer space for easy manuscript editing. Menu driven printer selection, or use your own special print driver. Works with DOS 3.3. "Screen format allows you to preview printed text".

All this and more, for $149.95.

Features:

FULL CURSOR CONTROL
Home Cursor
Scroll Page Forward
Scroll Page Backward
Pause Scroll
Scroll Line at Time
Scrolling Speed Control
Move Cursor Down
Beginning of Text

MULTIFUNCTION FORMAT LINE
Standard Formats a Default
Formats Easily Changed
Right Justification
Left Margin
Page Width
Line Spacing
Lines Per Page
Font Changing
Set Page#
Top Margin
Bottom Margin

Delete a Character
Insert a Character
Delete a Line
Insert a Line
Headers and Footers
Shift Lock and Release
Global and Local Search and Replacement
Underlining and Boldface
Automatic Centering
Horizontal Tabs
Special Print Characters
Split Catalog
Page Numbering up to 65535
Prints up to 256 Copies of Single Text File
Non Printing Text Commenting

FUNCTIONS
Delete All Text
Delete All After Cursor
Delete All Before Cursor
Delete Next Block
Delete Buffer
Move Next Block to Buffer
Add Next Block to Buffer
Insert Block From Buffer
Merge Text Files

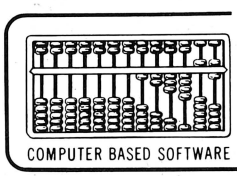
SEND FOR FREE CONTROL PAGE

* Data Base Merge
* Screen Format
* Font types changed in body of text
* With RH justification
* Multiple Print Drivers

NEC
Qume/Diablo
CENT 737
Epson
Paper Tiger
Write Your Own

Printer — Use any type: Epson NEC, Qume, CENT 737. All parallel, serial printer types.

DEALER INQUIRES INVITED



COMPUTER BASED SOFTWARE LJK T.M. ENTERPRISES INC.

LJK ENTERPRISES INC.
P.O. Box 10827
St. Louis, MO 63129
(314) 846-6124

DEALER INQUIRES INVITED

*Apple T.M. of Apple Computer Inc., — *T.M. M&R Enterprises

I.A.C. MEMBERSHIP INFORMATION

FULL MEMBERS

The International Apple Core is a non-profit organization composed of Apple Computer User Groups throughout the world. The IAC was formed to disseminate all types of information from Apple clubs and users, computer related industry, and Apple Computer, Inc. We distribute public domain software, application notes, general Apple information, and product information. Unaffiliated with Apple Computer, Inc. or any other manufacturer, we provide an objective input/output device for questions and problems on any subject dealing with Apple computers.

Apple user clubs have need for timely access to new developments in the Apple Computer world. This is the principal reason for the IAC's organization today. Our publication, the "Apple Orchard", provides the latest and best information available on a quarterly basis. Our member clubs receive technical information in the form of Apnotes. These cover Apple Computers, related equipment, and related programming from other manufacturers. We provide a newsletter exchange coordinator to facilitate newsletter swaps between clubs. The IAC also collects and distributes public domain software. Depending on the software's availability, new diskettes are sent out as frequently as once a month.

The International Apple Core provides many services beyond information dissemination. We support the special interest user groups which our member clubs cannot: education, handicapped, medical, ham radio, and legal SIG's are examples. The "Orchard" publishes a complete list of our member clubs so that interested users may easily get in touch. Our Standards Committee works with manufacturers to define hardward and software standards for Apple Computers. There is even a committee just to help new Apple clubs get started.

As a Full Member, your club will be able to participate in the election of IAC Directors in your region. There are currently eleven Directors world-wide. Directors provide one important link between member clubs and the IAC, and as your representatives they set the IAC's policies and guide its administration. When schedules permit, the Directors and Officers are available to meet with clubs for personal input and exchange of ideas.

Full Membership is open to all Apple Computer User Groups. Individuals may participate in the IAC through our member clubs or as Sponsors. The combined initiation fee and annual dues for Full Members in 1981 will be $100.00 (US). To enroll, simply return a completed membership application with your first year's dues.

SPONSORS

Manufacturers having business relating to Apple computers need timely access to information that the IAC distributes. In addition many will seek access to the IAC membership for business interests, either to promote a product or to conduct market research.

To satisfy the needs of software and hardware manufacturers, the International Apple Core has created the Sponsoring Membership. This type of membership is tailored to meet commercial interests.

As a Sponsor you receive:

- All printed information sent to IAC member clubs at no cost. Other materials are provided at a nominal fee.
- A subscription to "The Apple Orchard," our publication.
- Preferential placement of your advertising in the "Orchard".
- Free listing in each issue of the "Orchard".
- Up to date mailing lists of our membership upon request.
- Application notes from Apple Computer and other users. The International Apple Core will distribute appropriate application notes on your products as well.
- Voting membership on the Standards Committee. The Standards Committee will be setting hardware and software standards for Apple computers.
- A forum for new product feedback.
- The non-voting right to participate in other IAC activities.

The Sponsoring Membership is open to all corporations and individuals that wish it. The annual membership fee for 1981 has been set at $200. To enroll, simply return a completed membership application with your first year's dues.

(continued on page 73)

I.A.C. MISCELLANY

by Joe Budge, Secretary

Several clubs have written in confusion as they had missed IAC mailings for six months. Upon checking, I find the materials were sent, but not received by the club. Invariably the club will have had internal problems. The IAC contact person, who receives our mailings has either moved or otherwise become indisposed to convey the mailings to his club. When this is discovered, there's a huge uproar and scrambling to find or replace the missing mailings. To keep this to a minimum, the "Orchard" will begin listing materials that each club should have received. If your club missed some of these, start checking right away. Don't wait six months. If you act promplty, we will be happy to do our best to replace what you might have lost (within reason, of couse). Here is the first list, covering all materials sent through 2/10/81:

APNOTES: A complete set of Apnotes. The last Apnote printing was December 6, 1980. Your club should have copies of all Apnotes listed in the index for that date. New clubs are given a complete set when they join.

MINUTES: Minutes of all IAC meetings. These cover the First Annual Meeting of the Membership on March 13, 1980 and Board of Directors Meetings for October 27, 1979, March 13, 1980, and May 19, 1980. Again, new clubs should have a complete set.

SOFTWARE: Five official and IAC disks have been sent to clubs. For logistical reasons, we are unable to send new members back copies. Thus a club should have IAC software dating back only as far as that club's membership date on our books. Software was published in May, July, August, and December of 1980, and February, 1981.

BULLETIN: We began publishing a monthly news bulletin on December 1. Clubs should have received one per month subsequently. Important information in the Bulletins is reprinted in the "Orchard," so we do not send out back issues.

ORCHARD: Starting with this issue, all IAC clubs should receive one copy of each "Orchard."

OTHER: Clubs in eligible regions should have received Director Nomination forms in December. Those regions eligible to nominate a director this year were Asia, S. America, and all U.S. regions. Only the Northern and Eastern U.S. regions nominated more than one director, warranting an election. Ballots were mailed to those regions February 9. All clubs were sent a notice of the Annual Meeting on February 1.

The Associate membership was created to help educational, research, and charitable institutions that have an interest in Apple Computers but cannot join the IAC as full members for financial reasons. Associate members receive all the printed matter supplied free to the IAC member clubs. This material consists primarily of Apnotes and resource lists. This material does not include free software which the IAC sends to its Full members, nor does it include the "Orchard" which is available by subscription. If software and other additional IAC services are desired, the institution is encouraged to organize a user's group which may apply for Full membership. Associate members participate in IAC committees and activities.

The Associate Membership is open to all non-profit institutions at no cost. Membership applications must be accompanied by evidence that the institution is non-profit. If the membership will be care of an individual, evidence must be provided that the individual represents the entire institution to the IAC. Please submit whatever you feel is appropriate to demonstrate these requirements. Due to the diversity of institutions and countries, the IAC cannot set any fast rules. Associate Membership applications will be judged on their merits by the IAC President.

IAC Membership Information *(continued from page 72)*

ASSOCIATE MEMBERS

Educators and researchers who use Apple computers have need for timely access to new developments in the Apple Computer world. Our publication, the *Apple Orchard,* provides the latest and best information available on a quarterly basis. In addition, the IAC publishes technical information in the form of Apnotes. These cover Apple Computers, related equipment, and related programming from other manufacturers.

The International Apple Core provides many services beyond information dissemination. We support the special interest user groups which our clubs cannot: education, handicapped, medical, ham radio, and legal SIG's are examples. Our Standards Committee works with manufacturers to define hardware and software standards.

WHO WAS ASKING ABOUT ASCII?

by Val J. Golding

But First, a Bit About Bits

Actually, a whole bunch of people. Understanding the Apple's ASCII video display and its relationship to the ASCII character *set* is of paramount importance to efficient programming and manipulating screen displays or modifying DOS file names, etc.

To make the distinction between the ASCII character *set* and the screen display, keep in mind that the character set is the characters that you originally enter from the keyboard, i.e., INPUT, and the video display is the form in which you eventually *see* the characters, i.e., OUTPUT.

The standard ASCII character set is composed of 128 characters, some of which perform control functions and are not displayed. The display set basically is also 128 characters. However, there are three different display *modes,* and herein lies the area of confusion to many users. We will come back to the display set shortly, but first let's define the types of characters that make up the 128 character set, and also explain about the terms "positive" and "negative" ASCII.

Throughout this article we will use standard hexadecimal notation, i.e., the dollar sign [$] to express hexadecimal values, while decimal values will usually be shown in square brackets [ ] following the hex data.

In the Apple, as in most other mini and microcomputer systems, it is quite common to deal with figures in multiples of 8 or 16, because the hexadecimal number system [Base 16] is an easy form of notating binary numbers [Base 2], which is the number system used by the 6502 microprocessor. This explains in part why we so often see figures like 256, 1024, 8192, 32768 and 65536, as these are all exact multiples of 16.

A hex byte is composed of eight binary bits, and the maximum value that may be expressed in one byte is $FF [255]. (Note this is a total of 256 units, since the number of the first unit is zero.) If the value of a byte is greater than $7F [127], then it is said to be "negative", which is a way of expressing that the high order [most significant] bit is on. When the high order bit is present in an ASCII character, then we say that that character is negative ASCII. But we are getting ahead of ourselves.

Apple handles ASCII a little differently than most computers. Internally, the monitor converts most standard ASCII characters to negative ASCII, which it is set up to handle. Standard, or positive, ASCII, which is used by Applesoft and most other computers, uses only seven bits of each eight bit byte. Why? Because there are 128 characters, with values from $00 [0] to $7F [127]. In many systems the eighth bit is used for other purposes (parity checks, etc.) beyond the scope of our discussion here.

Both Applesoft and DOS use the high bit in a unique fashion in their respective command and error message tables, although the technique is not exclusive with the Apple. By setting the eighth bit "on" in the last character of a command name, we have provided a means for the appropriate operating system to recognize the end of a command name or error message. This makes no difference to the monitor, since the monitor will automatically convert all standard ASCII to negative ASCII before going to the output routine.

Standard ASCII

The standard ASCII character set is composed of four sub-groups of 32 characters each, as shown in the following table:

Sub-Group	Hex Values	Decimal Values
I Control	00-1F	0-31
II Numeric and special	20-3F	32-63
III Upper Case Alpha	40-5F	64-95
IV Lower Case Alpha*	60-7F	96-127

*The Apple video display will convert these to group II, unless you use one of the popular lower case adapters.

By using the CHR$(n) function in Applesoft, you can print any of the 96 characters of Apple's character set. (Values from 0 to 31 will not print, since these are the control characters.) This is one way in which you can access those few characters that are not available directly from Apple's keyboard, such as the backslash [\ ]. In addition, although it serves no worthwhile purpose, you can add 128 to any ASCII value, and using the CHR$ function, print the same character out. The following one-liner will demonstrate this by printing out two identical character sets:

```
FOR I = 0 TO 255: PRINT CHR$(I);
    : NEXT
```

Video Display

The ASCII screen character set is a horse of a different pigment. In designing the video display, Apple wanted to provide a means to display characters in either black on white or white on black, and in addition, to be able to switch rapidly from one mode to the other. These modes, in Applesoft, are named INVERSE, NORMAL and FLASH, respectively. In Integer Basic, they may be implemented with CALL -384, CALL -380, and POKE 50, 127, respectively.

Here is where the high bit bit comes in. We have started with the 128 characters of the regular ASCII set, none of which use the high bit, so by turning the high bit on, we have instantly created an additional 128 characters, or at least characters that the monitor can recognize as being non-standard. But this gives us only one alternate character set, and we want to be able to display characters in three different modes. How can this be resolved?

First of all, the Apple monitor doesn't like (and in fact, can't display) lower case characters, so there will be no need to provide INVERSE or FLASH for lower case, which reduces our requirements for one extra character set to 96. But 2 times 96 is 192, and we have only 128 alternate characters to work with. What else can we do?

You're right! The control characters. Since they don't print to the screen to begin with, we certainly don't need the ability to display them in INVERSE or FLASH. So let's dump them from our alternatives, and we are down to two sets of 64 characters each, which is just right.

Since the monitor converts standard ASCII to negative ASCII for normal display purposes, the normal *display* set uses the values $80-$FF [128-255], so our alternate sets will be using values from $00-$7F [0-127]. You will note that in a sense, "holes" have been left in the two alternate character sets by the removal of control and lower case characters, thus the sequence of characters used is non-standard. Table 7 on Page 15 of the Apple][reference manual shows the complete screen display set; the following table outlines the four non-standard display groups:

Group	Hex Values	Decimal Values
I Inverse alpha	00-1F	0-31
II Inverse numeric	20-3F	32-63
III Flashing alpha	40-5F	64-95
IV Flashing numeric	60-7F	96-127

This short Applesoft program will print, a character at a time, the entire display set to a central location on your screen, by actually changing the contents of memory location 1252. (The video display uses memory locations 1024 to 2047.)

```
100 FOR I = 0 TO 255 :
    POKE 1252, I :
    FOR J = 1 TO 250 :
    NEXT J,I
```

By knowing the screen coordinates of each memory location, characters can be POKEd to any part of the screen.

Lastly, the following program will display the entire ASCII screen character set to your video display.

```
10  REM

ASCII SCREEN DISPLAY

BY VAL J GOLDING

100 H$ = "0123456789ABCDEF"
110  HOME : HTAB 4: FOR I = 1 TO
     16: PRINT " "; MID$ (H$,I,1)
     ;: NEXT : POKE 34,2
120  HOME : FOR I = 1 TO 16: PRINT
     " "; MID$ (H$,I,1): NEXT : POKE
     33,32: POKE 32,4: HOME
130  INVERSE : FOR I = 0 TO 255
140  IF I < 17 THEN J = I + 64
150  IF I > 63 THEN  FLASH :J = I

160  IF I > 127 THEN  NORMAL :J =
     I - 64
170  IF I > 159 THEN J = I
180  PRINT  CHR$ (J);" ";:J = J +
     1: NEXT : PRINT : PRINT "
     HIT ANY KEY": CALL  - 756: TEXT

190  REM

NOTE CONTROL CHARACTERS $80-9F
WILL BE REPRESENTED BY THE
EQUIVALENT NORMAL CHARACTERS
```


SF APPLE CORE ANNOUNCES

"THE BEST OF THE CIDER PRESS 1980"

An anthology of articles & program listings from the SF Apple Core's newsletter **"The Cider Press".** Sections range from articles for the beginner to a larger section on PASCAL. Send your check or money order for $10.00 (US), add $2.00 outside of the United States to:

SF APPLE CORE
1515 Sloat Blvd., Suite 2
San Francisco, CA 94132

Name _____

Address _____

City _____ State _____

Zip _____ Country _____

The "Best of the Cider Press — 1978-1979" is still available for $5.00

Dealer Inquiries Invited

SYNERGISTIC SOFTWARE

presents

GREAT APVENTURES

Great adventure games utilizing the Apple's graphic capabilities.

DUNGEON CAMPAIGN

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

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

WILDERNESS CAMPAIGN

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

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

Get both Apventures on 1 disk for $32.50.

ODYSSEY: THE COMPLEAT APVENTURE

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

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

Requires 48K and disk. Integer only, $30.00

DOOM CAVERN

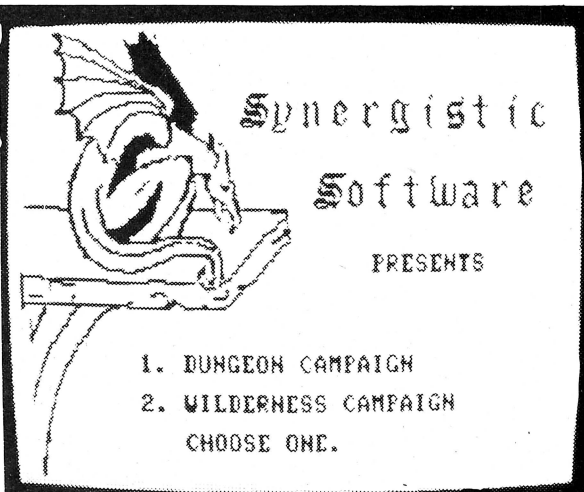
Doom Cavern is a high-resolution graphics version of the classic "Dungeon and Dragons" type games. Set up the persona (strength, intelligence, charisma, etc.) of your players with dice rolls, then venture forth into the dungeons of Hammardoom Castle. With perserverance, some luck, and reasoning, you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

SORCERER'S CHALLENGE

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

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

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



Available at your local dealer or send check or inquiry to
SYNERGISTIC SOFTWARE, 5221—120th Ave. S.E. Bellevue, WA 98006, (206) 226-3216
WA residents add 5.3% sales tax.

PASSING ARGUMENT VALUES TO MACHINE LANGUAGE SUBROUTINES IN APPLESOFT

by C. K. Mesztenyi
Washington Apple Pi

Applesoft provides input argument values to a machine language subroutine by POKEing them into fixed memory locations prior to the CALL statement, and provides retrieval of output values of the subroutine by PEEKing into fixed locations after the CALL statement. I not only find these procedures awkward, but if these values are in the internal registers (A, X, Y) of the 6502 processor, then I am forced to write an interface program in machine language to pick up or store these values. This become especially painful when I only want to test out one of the many "useful" subroutines published which are in the Monitor, Applesoft, interface cards, etc. Hopefully, one of these days some of the entry points of DOS will also be published.

As a partial remedy to this problem, I wrote the following somewhat general interface program which sets up the internal registers prior to jumping into any given machine language program. The interface program uses Applesoft subroutines GETBYTC, FRMNUM, GETADR, CHKCOM and memory address LINUM as published by J. Crossley in the March/April 1980 Apple Orchard. The use of this interface program from Applesoft is by the statement:

CALL origin, A-expr,X-expr,Y-expr,location

where

origin = decimal address of the interface program entry

```
                        ********************************
                        *                              *
                        * APPLESOFT ARGUMENT INTERFACE *
                        *                              *
                        *     BY C. K. MESZTENYI       *
                        *     WASHINGTON APPLE PI      *
                        *                              *
                        ********************************
                        *
                        *
LINUM       EQU   $50
FRMNUM      EQU   $DD67
CHKCOM      EQU   $DEBE
GETBYTC     EQU   $E6F5
GETADR      EQU   $E752
            *
            *
            ORG   $6000
            OBJ   $6000
            *
            *
6000: 20 F5 E6  JSR   GETBYTC
6003: 8A        TXA
6004: 48        PHA
6005: 20 F5 E6  JSR   GETBYTC
6008: 8A        TXA
6009: 48        PHA
600A: 20 F5 E6  JSR   GETBYTC
600D: 8A        TXA
600E: 48        PHA
600F: 20 BE DE  JSR   CHKCOM
6012: 20 67 DD  JSR   FRMNUM
6015: 20 52 E7  JSR   GETADR
6018: 68        PLA
6019: A8        TAY
601A: 68        PLA
601B: AA        TAX
601C: 68        PLA
601D: 6C 50 00  JMP   (LINUM)
            *
            SYM

--- END ASSEMBLY ---
TOTAL ERRORS: 0
32 BYTES GENERATED THIS ASSEMBLY
```

A-, X-, Y-expr = valid expressions which evaluate to single byte integers (0 < expr < 255) to be loaded into the registers A, X and Y, respectively.
location = decimal address of the machine language subroutine to be used with the above input values.

The interface program in assembly language is shown below.

Example:

Assume that the above interface program is loaded into memory location $origin = $6000 [origin = 24576] for this example. (Note that you may load it anywhere.) To use the Monitor subroutine PRNTAX [$F941] to output two bytes in hexadecimal, we have to load the registers A and X with these numbers (see Reference Manual p. 61). $F941 is equivalent to 63809 in decimal. Let the two bytes for output be the values of V1 and V2 of Applesoft variables. Now the following Applesoft statements will output those values:

V1 = . . .
V2 = . . .
CALL 24576,V1,V2,0,63809

The above statements can either be part of a program or also can be executed in immediate mode. Furthermore, V1 and/or V2 in the CALL statement can be replaced with the appropriate expressions shown with dots above it. Since the subroutine PRNTAX does not require input in register Y, the corresponding entry in the CALL statement contains a zero which is arbitrary.

Carrying the interface routine to a further conclusion, it is possible to enable it using that most underrated of Applesoft tokens, the Ampersand [&]. The only thing further that need be done is to enter a BASIC statement with three POKEs to set up the Ampersand vector, in the form:

POKE 1013,76 : POKE 1014,0 : POKE 1015,ad

. . . where ad is the decimal starting address of the assembly language routine divided by 256, provided that ad is evenly divisable.

If we establish the following Applesoft variable names:

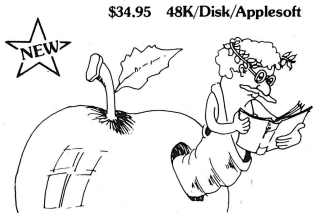
A (for Accumulator)
X (for X-Register)
Y (for Y-Register) and
PC (for Program Counter) or other name appropriate to the name of the routine being called, we can pass the arguments in the form:

&,A,X,Y,PC

The Applesoft program listing provided uses the S. H. Lam routine to put the machine code into memory, thus a separate binary load is not required. It contains an example use of the interface as indicated previously, printing the contents of the A and X registers as two hex bytes, a simple dec-hex converter.

STONEWARE
MICROCOMPUTER PRODUCTS
50 Belvedere Street, San Rafael, CA 94901 (415) 454-6500

Aristotle's Apple
$34.95 48K/Disk/Applesoft

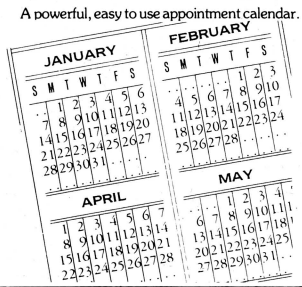


A computerized tutor for ANY subject, at ANY level. *by Scot Kamins*

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

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

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



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


```
]LIST

10  REM

AMPERSAND REGISTER LOADER

BY C K MESZTENYI & VAL J GOLDING

Passes arguments to A, X, and Y
Registers and Program Counter,
using the Ampersand

50  GOTO 500
100   INPUT A,X
110   & ,A,X,Y,AX
120   GOTO 100
500   POKE 1013,76: POKE 1014,0: POKE
      1015,96: REM
Set up Ampersand Vector

510 AX = 63809: REM
Set up variable as CALL
address for PRNTAX

520 Y = 0
700 A$ = "6000:20 F5 E6 8A 48 20
      F5 E6 8A 48 20 F5 E6 8A 48 2
      0 BE DE 20 67 DD 20 52 E7 68
       A8 68 AA 68 6C 50 00 20 ND8
      23G"
710   FOR I = 1 TO  LEN (A$): POKE
      511 + I, ASC ( MID$ (A$,I,1)
      ) + 128: NEXT : POKE 72,0: CALL
        - 144
720   GOTO 100: REM

Above routine by S H Lam stuffs
all the machine code in memory.

To relocate it, change the POKE
1015 in Line 500 and the address
in Line 700
```

AN ENTIRE STAR FLEET FOR $39.95. YOU'LL LOVE IT!

Like some future starship admiral hurtling through the vast void of hyperspace at speeds beyond comprehension, you are challenged to a battle for cosmic supremacy.

THE WARP FACTOR, the latest computer strategy game from SSI, is what every space war fan has been waiting for – the ability to command a star fleet in realistic battle simulation against alien vessels.

It is light years ahead of all other "space" games because it doesn't just fill your screen with pretty pictures and little substance. THE WARP FACTOR is a high-powered tactical simulation that places you squarely in the Captain's role, dealing with the critical parameters of interstellar battle such as sensor and scanner readings; energy allocation for weapons (phasers, disruptor bolts, photon and plasma torpedoes), shields, and warp engines; and battle damage.

THE STARSHIPS. With twelve different starship designs – ranging from dreadnoughts and fighters to star bases and base stations – representing five Galactic Empires, you can set up an astronomical variety of confrontations against another player or the computer.

Each class of vessels is awarded a point value to reflect its relative strength so you can assemble fleets of comparable power for a balanced game. Of course, your're free to play the intrepid hero against seemingly hopeless odds –perhaps mere fighters against a star base!

Employing up to ten ships, both sides can give individual or fleet orders, the latter allowing all your ships to execute your commands in unison.

THE COMPUTER, aside from being the game's perfect administrator and referee, also serves as your ever-ready, ever-capable nemesis in the multiple solitaire scenarios provided: The Reman Chase (replete with the Cloaking Device, Plasma Torpedoes, and Neutral Zone); Attack on Star Base; Attack on Base Station; and Dogfight.

THE TWO-PLAYER VERSION is essentially free-form. With each player choosing starships from a different Empire, you can create scenarios ranging from space skirmishes to a full-scale, all-out star war!

FOR $39.95, THE WARP FACTOR is undeniably the most complete and detailed simulation of tactical starship combat yet designed. It comes with the 5¼" program disc; a Starship Operating Manual; 3 Starship Data Cards; and a Game Selection Card – all of which will convert your computer into the gateway to galactic adventure.

THE WARP FACTOR™. The Universe Awaits Your Command.

Credit card holders, if you own an Apple®II 48K (Applesoft ROM) and a mini-floppy disc drive, call **800-227-1617 ext. 335 (toll free)** and charge your order to your VISA or MASTERCARD. In California, call 800-772-3545, ext. 335.

To order by mail, send your check to: Strategic Simulations Inc, Dept. AO, 465 Fairchild Drive, Suite 108, Mountain View, CA 94043.

All our games carry a 14-day money-back guarantee.

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

FOR YOUR APPLE®:

- ☐ **Computer Bismarck:** $59.95.
- ☐ **Computer Ambush** (a tactical simulation of man-to-man combat in WWII): $59.95.
- ☐ **Computer Napoleonics**, the Battle of Waterloo: $59.95.
- ☐ **Computer Quarterback** (a real-time strategy football game): $39.95.
- ☐ **Computer Conflict** (two modern-day tactical warfare simulations featuring *REBEL FORCE* and *RED ATTACK!*): $39.95
- ☐ **Computer Air Combat** (a simulation of air combat in WWII): $59.95.

FOR YOUR TRS-80®:

- ☐ **Computer Bismarck**, 48K Disc: $59.95. 32K Cassette: $49.95.

Apple is a registered trademark of Apple Computer Inc.

TRS-80 is a registered trademark of Tandy Corporation.

LOW RESOLUTION GRAPHICS IN PASCAL

by Bill Shepard

I have begun a concerted effort to convert all of my BASIC programs to Pascal. While the PEEKs and POKEs and unstructured GOTO's sometimes make the task difficult, I believe the result is well worth the effort. The resulting software is more reliable, runs faster, and is considerably easier to enhance. But the Pascal System has one missing element; it lacks low resolution graphics.

Certain applications are better suited for low resolution graphics than to high. While the border low resolution lines can be simulated with multiple high resolution lines, one must sacrifice both speed and the variety of color. The solution is to integrate low resolution capabilities into Pascal. This has been accomplished and has taken the form of a Pascal unit. To complete the unit, it was necessary to include the important text functions in addition to those of graphics.

To make code conversion as easy as possible, the BASIC functions were simply reproduced in Pascal. For example, to draw a horizontal line, BASIC uses the HLIN statement. In the Pascal unit, HLIN has three parameters which are identical to those of BASIC. The following table shows the similarities.

BASIC	PASCAL
PLOT x, y	Plot (x, y)
HLIN x1, x2 AT y	HLIN (x1, x2, y)
VLIN y1, y2 AT x	VLIN (y1, y2, x)
SCRN (x, y)	SCRN (x, y)
COLOR = 0 to 15	COLOR : = 0 to 15
TEXT	All_Text
GR	Mixed
GR: POKE -16302, 0	All_Graphics
POKE 34, lines	Top_Margin (lines)
HOME	Home
FLASH	Flash
INVERSE	Inverse
NORMAL	Normal
PRINT;	Print
PRINT	Print_Ln
HTAB x	HTAB (x)
VTAB y	VTAB (y)

Because of the nearly identical presentation of the Pascal procedures to those of BASIC, a detailed functional description of each will not be given. A clear presentation is contained in the Applesoft Programming Reference Manual.

My initial programming goal was to code the entire unit in Pascal. Through the use of POKEs, this was certainly possible, but clarity and speed of execution would be sacrificed. In the final designs, Pascal code is used exclusively with POKEs only being used for the display mode changes. To gain a full understanding of the design of the unit, the workings of the Apple graphics first should be understood. Pages 12 through 18 of the Apple][Reference Manual contain this information.

The Pascal variant record and the pointer type provide the necessary features to simplify the code. First the "Plot_Array" type was defined as a packed array of 0..15. This results in the ability to directly address a nibble (half-byte) without requiring the inefficiencies associated with the typical integer divide and mod operations. The "Plot_Type" variant record type contains a single variable, which is multiply defined as an integer, and as a pointer to the "Plot_Array". Thus if the integer value is set at 1024 (address of the beginning of the first text and low resolution page), and the zeroth element of the array is referenced, the actual location is that of memory address 1024. The variable "plot_buffer" is thus a 1024 byte array at address 1024 and does not require additional memory within the address space of the procedure.

With the addressing of the text/graphics page completed by simple

```
{$S+}

UNIT LOW_RES;

INTERFACE

  VAR
       color                              : Integer;

  PROCEDURE Plot (x, y: Integer);
  PROCEDURE HLIN (x1, x2, y: Integer);
  PROCEDURE VLIN (y1, y2, x: Integer);
  FUNCTION  SCRN (x, y: Integer): Integer;

  PROCEDURE Normal;
  PROCEDURE Inverse;
  PROCEDURE Flashing;

  PROCEDURE Print_Ch (Ch: Char);
  PROCEDURE Print    (line: String);
  PROCEDURE Print_Ln (line: String);
  PROCEDURE HTAB (x: Integer);
  PROCEDURE VTAB (y: Integer);

  PROCEDURE All_Text;
  PROCEDURE All_Graphics;
  PROCEDURE Mixed;
  PROCEDURE Home;
  PROCEDURE Top_Margin (margin: Integer);

  FUNCTION  Peek (address: Integer): Integer;
  PROCEDURE Poke (address: Integer;  ch: Char);


IMPLEMENTATION

  TYPE
       Plot_Array                         = Packed Array [0..2047] OF 0..15;

       Plot_Type                          =
         RECORD
           CASE Boolean
             OF
               true:   (pointer           : ^Plot_Array);
               false:  (Location          : Integer);
           END {CASE};

       Text_Array                         = Packed Array [0..1023] OF Char;

       Text_Type                          =
         RECORD
           CASE Boolean
             OF
               true:   (pointer           : ^Text_Array);
               false:  (location          : Integer);
           END {CASE};
```

subscripting, we can turn our attention to the interleaving of the lines within the page and the top/bottom orientation of the nibbles within the graphics byte. Repeated calculation of the subscript value is much too slow. Therefore, the initialization routine constructs a table of base addresses for both text and graphics. The table values represent the displacement into the array, given any "y" value. Thus addressing becomes indirect, and requires only a double subscript resolution. The graphics is only slightly more complicated, with the interleaving of the top and bottom nibbles within each byte.

Since the display mode of the characters is controlled by the use of a non-ASCII character set, it is necessary to translate all character strings depending on the mode. To simplify and speed this process, the mode procedures create a 256 byte translation table. To use this table, the ordinal of the ASCII character is used as a subscript into the table. The character at this location is the new character. Thus character translation also becomes a simple double subscript resolution.

The resulting code of this low resolution unit is straightforward and relatively fast. It should prove quite useful to any one desiring Pascal low resolution graphics.

```
PROCEDURE Mixed;

  VAR
       y                                : Integer;


       PA                               = Packed Array [0..0] OF Char;

       Memory_Type                      =
         RECORD
           CASE Boolean
             OF
               true:  (pointer          : ^PA);
               false: (location         : Integer);
           END {CASE};

VAR
       plot_buffer                      : Plot_Type;
       text_buffer                      : Text_Type;

       base_text                        : Array [0..23] OF Integer;
       base_graphics                    : Array [0..47] OF Integer;
       mode_table                       : Packed Array [0..255] of Char;

       null,
       blank                            : Char;

       i,
       y,
       top_y,
       cursor_x,
       cursor_y                         : Integer;


PROCEDURE Clear (ch: Char);

  BEGIN
    Fill_Char (text_buffer.pointer^ [  0], 120, ch);
    Fill_Char (text_buffer.pointer^ [128], 120, ch);
    Fill_Char (text_buffer.pointer^ [256], 120, ch);
    Fill_Char (text_buffer.pointer^ [384], 120, ch);
    Fill_Char (text_buffer.pointer^ [512], 120, ch);
    Fill_Char (text_buffer.pointer^ [640], 120, ch);
    Fill_Char (text_buffer.pointer^ [768], 120, ch);
    Fill_Char (text_buffer.pointer^ [896], 120, ch);
    cursor_x := 0;
    cursor_y := 0;
  END {Clear};

PROCEDURE Str_I (I: Integer;  VAR Item: String);

  VAR
       L                                : Integer;

  BEGIN
    L := I;
    Str (L, Item);
  END {Str_I};
```

```
FUNCTION  Peek {(address: Integer): Integer};


  VAR
       byte                              : Memory_Type;

  BEGIN
    byte.location := address;
    Peek := Ord (byte.pointer^ [0]);
  END {Peek};


PROCEDURE Poke {(address: Integer;  ch: Char)};


   VAR
        byte                             : Memory_Type;


   BEGIN
     byte.location := address;
     byte.pointer^ [0] := ch;
   END {Poke};


PROCEDURE All_Text;


  BEGIN
    Clear (blank);
    Poke (-16300, null);
    Poke (-16302, null);
    Poke (-16303, null);
    top_y := 0;
    cursor_x := 0;
    cursor_y := 0;
  END;


PROCEDURE All_Graphics;


  BEGIN
    color := 0;
    Poke (-16298, null);
    Poke (-16300, null);
    Poke (-16302, null);
    Poke (-16304, null);
    Clear (null);
    top_y := 0;
    cursor_x := 0;
    cursor_y := 0;
  END;
```

CASES NOW AVAILABLE

- AP101 Apple and Single Disk Drive $109
- AP102 Apple and Double Disk Drive 119
- AP103 Apple, 9 inch Monitor and Double Drive 129
- RS201 TRS-80, Expansion Unit and Double Drive 109
- RS202 TRS-80 Monitor and Accessories 84
- P401 Paper Tiger Printer 99
- P402 Line Printer II-Centronics 730 89
- CC90 Matching Attache Case 75

CASES SOON AVAILABLE

- RS204 Model III Radio Shack
- RS205 Radio Shack Color Computer
- AP104 Apple/// with Disk Drives and Silentype
- AP105 12 inch Monitor and Accessories

computer case company

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

```
BEGIN
  color := 0;
  Poke (-16298, null);
  Poke (-16300, null);
  Poke (-16301, null);
  Poke (-16304, null);
  Clear (null);
  Fill_Char (text_buffer.pointer^ [592], 40, blank);
  Fill_Char (text_buffer.pointer^ [720], 40, blank);
  Fill_Char (text_buffer.pointer^ [848], 40, blank);
  Fill_Char (text_buffer.pointer^ [976], 40, blank);
  top_y := 20;
  cursor_x := 0;
  cursor_y := top_y;
END;



PROCEDURE Plot {(x, y: Integer)};

  BEGIN
    x := x MOD 40;
    y := y MOD 48;
    plot_buffer.pointer^ [base_graphics [y] + x + x + (y MOD 2)] := color;
  END {Plot};



PROCEDURE HLIN {(x1, x2, y: Integer)};



  VAR
        base,
        x                                : Integer;



   BEGIN
     x1 := x1 MOD 40;
     x2 := x2 MOD 40;
     y  := y  MOD 48;
     base := base_graphics [y] + (y MOD 2);
     x := x1 + x1;
     WHILE x <= (x2 + x2)
       DO
         BEGIN
           plot_buffer.pointer^ [base + x] := color;
           x := x + 2;
         END {WHILE};
   END {HLIN};



PROCEDURE VLIN {(y1, y2, x: Integer)};

  VAR
        base,
        y                                : Integer;
```

```
BEGIN
  x  := x  MOD 40;
  y1 := y1 MOD 48;
  y2 := y2 MOD 48;
  base := x + x;
  FOR y := y1 TO y2
    DO
      plot_buffer.pointer^ [base_graphics [y] + base + (y MOD 2)] := color;
END {VLIN};


FUNCTION  SCRN {(x, y: Integer): Integer};

  BEGIN
    SCRN := plot_buffer.pointer^ [base_graphics [y] + x + x + (y MOD 2)];
  END {SCRN};

PROCEDURE Scroll;

  VAR
       y                                : Integer;

  BEGIN
    FOR y := (top_y + 1) TO 23
      DO
        Move_Left (text_buffer.pointer^ [base_text [y]],
                   text_buffer.pointer^ [base_text [y - 1]], 40);
    Fill_Char (text_buffer.pointer^ [base_text [23]], 40, blank);
    cursor_x := 0;
    cursor_y := 23;
  END {Scroll};




PROCEDURE HTAB {(x: Integer)};



  BEGIN
    cursor_x := (x - 1) MOD 39;
  END {HTAB};



PROCEDURE VTAB {(y: Integer)};



  BEGIN
    cursor_y := (y - 1) MOD 23;
    IF cursor_y < top_y
      THEN
        cursor_y := top_y;
  END {VTAB};
```


GALAXY SPACE WAR I

Galaxy Space War 1 (WAR1) is a game of strategy in which the player has complete control of his space fleet's tactical maneuvers. Each fleet battles its way toward the opponents galaxy in an attempt to destroy it and win the war. WAR1 simulates the actual environment encountered in a space war between two galaxies. Optimum use is made of Apple's high resolution graphics (HIRES) and colors in displaying the twinkling stars universe, the colored ships of each fleet, long range sensors colored illuminations, and the alternating blinking colors used in battles between ships. Complementing HIRES are the sounds of war produced by Apple's speaker.

WAR1 is played between Apple and a player or between two players. You may play with total knowledge of each others fleet or only ships sensor knowledge of the opponents fleet. Each player builds his starting fleet and adds to it during the game. This building process consists of creating the size and shape of each ship, positioning it, and then allocating the total amount of energy for each ship.

During a player's turn he may dynamcially allocate his ships total energy between his screen/detection and attack/move partitions. The percentage of the total energy allocated to each partition determines its characteristics. The screen/detection partition determines how much energy is in a ship's screens and the detection sector range of its short range sensors. The attack/move determines the amount of energy the ship can attack with, its attack sector range, and the number of sectors it can move in normal or hyperspace.

When an enemy ship is detected by short range sensors, it is displayed on the universe and a text enemy report appears. The report identifies the ship, its position, amount of energy in its screens, probable attack and total energy, a calculated detection/attack/move range, and size of the ship. Also shown is the number of days since you last knew these parameters about the ship. When a ship's long range sensor probes indicate the existence of an enemy presence at a sector in space, this sector is illuminated on the universe.

An enemy ship is attacked and destroyed with attack energy. If your attack energy breaks through his screens, then his attack energy is reduced by two units of energy for every unit you attack with. A text battle report is output after each attack. The program maintains your ship's data and the latest known data about each enemy ship. You may show either data in text reports or display the last known enemy positions on the universe. You can also get battle predictions between opposing ships. The text output calculates the amount of energy required to destroy each ship for different energy allocations.

APPLE® II, 48K, APPLESOFT ROM CARD, DISK II DOS 3.2
WAR1 DISK & MANUAL ...$39.95
(CA residents add 6% sales tax)
Write or call for more information

GALAXY
DEPT. A03
P.O. BOX 22072
SAN DIEGO, CA 92122
(714) 452-1072

COLOR COMPLIMENTS FOR BLACK AND WHITE

by Tom Jacobsen
Green Apples

Apple's color graphics is one of its most useful features. Problems crop up, however, when graphics programs are run on black and white monitors. Some color combinations are indistinguishable. The following charts are offered so that your future programs will work on both color and black and white sets. Choose a color on the left side. The numbers to the right are colors that will be distinguishable from it on a black and white set.

HIGH RES

0	1-3,5-7
1	0,3-4,7
2	0,3-4,7
3	0-2,4-6
4	1-3,5-7
5	0,3-4,7
6	0,3-4,7
7	0-2,4-6

LOW RES

0	1-15
1	0,3,5-7,9-15
2	0,3,5-7,9-15
3	0-2,4, 7-8,11,13-15
4	0,3,5-7,9-15
5	0-2,4,7-8,11,14-15
6	0-2,4,7-8,11,13-15
7	0-6,8-10,12,15
8	0,3,5-7,9-15
9	0-2,4,7-8,11,13-15
10	0-2,4,7-8,11,13-15
11	0-6,8-10,12,15
12	0-2,4-5,8,11,13-15
13	0-6,8-10,12,15
14	0-6,8-10,12,15
15	0-14

```
]LIST

10  REM

BLOAD ADDRESS FINDER

  BY VAL J GOLDING

100  HOME : VTAB 4: INPUT "ENTER
     BINARY FILE NAME TO BLOAD";F
     ILE$: PRINT  CHR$ (4)"BLOAD"
     FILE$:MEMSIZE = (( PEEK (978
     ) + 13) * 256) + 114
110  GOSUB 200:ST = LOBLOD + HIBL
     OD * 256:MEMSIZE = MEMSIZE -
     18: GOSUB 200:LNG = LOBLOD +
     HIBLOD * 256
120  PRINT : PRINT "[START ADR= "
     ;ST;"]  [LENGTH= ";LNG;"]": END

200 LOBLOD =  PEEK (MEMSIZE):HIBL
     OD =  PEEK (MEMSIZE + 1): RETURN
```

GREEN APPLE BITS

It is a rare occasion when a fledgling Apple user group (35 members) can publish its first newsletter with content suitable for reprint in the Orchard. Such is the case with the Green Apples-North Central Iowa Apple Users Group. Two of their articles appear on this page.

TUNE YOUR APPLE

by Mark Welty
Green Apples

The following information is from Mark Welty. The numbers given are to be poked for pitch when you are using the tone routines of the type published in Apple's Red Book (Ref. Manual). Anyone having the numbers to poke for the relative durations of standard note lengths should submit them for a future newsletter.

OCTAVE

Note	1st	2nd	3rd
C	192	96	47
C#	180	91	45
Db	180	91	45
D	171	86	42
D#	161	81	40
Eb	161	81	40
E	153	76	37
F	144	72	35
F#	136	68	33
Gb	136	68	33
G	129	64	31
G#	122	60	29
Ab	122	60	29
A	115	57	27
A#	108	54	26
Bb	108	54	26
B	102	50	24
C	96	47	23

```
PROCEDURE Normal;


  BEGIN
    FOR i := 0 TO 31
      DO
        mode_table [i] := Chr (i);
    FOR i := 32 TO 95
      DO
        mode_table [i] := Chr (i + 128);
    FOR i := 96 TO 127
      DO
        mode_table [i] := Chr (i - 96);
  END {Normal};



PROCEDURE Inverse;


  BEGIN
    FOR i := 0 TO 31
      DO
        mode_table [i] := Chr (i + 192);
    FOR i := 32 TO 63
      DO
        mode_table [i] := Chr (i);
    FOR i := 64 TO 95
      DO
        mode_table [i] := Chr (i - 64);
    FOR i := 96 TO 127
      DO
        mode_table [i] := Chr (i + 96);
  END {Inverse};



PROCEDURE Flashing;


BEGIN
  FOR i := 0 TO 31
    DO
      mode_table [i] := Chr (i);
  FOR i := 32 TO 63
    DO
      mode_table [i] := Chr (i + 64);
  FOR i := 64 TO 95
    DO
      mode_table [i] := Chr (i);
  FOR i := 96 TO 127
    DO
      mode_table [i] := Chr (i - 32);
END {Flashing};
```


KILL SURGES LIKE LIGHTNING!

AC power line surges are destructive, can cost you money, and can't be prevented. But you can stop them from reaching your sensitive electronic equipment with a Surge Sentry.

Surge Sentry acts in picoseconds to dissipate up to a 1,000,000 W, 100μ second surge. Triggers at 10% above nominal peak voltage. Works in parallel with the power line. Is easy to install for immediate protection. No complicated wiring or special tools required.

Several different models to choose from, including an OEM version. Call or write today for a *free* brochure.

SURGE SENTRY

It'll clean up your AC

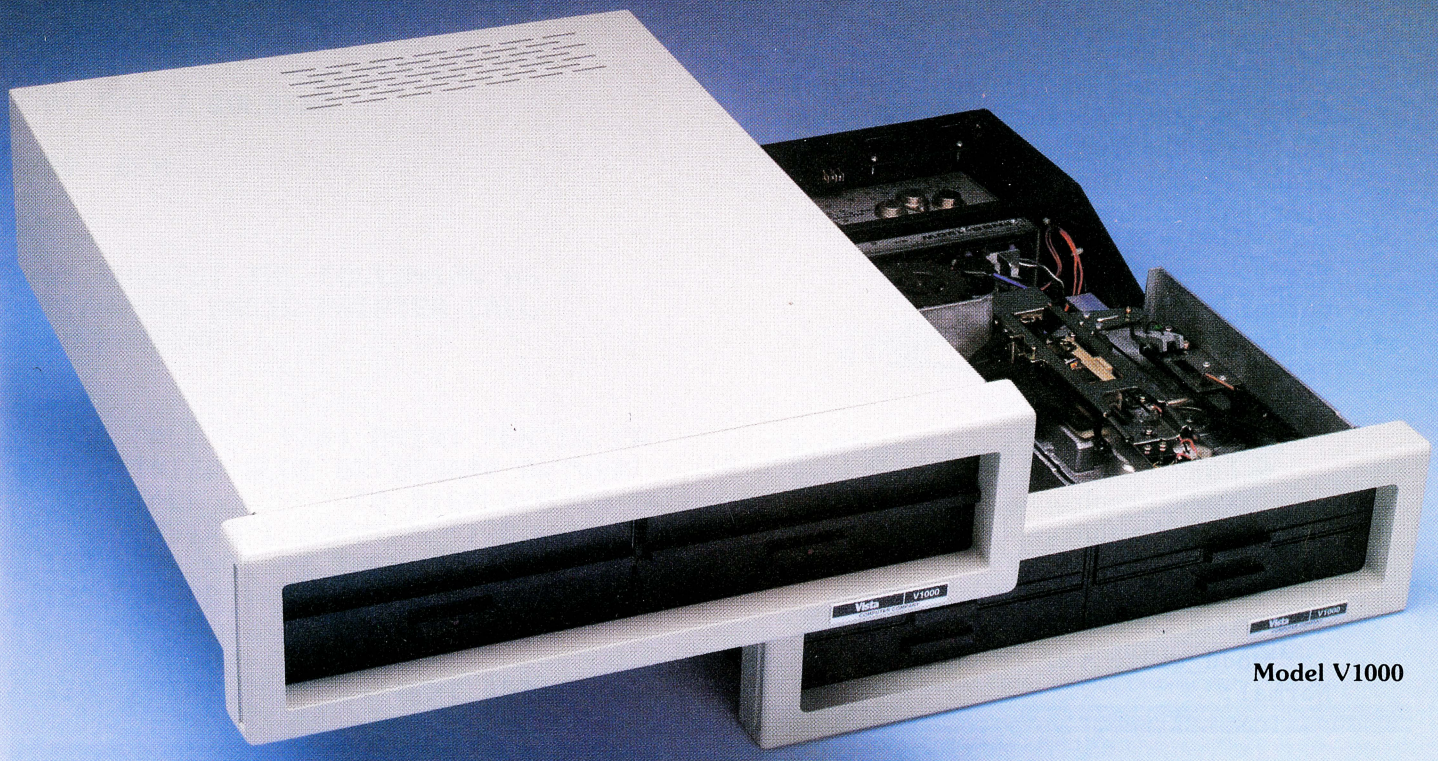
RKS ENTERPRISES, INC.

643 South 6th Street, San Jose, CA 95112
(408) 288-5565

DEALER INQUIRIES INVITED

```
PROCEDURE Home;

  VAR
        y                                   : Integer;

  BEGIN
    FOR y := top_y TO 23
      DO
        Fill_Char (text_buffer.pointer^ [base_text [y]], 40, blank);
    cursor_x := 0;
    cursor_y := top_y;
  END {Home};

PROCEDURE Print_Ch {(ch: Char)};

  BEGIN
    text_buffer.pointer^ [base_text [cursor_y] + cursor_x] :=
                         mode_table [Ord (ch)];
    IF cursor_x < 39
      THEN
        cursor_x := cursor_x + 1
      ELSE
        BEGIN
          IF cursor_y < 23
            THEN
              BEGIN
                cursor_x := 0;
                cursor_y := cursor_y + 1;
              END {THEN}
            ELSE
              Scroll;
        END {ELSE};
  END {Print_Ch};


PROCEDURE Print {(line: String)};

  VAR
        index                               : Integer;

  BEGIN
    FOR index := 1 TO Length (line)
      DO
        Print_Ch (line [index]);
  END {Print};

PROCEDURE Print_Ln {(line: String)};

  VAR
        index                               : Integer;

  BEGIN
    FOR index := 1 TO Length (line)
      DO
        Print_Ch (line [index]);
    WHILE cursor_x <> 0
      DO
        Print_Ch (' ');
  END {Print_Ln};
```

Vista



Model V1000

. . . Looking Out For You.

Eight Inch Floppy Disk Drive Subsystem Model V1000

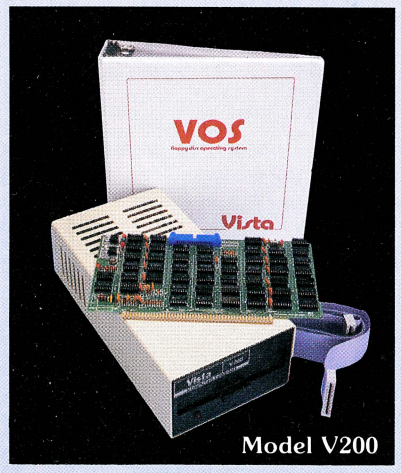
The V1000, Vista's sophisticated new disk drive subsystem, sets new standards for ease of access and use. Its innovative design permits disk drives to be mounted or removed quickly and easily for system reconfiguration or servicing.

Features:

- Deluxe chassis with internal slide allows easy access.
- Storage capacity from 250 kilobytes to 2.5 megabytes.
- Desk or rack mountable.
- Accomodates both single-sided and double-sided drives.
- Industrial quality cabinet with die cast front bezel.
- Drives pull out for easy service and maintenance.

Prices:

Cabinet with (2) single sided drives w/power supply	$1595.00
Cabinet with (2) double sided drives w/power supply	$2295.00
Cabinet (only)	$ 395.00



Model V200

Vista's Line of High Performance, High Reliability Products also Includes these Advanced Components

Daisy Wheel Printer Model V300

Features:

- 96-character proportional, bi-directional printing
- Interface - Parallel or RS232-C option

Prices:

V300-25cps	$1895.00
V300-45cps	$2195.00

Minifloppy Disk System, Model V200

Features:

- Storage capacity from 200K bytes to 1.2 megabytes
- Compatible with industry standard S100 main-frames.
- System software - Vista CP/M™ VOS Disk Operating System and Basic - E compiler.

Prices:

Starting as low as $695.00
V200-Exidy version $1199.00

Model V300

Vista Computer Company 1317 E. Edinger Avenue • Santa Ana, CA. 92705 • (714) 953-0523

Dealer Inquiries Call Toll Free (800) 854-8017

™ is a registered trademark of Digital Research

```
PROCEDURE Top_Margin {(margin: Integer)};

  BEGIN
    top_y := margin MOD 24;
  END {Top_Margin};



  BEGIN {Low_Res Initialization}
    null  := Chr (0);
    blank := Chr (Ord (' ') + 128);
    plot_buffer.location := 1024;
    text_buffer.location := 1024;
    top_y    := 0;
    cursor_x := 0;
    cursor_y := 0;
    FOR y := 0 TO 23
      DO
        BEGIN
          base_text [y] := (128 * (y MOD 8)) + (40 * (y div 8));
          base_graphics [2 * y] := 2 * base_text [y];
          base_graphics [(2 * y) + 1] := base_graphics [2 * y];
        END {FOR};
    FOR i := 128 TO 255
      DO
        mode_table [i] := Chr (i);
    Normal;
  END {Initialize}.
```


Apple Owners:

DEPRECIATION PROGRAM

- ★ 5 DEPRECIATION RATES
- ★ UP TO 99 YR TERM
- ★ RECORDS UP TO 600 ITEMS ON DISK
- ★ UP TO $1 MILLION FOR EACH ITEM
- ★ REPORTS EACH MONTH, QUARTER, OR ANNUALLY
- ★ BONUS DEPR., INVESTMENT CREDIT
- ★ PRO-RATES DEPRECIATION
- ★ UPDATE RECORDS EACH YEAR
- ★ EQUIPMENT INVENTORY
- ★ FISCAL YEAR BASED
- ★ CONVERT METHODS ANY TIME
- ★ AN ACCOUNTANTS DREAM

APPLESOFT.....32K MIN.....$225.00
HANDBOOK.....$5.00

VISA & M/C USERS – CALL
509-943-9004

MONEYDISK
P. O. Box 1531
RICHLAND, WA 99352

APPLE IS A REGISTERED TRADEMARK OF APPLE COMPUTER INC.
WA Residents, add 5% sales tax
DEALER INQUIRIES INVITED

VISA master charge

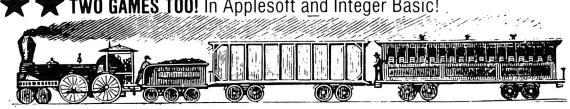
2 Games, Book & Command Chart $16.

APPLE USERS! Here's a special 4-item offer from Beagle Bros.!

★ **FIRST, OUR APPLE COMMAND CHART!** A heavy-duty 11x17 listing of EVERY Applesoft, Integer and DOS Command AND its function. Indispensable to any Apple user. **Experiment** with commands you **never knew existed!**

★ **PLUS our 36-pg APPLE TIP BOOK** of tricks, listings and articles on Customizing your Apple; Printing "unprintable" characters; Fixing garbaged program lines; Converting languages; Entering machine code . . . B Bros. Catalog too!

★★ **TWO GAMES TOO!** In Applesoft and Integer Basic!



Game #1: TEXTTRAIN! You keyboard control a text-format "freight train" on your Apple video layout! **Real-time switching & coupling** simulations! Race the on-screen clock & avoid a game-ending derail! One or more players test their skills, strategically assembling a user-defined series of computer freight cars! A fascinating demonstration of the versatility of your Apple! ($16, Winter Catalog)

Game #2: SUB SEARCH! Your assignment: Find a gang of invisible enemy subs on your Apple color graphics scope before your oxygen & fuel run out. Use your deep-sea scanner switch & video tracer to watch for blips & clues to each sub's location! ($12, Winter Catalog)

TextTrain & Sub Search are 2 more **machine language**-enhanced winners from Beagle Bros.! Designed & programmed exclusively for **the best** Micro there is!

(Use this coupon or separate sheet)

Rush TextTrain, Sub Search, the Apple Tip Book & Command Chart!

☐ **DISK $16.00** (BOTH Applesoft AND Integer)
☐ **CASSETTE $16.00** (SPECIFY Applesoft OR Integer)

All Shipped First Class within 24 hours!

PLEASE ADD $1.00 for First Class Shipping and Handling.

NAME$=**_____ **Amt. Enclosed=$_____
(Cal. residents, add 6% tax)

ADDR$=_____

CITY$=_____ **ZIP=**_____

VISA/MASTERCARD, include **ACCOUNT NO., EXP. DATE & SIGNATURE.**

ao31 VISA

Beagle Bros™ MICRO SOFTWARE

master charge THE INTERBANK CARD

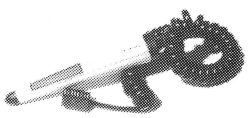
Mail to: **4315 Sierra Vista / San Diego, CA 92103**

```
]LIST

10  REM

PROPER
PRINTER
PROTOCOL

BY VAL J GOLDING

20  REM


WE SUGGEST THE FOLLOWING AS A
STANDARD BASIC PRINTER CALL
SUBROUTINE


100  CALL  - 936: PRINT "ENTER PR
     INTER SLOT OR BINARY DRIVER
     CALL          [0 DISPLAYS TO S
     CREEN]": INPUT " ";PR
110  IF  NOT PR THEN  RETURN
120  IF PR > 7 THEN  CALL PR: RETURN


130  PRINT D$;"PR#";PR: RETURN
```


IMPROVE Your Data Entry!

With ABT Apple Peripherals*

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



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

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

Available from your local Apple Dealer*

ADVANCED BUSINESS TECHNOLOGY, INC.

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

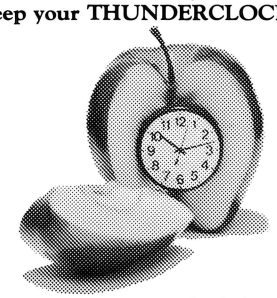
*Trademark of APPLE COMPUTER, INC. **Trademark of McGRAW-HILL

THUNDERCLOCK PLUS™

PUT TIME AND REMOTE CONTROL IN YOUR APPLE II

The THUNDERCLOCK PLUS is two peripheral systems on one card for your APPLE II OR II PLUS. An accurate, reliable, real-time clock/calendar and an interface for the popular BSR X-10 Home Control System.

The THUNDERCLOCK clock/calendar makes accurate time and date available to your programs: month, date, day-of-week, hour, minute, and second, in any of four software selectable formats. On-board batteries keep your THUNDERCLOCK running when your APPLE II is turned off - for up to four years before battery replacement. On-card 1K firmware makes reading or setting the time easy from APPLESOFT or INTEGER BASIC, PASCAL, or assembly language programs. And it provides software selectable interrupts at any of three rates: 64, 256, or 2048 interrupts/second.

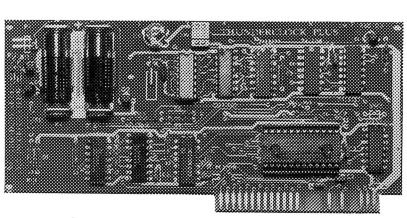


THE PLUS

Add THUNDERWARE'S X-10 ULTRASONIC INTERFACE OPTION to your THUNDERCLOCK and your programs can send all 22 BSR X-10 commands so you can remotely control lights and appliances. A full 128 dim/bright levels. And a powerful disk software package! The THUNDERWARE SCHEDULER software lets you create schedules to control lights, appliances, security systems, or almost any other electrical device. The software includes: SCUTIL- the SCHEDULER utility that lets you make or change a schedule, and SCHED- executes your schedules in real-time using the THUNDERCLOCK. SCHED runs in the 'background' so you can run other programs in the 'foreground'.

The THUNDERCLOCK PLUS is a SYSTEM for your APPLE II. Supported by intelligent, easy to use firmware, a powerful software package, and good documentation!

Available through your dealer.



BSR X-10 is a trademark of BSR (USA) LTD.
APPLE II is a trademark of APPLE COMPUTER, INC

Suggested retail prices:

THUNDERCLOCK PLUS.....$139
Clock/calendar card with batteries and user's manual

X-10 INTERFACE OPTION.....$49
BSR X-10 Ultrasonic interface, disk with SCHEDULER SOFTWARE & demos, and user's manual

PASCAL SOFTWARE.....$29
Disk with PASCAL interface for clock and X-10 interface, and user's guide

MANUALS ONLY, each.....$5

California residents add 6% sales tax

If your dealer doesn't carry the THUNDERCLOCK PLUS:
ORDER TOLL FREE (VISA/MC) CALL:
800-227-6204 EXT 307 (Outside California)
800-632-2131 EXT 307 (California Only)

OR WRITE TO:

THUNDERWARE INCORPORATED
P.O. Box 13322, Oakland, CA 94661

SINGLE DRIVE COPY.3

by Steve Adams
Call -A.P.P.L.E.

Single Drive Copy is the final version of a program I first completed in August 1979. At that time, Apple Computer's only copy program required two disk drives, and although several user group newsletters had dropped a few hints on the use of RWTS, no one had written a "public domain" program to allow copying a diskette using a single drive.

The original program copied a diskette by reading as many of the diskette's 455 sectors into memory as the computer would hold, and then dumped them onto the new diskette. Starting at Track 0, Sector 0, and using a simple nested loop, the program copied a diskette in four passes.

Although the program was effective, it wasted time by copying unused sectors. Some of these would generate read errors, causing the disk drive to emit obnoxious grinding noises in protest. It was written in Integer Basic for DOS 3.2 systems.

About a year later, I got rid of the nested loop and replaced it with a routine that would "read" the Volume Table of Contents (VTOC) to find which sectors were in use. The program then copied only those sectors, reducing copying time and silencing the drive.

The final (and last) version of the program now includes a method of determining which version of DOS is in use, thereby allowing anyone with a Disk II to use it without modification. It isn't the fastest program available, since I haven't learned machine language yet, but it works and it's free. Here's how it works:

Line 0 initializes variables. Due to the nature of the program, this is necessary so the end-of-variables pointer (CV) doesn't move upwards in memory and wipe out other data. Line 1030 does a quick check of memory size by PEEKing memory location 77 (77=$4D, the hi-order byte of the HIMEM pointer) and determining where in memory the IOB starts. Actually, variable "IOB" is equal to one byte below the IOB so that I could use the "IOB+" values in the RWTS section of the DOS Manual.

Since using RWTS from a program requires a controlling subroutine, line 1050 loads the subroutine into Page 0. See page 94 of the DOS Manual for an explanation. Note the POKE for location 1: for a 48K machine the value for "IOB" is greater than 32767, hence negative; for a 32K machine "IOB" is less than 32767. This calculation compensates for these different memory configurations. Using the RWTS from where it normally resides, rather than inserting it into the program, merely shows that it can be done; whether it's "better" to do it this way is probably debatable. But then, figuring out how to do it taught me something.

Lines 1065 and 1070 transfer the entire VTOC into memory starting at $02D0 (720 decimal). This saves disk access time later in the program when the VTOC is "decoded" to determine in-use sectors. See page 96 of the DOS Manual for an explanation of the variables in line 1065. Using memory below LOMEM frees more RAM for use by the program. That portion of memory now containing the track bit maps ($0308-$0393) is unused by both DOS and the Monitor.

"BITMAP" is the location of the first track bit map, that of Track $0. Line 1080 also determines which version of DOS is in use by reading byte $2 of the VTOC. If this value is $F, DOS 3.3 is resident; if $C, DOS 3.2. The variable "VER," therefore, equals the number of sectors per track.

Line 2010 defines the variable "TBL," the pointer to the start of the memory area in which the program stores the names of in-use sectors. This routine places the start of the table at one byte above the "end of variables" pointer. "PTR" is the running pointer to the next location to store these names and is initially set to the beginning of the table.

Line 2020 is just a gentle reminder to not bother your Apple while he/she is thinking (my Apple answers to the name of "Louis," but yours may not share the gender).

The next section of the program (lines 2040-3030) decodes the track bit maps to determine which sectors on the original diskette are in use. Refer to "Track Bit Maps" in the DOS Manual (page 134) for an in-depth explanation. Starting at byte $38 of the VTOC ("BITMAP"), the program looks at successive pairs of bytes. If the first byte of the pair contains $FF (255) then sectors $8-$F ($5-$C for DOS 3.2) are empty. If so, the program reduces "SEC" to the highest sector value coded in the second byte of the track bit map and continues. If not, it jumps to line 2070 and converts the value of that first byte to binary, bit by bit. As each binary bit is converted, the subroutine at line 3020 builds the table; if a bit is 0, that track and sector contain information, and those values are POKEd into the table. If not, SEC is decre-

mented and the program checks the next bit of the pair of bytes.

Line 2060 performs a function similar to that of 2050, only for the second byte of the pair in each track bit map. Once all the track bit maps have been decoded, the program jumps to the copy loop.

Line 4010 recalls the variable "PTR" which is now equal to one byte above the upper end of the table we just created; it is at this location that the program starts dumping data from the diskette into memory. This line also POKEs this initial value into applicable places in the IOB.

In line 4020, the variable "REPS" is set equal to the number of 255-byte blocks of memory between the top of the table and the start of the program. With a completely full diskette, a 48K machine will hold 127 sectors. Therefore, this program will copy a completely full 16-sector diskette in five passes, and a 13-sector diskette in four.

The remainder of the copy loop alternately reads and writes data. Note the redefinition of the variable "PTR" in line 4030; during the read process, "PTR" steps upwards through the table, allowing line 4040 to read values of used track and sector combinations from the table. For each such pair of values, the hi-order location of the data buffer is POKEd into the IOB.

Line 4050 CALLs RWTS. Line 4060 updates the location of the data buffer and increments "PTR." If PTR has not yet reached the end of the table (which it won't do until the last of the track and sector combinations has been loaded into memory), another sector is dumped into memory.

When Apple's memory is full (exit of the REP loop), line 4090 prompts the user to switch diskettes. After this is done, the program repeats the REP loop, only this time in a "write" mode. The "OLDPTR-PTR" switches in lines 4030 and 4110 keep track of where in the table you left off at during the last read process.

Note line 4060 one more time. Once the program reads the last track and sector pair from the table, the program directs a diskette switch (if CMD=1); if the program had been in a "write" mode (CMD=2), then you are finished. The program jumps to line 4070 and ENDS.

I tried to leave the structure of the program in as much of a "top-down" appearance as possible, mainly to make the program easy to understand. It could probably be made to run faster, but I was not concerned with speed of execution.

A couple of notes. If you have manually changed HIMEM to something other than normal (that set by Apple upon DOS boot-up), the program will not be able to find the IOB and strange things will happen; my Apple started playing music. If you try to copy onto a write-protected diskette, the program will not report any errors; it will whiz away to completion (you will of course switch diskettes on command). When you try to check your "work" by CATALOGing the new diskette, you will not get the expected results and think the program doesn't work. This caused me an hour or so of fruitless bug-chasing while I was writing the program.

This program is written in Integer Basic; converting it to work under Applesoft should be fairly straightforward. Perhaps a reader will do so, and submit it.

One last caution: until you are *absolutely sure* you have keyed in the program correctly, cover the write-protect cutout on your original diskette. This will save you from wiping out a valuable diskette due to program errors. It'll also help me sleep better. I would appreciate any suggestions for improvement to this program.

Integer BASIC

```
  0 A=B=PTR=LOC=RWTS=IBTRK=IBSECT=
    IBBUFP=REP=REPS=CMD=TBL=IOB=
    TRK=SEC=VER=BYTE1=BYTE2=OLDPTR=
    BITMAP=BUFLO=BUFHI=IBVOL=IBCMD=
    0

 10 REM ****************************
 20 REM *                          *
 30 REM *   SINGLE DRIVE COPY.3    *
 40 REM *                          *
 50 REM ****************************
 60 REM *                          *
 70 REM *        STEVE ADAMS       *
 80 REM *     RFD #1, BOX 432      *
 90 REM *  PLATTSBURGH, NY 12901   *
100 REM *      (518)  561-8100     *
110 REM *                          *
120 REM ****************************
130 REM

1000 TEXT : CALL -936: VTAB 5: TAB
     11: PRINT "SINGLE DRIVE COPY.3"
     : PRINT : PRINT : TAB 13: PRINT
     "BY STEVE ADAMS"

1010 VTAB 20: PRINT "INSERT THE DISKE
     TTE YOU WISH TO COPY ANDHIT RETU
     RN TO BEGIN.": GOSUB 5010

1020 REM
1021 REM  *** FIND THE IOB ***

1022 REM
1030 A= PEEK (77): IF A>94 THEN
     A=A-256:IOB=(A+33)*256+231

1040 REM
1041 REM  *** LOAD CONTROLLING ***
1042 REM  *** SUBROUTINE IN    ***
1043 REM  *** PAGE 0           ***
1044 REM
```

```
1050 POKE 0,169: POKE 1,IOB/256+
     255*(IOB<0): POKE 2,160: POKE
     3,232: POKE 4,32: POKE 5,217
     : POKE 6,3: POKE 7,96
1060 REM
1061 REM  *** LOAD THE VTOC INTO ***
1062 REM  *** MEMORY STARTING    ***
1063 REM  *** AT $02D0           ***
1064 REM
1065 IBVOL=IOB+4:IBTRK=IOB+5:IBSECT=
     IOB+6:IBBUFP=IOB+10:IBCMD=IOB+
     13
1070 POKE IBVOL,0: POKE IBTRK,17
     : POKE IBSECT,0: POKE IBBUFP-
     1,208: POKE IBBUFP,2: POKE
     IBCMD,1: CALL RWTS
1080 BITMAP=776:VER= PEEK (722)+
     1
2000 REM
2001 REM  *** THE TABLE STARTS ***
2002 REM  *** AT "TBL"         ***
2003 REM
2010 TBL= PEEK (204)+ PEEK (205)
     *256+1:PTR=TBL:TRK=0
2020 VTAB 10: TAB 15: PRINT "I'M THI
    NKING"



2030 REM
2031 REM  *** READ "BIT MAPS" IN ***
2032 REM  *** THE VTOC AND CON-  ***
2033 REM  *** VERT TO BINARY     ***
2034 REM
2040 FOR TRK=0 TO 34:BYTE1=BITMAP+
     TRK*4:BYTE2=BYTE1+1:SEC=VER-
     1
2050 A= PEEK (BYTE1): IF A#255 THEN
     2070:SEC=VER-9
2060 A= PEEK (BYTE2): IF A=248 AND
     VER=13 OR A=255 AND VER=16 THEN
     2100
2070 B=A/128:A=A-B*128: GOSUB 3020
     :B=A/64:A=A-B*64: GOSUB 3020
     :B=A/32:A=A-B*32: GOSUB 3020
     :B=A/16:A=A-B*16: GOSUB 3020
     :B=A/8:A=A-B*8: GOSUB 3020
2080 IF SEC<0 AND VER=13 THEN 2100



2090 B=A/4:A=A-B*4: GOSUB 3020:B=
     A/2:A=A-B*2: GOSUB 3020:B=A:
      GOSUB 3020
2095 IF SEC=7 AND VER=16 OR VER=
     13 THEN 2060
2100 NEXT TRK: GOTO 4010
3000 REM
3001 REM  *** IF THE TRACK BIT  ***
3002 REM  *** MAP INDICATES AN  ***
3003 REM  *** IN-USE SECTOR,    ***
3004 REM  *** POKE TRK & SEC    ***
3005 REM  *** INTO THE TABLE    ***
3006 REM  *** STARTING AT "TBL" ***
3007 REM
3020 IF B THEN 3030: POKE PTR,TRK:
      POKE PTR+1,SEC:PTR=PTR+2
3030 SEC=SEC-1: RETURN
4000 REM
4001 REM   **** COPY ****
4002 REM
4010 BUFLO=(PTR) MOD 256:BUFHI=(
     PTR)/256: POKE IBBUFP-1,BUFLO:
      POKE IBBUFP,BUFHI
4020 REPS= PEEK (203)-BUFHI-( PEEK
     (202)<BUFLO):OLDPTR=TBL:TBL=
     PTR
4030 FOR CMD=1 TO 2: CALL -936: IF
     CMD=1 THEN PRINT "READING":
      IF CMD=2 THEN PRINT "WRITING"
     : POKE IBCMD,CMD:LOC=BUFHI:
     PTR=OLDPTR
4040 FOR REP=1 TO REPS: POKE IBTRK,
      PEEK (PTR): POKE IBSECT, PEEK
     (PTR+1): POKE IBBUFP,LOC
4045 VTAB 3: PRINT "TRACK="; PEEK
     (IBTRK);: TAB 12: PRINT "SEC="
     ; PEEK (IBSECT);" "
4050 CALL RWTS



4060 LOC=LOC+1:PTR=PTR+2: IF PTR#
     TBL THEN 4080
4070 IF CMD=1 THEN 4090: CALL -936
     : PRINT "FINISHED": END
4080 NEXT REP
4090 FOR A=1 TO 1000: NEXT A: CALL
     -936: VTAB 5: PRINT "INSERT
     THE ";: IF CMD=1 THEN PRINT
     "DUPLICATE";
4100 IF CMD=2 THEN PRINT "ORIGINAL"
     ;: PRINT " AND HIT RETURN":
      GOSUB 5010
4110 NEXT CMD:OLDPTR=PTR: GOTO 4030



5000 REM
5001 REM  *** WAIT FOR 'RETURN' ***
5002 REM
5010 POKE -16368,0
5020 IF PEEK (-16384)#141 THEN 5020
     : POKE -16368,0: CALL -936:
      RETURN
```

ADVERTISER INDEX

	PAGE
Advanced Business Technology	91
Advanced Computer Products	69
Applefest '81	21
ATV Research	44
Automated Simulations	7
Avant-Garde Creations	59
Beagle Bros.	90
Computer Case Company	83
Computer Simulations	79
Computer Station	24
Computers Are Fun	IFC
dilithium Press	39
Eastern House Software	29
Galaxy	85
Highlands Computer Services	50
Information Unlimited Software	96
Instant Software Inc.	48, 49
Interface Age	35
International Apple Core	60
LJK Inc.	71
Lobo Drives International	2
Microsoft	5
Money Disk	90
Mountain Computer	BC
Nibble	56
Programma International	31, IBC
Rainbow Computing	16
RKS Enterprises Inc.	87
Rochester Data	59
San Francisco Apple Core	75
Soft CTRL Systems	55
Software Publishing Corp.	1
SSM Microcomputer Products	15
Stoneware Microcomputer Products	23, 65, 78
Synergistic Software	76
Thunderware, Inc.	91
Verbatim	8
Videx	67
Vista Computers	89
Zeus Computer	25

INTERNATIONAL APPLE CORE™

Apple Orchard SUBSCRIPTIONS

P.O. Box 1493, Beaverton, Oregon 97075

Please enter a one-year subscription to Apple Orchard for:

NAME _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

COUNTRY _____

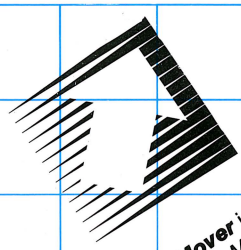
Annual subscription rate: $10.00 per year.
First class postage: $5.00 per year additional (required for Canada, Mexico, APO, and FPO addresses).
Overseas and other foreign air mail postage (required): $10 per year additional.

Charge it to my:
☐ VISA ☐ MasterCard No. _____ Expiration date _____

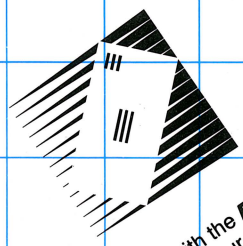
Or make check or money order payable to "International Apple Core" and return with this form to:

Apple Orchard Subscriptions
P.O. Box 1493
Beaverton, Oregon 97075

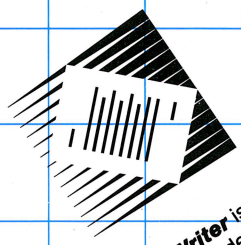
TOTAL REMITTANCE ENCLOSED: $ (USA) _____



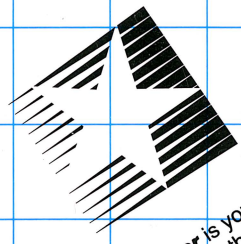
EasyMover is the first *Electronic Mail System* which combines the versatility of a word processor with the ability to move or transmit text files to another computer.



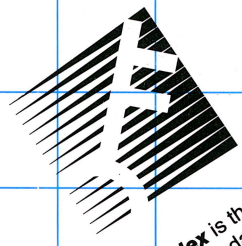
Now, with the ***EasyMailer*** system, your Apple can be used to greatly reduce time, paperwork, and money spent on form letters, mailing labels, and other documents.



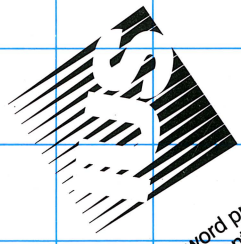
EasyWriter is a powerful word processor designed for the people who want the *best*.



TellStar is your computer window to the celestial objects as they appear at your home, or any location on the Earth you desire.



Datadex is the key to interactive data management for your Apple computer and the heart of an automated office's operation.
(Available first quarter 1981.)

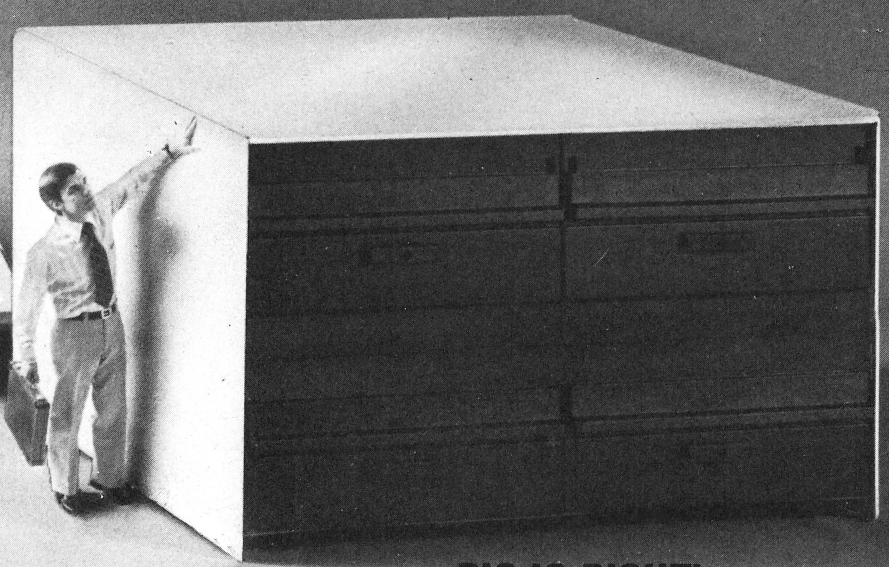


IUS is word processing, data management, data communications, and educational software for the Apple computer. Write or call your local dealer to find out more about the automated office made easy!

IUS (***I***nformation ***U***nlimited ***S***oftware, ***I***nc.)
281 Arlington Avenue, Berkeley, CA 94707
415-525-9452

Apple is a (tm) of Apple Computers, Inc.
EasyWriter copyright 1980, Cap'n Software
Datadex copyright 1980, Sonoma Softworks
TellStar copyright 1980, Scharf Software Services

Our new Prodisk card gives the small businessman big business storage.



BIG IS RIGHT!

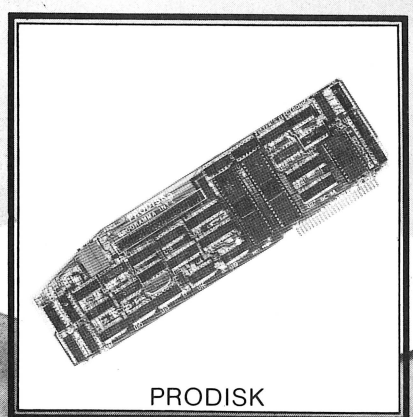
Up to 5 million bytes of on-line storage with 8" floppies. And, without stealing power from your Apple II*!

Your business is more complicated than anyone realizes. Or, you're growing faster than expected. Does that mean you have to trade in your reliable Apple II for a bigger, more costly system? Or, is there a simple, reasonable solution?

You bet there is, now that Programma International has introduced the Teeter Electronics PRODISK controller card.

PRODISK is like having the storage capability of ten Apples! Because with just four 8" floppy disk drives, its on-line storage capacity goes to a business-size 5 million bytes. Plus it delivers high-speed transfer of a half million bits per second.

With storage and speed like that you can really get a handle on your entire business. And, it won't interfere with your Apple's operation. You see, the new PRODISK card is powered from the drives, not the Apple. Since there's no significant power drain, other cards can be used with no problem.



PRODISK

Technically Inclined?

The card operates under Apple DOS 3.1 or 3.2, with 48K Apple II. It has full compatability with mini-disks. Handles from one to four 8" floppy drives. Single or double density disks use DMA transfer techniques . . . with high speed transfer of half million bits per second.

Priced Right

The amazing PRODISK controller card is priced at $645.00. And it's a tax deductible business expense. Its low price works out to be even lower! Same for the special Programma 8" floppy disk drives (800 or 850 Shugart equivalent). Example: two single sided drives, priced at $1549.00.

Get big business storage capacity for your small business right now . . . with PRODISK, available at your better computer stores, or direct from Programma International.

Big power for small business

PROGRAMMA

Programma International Inc.
2908 No. Naomi Street
Burbank, CA 91504
(213) 954-0240

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

Mountain Computer
can now
EXPAND
Your Apple II® Peripheral Capacity
EXPANSION CHASSIS

Quality You Expect

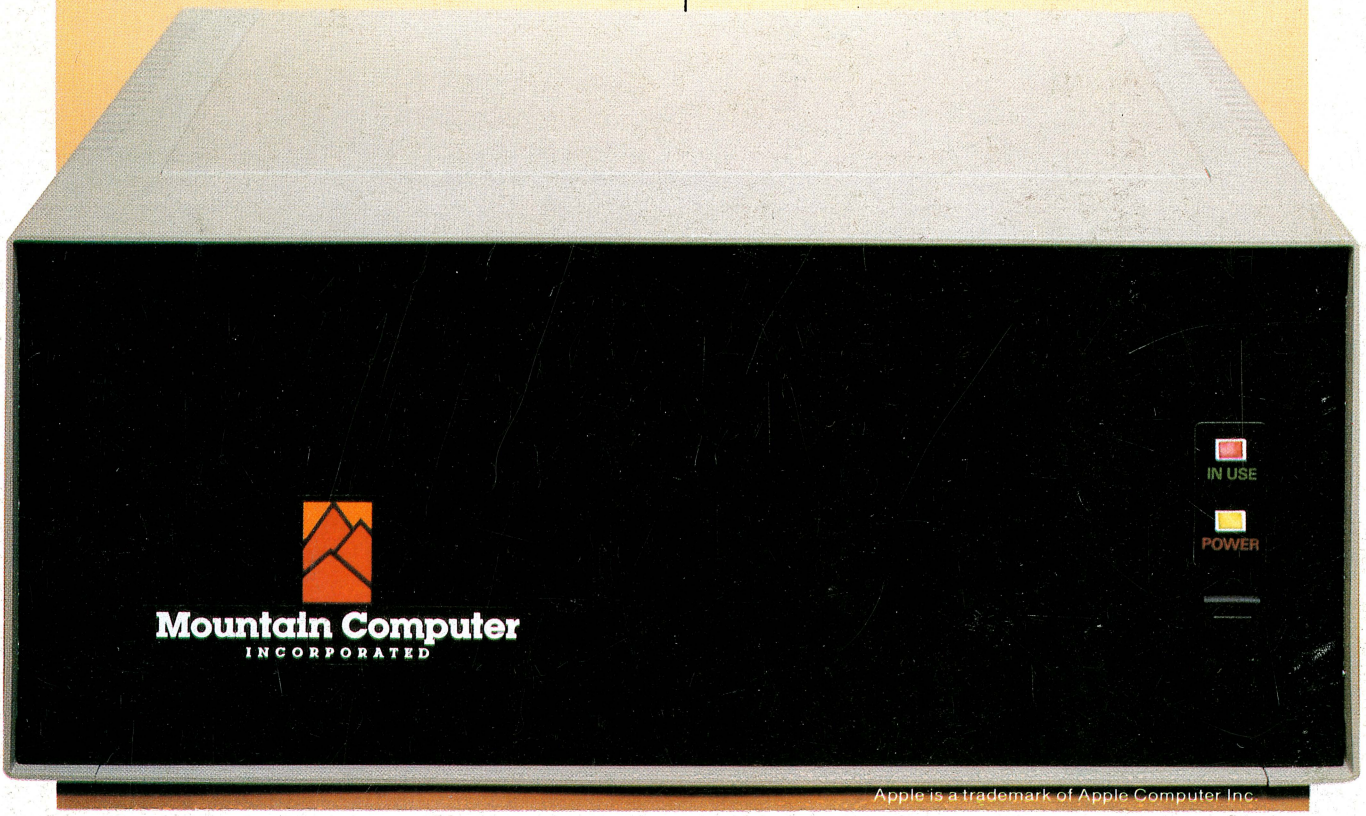
Eight more slots for your Apple! Now you can bank-select eight more peripheral slots with immediate or deferred software commands—like having up to 15 peripheral cards "on line"—or use the Select/Deselect switch mounted on the front panel. Expansion Chassis' heavy-duty power supply is primarily for peripherals, without the heavy demand of motherboard support chips required in your Apple. This means much more power is available for peripherals than in your Apple itself! If you've run out of room in your Apple—Expansion Chassis is your answer. Drop by your Apple dealer for a demonstration, or contact Mountain Computer for the location of the dealer nearest you.

Performance You Demand

- Eight mirror image I/O slots of the Apple
- Fully buffered, bi-directional data lines
- Apple II compatible interface card
- Dual selection capability; hardware or software
- Immediate or deferred selection in software mode
- From BASIC, a single POKE command turns the chassis ON or OFF
- Compatible with all software
- Dedicated power supply with approved power transformer

Mountain Computer
INCORPORATED

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



Apple is a trademark of Apple Computer Inc.