$2

Washington Apple Pi

The Journal of Washington Apple Pi, Ltd.

Volume 6 December 1984 Number 12

Highlights

ASSEMBLY LANGUAGE SPECIAL
FORTH FLOATING POINT ARITHMETIC
USING REMEMBER II
BRINGING BACK GAMESIG

In This Issue

Software City®

OUR APPROACH

- All Products Always Discounted
- Reference Literature & Guides For Browsing
- Demo Machines
- Special Orders Our Speciality
- Gift Certificates Available
- Rainchecks For Out-Of-Stock Specials
- Comfortable Showroom Environment
- Special Corporate, Government, Educational & User Group Accounts

SOFTWARE

- Business
- Scientific
- Education
- Recreation
- Programming
- Home & Personal
- Specialized

HARDWARE

- Monitors
- Printers
- Drives
- Interfaces
- Modems
- Expansion Boards
- Cables

LITERATURE

- Books
- Magazines
- Free Product Information

ACCESSORIES & SUPPLIES

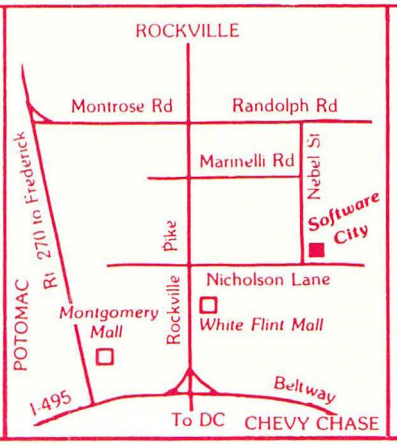
- Media & Media Storage
- PC Protection & Care Products
- Paper & Custom Forms

STORE HOURS

Mon-Fri	10-6
Thurs	10-8
Sat	10-5

468-1001

11621 Nebel Street (Extended)
Rockville, Maryland 20852



Complete coupon to be placed on Software City Special Mailing List.

Name _____

Address _____

_____

Phone _____

Type PC at Work _____

Type PC at Home _____

OFFICERS & STAFF

President	- David Morganstein	(301) 972-4263	Demonstrations	- Signe Larson	(703) 524-4541
Vice Pres-Programs	- Tom Warrick	(301) 656-4389	General Counsel	- Jim Burger (Burger & Kendall)	
Vice Pres-SIGs	- Bob Platt	(806) 378-1065			day (202) 293-7170
Treasurer	- Edward Myerson	(703) 759-5479	Membership	- Dana Schwartz	(301) 654-8060
Secretary	- Nancy Little	(301) 762-3215	Program	- Cara Cira	(301) 468-6118
Directors	- Bernie Benson	(301) 951-5294	Publicity Chairman	- Hunter Alexander	(703) 820-8304
	- Peter Combes	(301) 251-6369	Public Relations	- Lee Raesly	(301) 460-0754
	- J.T. (Tom) DeMay Jr	(301) 779-4632	Rules & Elections	- Bob Platt	(806) 378-1065
	- Bruce Field	(301) 340-7038	School Coordinators:		
	- Nancy Philipp	(301) 924-2354	Virginia	- Barbara Larson	(703) 451-9373
	- Jay Thal	(202) 244-3649		- Nancy Strange	(703) 691-1619
	- Rich Wasserstrom	(703) 893-9147	Mont. Co. MD	- Margie Stearns	(301) 657-2353
Editor	- Bernie Urban	(301) 229-3458	Pr. Geo. Co. MD	- Conrad Fleck	(301) 699-8200
Associate Editor	- Genevie Urban	(301) 229-3458	Special Publctns.	- Betsy Harriman	(202) 363-5963
Journal Staff:			SYSOP	- Tom Warrick	(301) 656-4389
Store Distrbtn.	- Ray Hobbs	(301) 490-7484	Tutorials	- Steve Stern	(301) 881-2543
Columnists:				- Leon Raesly	(301) 460-0754
Applesoft	- J.T. (Tom) DeMay Jr	(301) 779-4632	Volunteer Coord.	- Sue Roth	(703) 356-9025
DisabledSIG	- Jay Thal	(202) 244-3649	SIG Coordinator	- Bob Platt	(806) 378-1065
EDSIG	- Peter Combes	(301) 251-6369	SIG Chairmen:		
Pascal (PIG)	- Mike Hartman	(301) 445-1583	Appleseeds	- Ian Thal	(202) 244-3649
Q & A	- Bruce Field	(301) 340-7038	Apple ///	- Jerry Chandler &	(703) 790-1651
Telecomm	- Dave Harvey	(703) 527-2704		- Bill Hershey	(301) 588-1992
VisiCalc	- Walt Francis	(202) 966-5742	Apple IIc	- Chuck Holzwarth	(703) 751-7575
Review Coord.:			CP/M	- Charles Franklin	(301) 997-9138
Hardware	- Scott Rullman	(301) 779-5714	DisabledSIG	- Jay Thal	(202) 244-3649
Software	- Raymond Hobbs	(301) 490-7484	EDSIG	- Peter Combes	(301) 251-6369
Group Purchases	- Rich Wasserstrom	*(301) 951-3919	Forth SIG	- Kevin Nealon	(703) 280-1136
	*(Call Kevin at office)		Frederick Slice	- Randy Pasley Jr.	(301) 695-9416
Disketeria Staff:	- John Malcolm	(301) 384-1070	GAMESIG	- Ronald Wartow	(301) 654-4439
	- Dave Weikert, Joy Aso, Ed Lang,		LAWSIG	-	
	- Jim & Nancy Little, Pat Foreman,		LISASIG	- Gordon Stubbs	(703) 750-0224
	- Fred Edwards, Gordon Stubbs		LOGOSIG	- Nancy Strange	(703) 691-1619
New Disks	- Dana Schwartz	(301) 725-6281	SIG Mac	- Steve Hunt	(301) 262-9080
Pascal Lib.	- John Dyer	(703) 538-5636	NEWSIG	- Bernie Benson	(301) 951-5294
CP/M Lib.	-		Pascal (PIG)	- Harry Bishop/	(703) 931-4937
SigMac Lib.	- Tony Anderson	(301) 277-0386		- Jim Harvison	(301) 593-2993
Head Reading Lib.	- Walt Francis	(202) 966-5742	PI-SIG	- Raymond Hobbs	(301) 490-7484
Apple Tea Coord.	- Paula Benson	(202) 951-5294	STOCKSIG	- Robert Wood	(703) 893-9591
Arrangements	- Jim Taylor	(301) 926-7869	Telecomm. SIG	- George Kinal	(202) 546-7270

Washington Apple Pi, Ltd.
8227 Woodmont Avenue, Suite 201
Bethesda, MD 20814
Office (301) 654-8060

ABBS (301) 986-8085 - 24 hours/day
ABBS-Buy and Sell (301) 871-7978 - 7:30 PM to 7:30 AM

Copyright 1984, Washington Apple Pi, Ltd.

EDITORIAL

What's in it for me? We often hear similar words which are more temperate and diplomatic from some members (or potential members) of our organization. However, we are trying to get your attention and hopefully have succeeded for the moment.

Any volunteer organization succeeds or fails on the basis of the extent and quality of participation by its membership. But their participation should be beneficial to them in some way or other. Focus, if you will, on the contributors to these pages. We do not offer page rates, nor are we likely to do so. So, "What", you may ask, "is in it for me?" Well, it's fun to some of us to write about something that is our avocation rather than our vocation. We write to get things off our chests, for relaxation, perhaps, or for the enjoyment of seeing our names in print. Others write as a way of repaying the group for services which they have received or as a down payment for services which they hope to receive. Some contribute merely on the basis of feeling a sense of comraderie. Whatever the reason, continuez s'il vous plait.

We have over these now six years published the good works of many individuals, some of whom have contributed extensively, e.g. our illustrious President David Morganstein, Dr. Wo of Pascal fame, Mark Crosby, Lee Raesly, our Q&A answerman Bruce Field, Peter Combes, Nancy Strange, Bill Wurzel, Ray Hobbs, Bob Platt, Tom Warrick, and numerous others. Joining the ranks is John Love, who in this issue alone contributes 15 1/2 pages of copy. We could have drawn his contribution out over several issues but decided to put it into one so that he might be able to use this issue as an example of what he can do. That's what's in it for him!

Not to be ignored are the many other fine articles which are included in this issue.

A reminder - with the Holidays hard upon us, a membership to Washington Apple Pi might be just the right stocking stuffer.

PRESIDENT'S CORNER

by David Morganstein

NEW MEMBERS' MEETINGS. Again this year, Bernie and Paula Benson have scheduled a volunteer to be at the office every Thursday evening, from now until the end of June. This meeting provides you, the new computer owner or new member of the WAP, an informal opportunity to pose questions and learn with others about your equipment and our club. Please feel free to drop in on any Thursday between 7:30 and 9:00 and join in. We want to express our thanks to the volunteers who are helping others in this way. They are: Tom DeMay, Dave Harvey, George Kinal, Paul Koskos, Boris Levine, Ted Meyer, David Moses (member No. 500), Gabriel Roth, and Jeanne Sella. Some of these folks are members of many years while others are relative newcomers. It is just another indication of the spirit that has permeated our organization and helped it to help you.

RETURN OF GAMES SIG. It is good to know someone is reading this column! Ron Wartow responded to our challenge to get GAMESIG going again. He will begin a column about games of interest and is seeking your contributions and advice. Ron will be holding an organizational meeting after the regular January meeting. If you are a gamer, novice or expert, please plan to attend and help shape the content of the SIG's activities.

TUESDAY EVENING TUTORIALS. We have added several new instructors to the Tuesday evening tutorial. We welcome the help of Tom Demay, Boris Levine and Mark Pankin. Ed Myerson and Chuck Seagrave have also expressed interest in tutoring. Bruce Field and the other regular instructors (Lee Raesly and Tom Warrick) are restructuring the first two evenings of the series. Look for a revised outline to appear soon. For those unfamiliar with the tutorials, there are four sessions which can be attended individually. Two of the sessions deal with use of the computer and the running of software. Two are dedicated to Applesoft Basic. There is a modest fee charged to attendees ($10.00 if you bring your computer and $15.00 if you don't). As of January, all new members will receive a coupon from the office good for one evening of the series as a way of promoting its benefit.

GARAGE SALE. Our semi-annual garage sale, held in honor of Apple's humble origins, will be the main activity at the December meeting. There will be a $1.00 entrance fee, charged to help us raise funds. If you have something to sell or trade, you will be allowed in to set up at 9:30. If you have nothing to swap, you must wait until the business meeting is completed which will be no later than 10:30.

IIc ADVERTISEMENT. Have you seen Apple's ad for the IIc. It has changed a bit from the first versions which showed the flat screen display. The flat screen is not available and will not be until 1985, and no longer appears in the ads. Apple drew some fire for pointing out how light the IIc was, some would say implying that the machine is portable. After their mention of its 8-pound weight, an asterisk refers you to the following "cutesy" expression, "Don't asterisks make you suspicious as all get out? Well, all this one means is that the IIc CPU alone weighs 7.5 pounds. The power pack, monitor, and extra disk drive, a printer and several bricks will make the IIc weigh more. Our lawyers were concerned that you might not be able to figure this one out for yourself." Unfortunately, anyone interested in using the computer as a portable needs to know how much the monitor and either power supply or power pack will weight, if they are to evaluate it. Knowing that the IIc alone weighs only 8 pounds is really of little benefit to anyone.

UPDATE ON MACWRITE WITH BUG. Our local dealer made available what he thought was the latest version of MacWrite, the one which permits disk based files. Apparently, it is not an official release, having no version number, only an August 1984 date. I assume that it is the version described in the recent MacWorld magazine article. Several problems have been found using it. Some reported difficulties with files over 24 pages in length. I have seen it drop out a MacPaint picture stored in the middle of text. When opening a header, the time appears in somewhat random locations around the bottom right of each page. Beware!

MINUTES

OCTOBER GENERAL MEETING

Washington Apple Pi, Ltd. met at USUHS on October 27, 1984 with Tom Warrick presiding. Rich Wasserstrom gave a report on Group Purchases. The Apple //c SIG will meet to organize after the main meeting. The Law SIG needs a coordinator - if interested please contact the office. The SIG officer elections will be held in October. Bernie Urban asked that members with graphic art expertise volunteer to help dress up the Journal. Members were asked to remember our status as guests at USUHS by restoring order to the facility after each meeting. Dana Schwartz reported that the new Membership Directory is ready for distribution. Members authorized by voice vote a $1.00 entrance fee for the December garage sale. Future meetings will present tax programs, accounting packages and ProDos.

The meeting was turned over to the main presentation, a panel discussion on graphics packages.

EVENT QUEUE

Washington Apple Pi meets on the 4th Saturday (usually) of each month at the Uniformed Services University of the Health Sciences (USUHS), Building B, 4301 Jones Bridge Road, Bethesda, MD, on the campus of the National Naval Medical Center. Library transactions, journal pickup, memberships, etc. are from 8:45 - 10:00 AM. From 9:00 to 10:00 AM there is an informal "Help" session in the auditorium. The main meeting starts promptly at 10:00, at which time all sales and services close so that volunteers can attend the meeting. A sign interpreter and reserved seating are provided for the hearing impaired.

Following are dates and topics for upcoming months:

December 15 - Garage Sale (** Third Saturday **)
January 26 - Home Accounting and Financial Packages
February 23 - Tax Preparation on Personal Computers

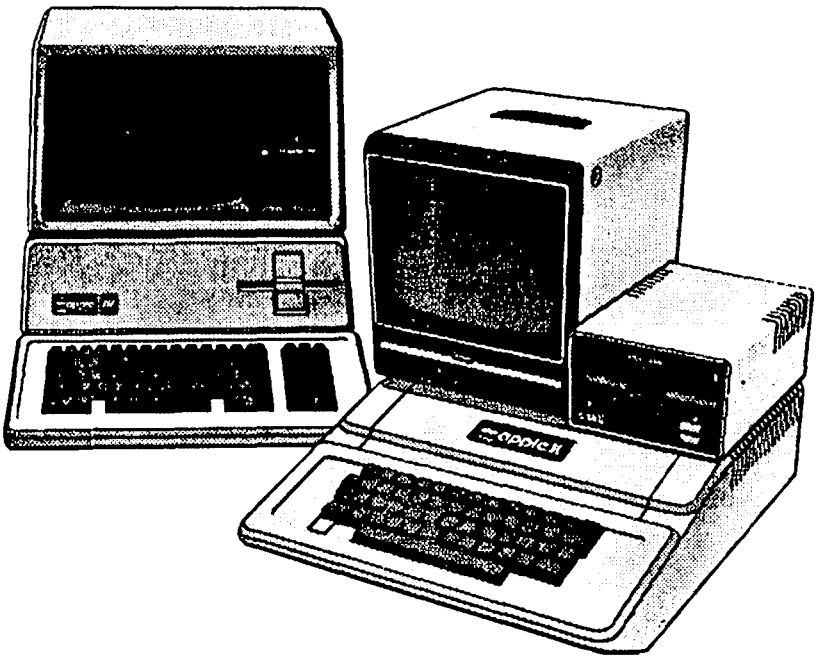
The Executive Board of Washington Apple Pi meets on the second Wednesday of each month at 7:30 PM at the office. All members are welcome to attend. (Sometimes an alternate date is selected. Call the office for any late changes.)

There's only one place to buy apples.

AT FREDERICK COMPUTER PRODUCTS !!!

As an Authorized Apple Dealer we offer a full selection of both Apple Hardware and Software at low, low prices.

Also, as an Authorized Service Center we provide fast and efficient service on all Apple products.



FREDERICK COMPUTER PRODUCTS, INC.

Microcomputer Systems And Peripherals

5726 INDUSTRY LANE-FREDERICK, MD.- (301)694-8884

apple computer
® Authorized Dealer

* December 1984 *

WAP

SUNDAY	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY
						1
2	3	4 EDSIG USUHS 7:30 PM	5	6 Sig Mac 7:30PM-Lady of Lourdes; DisabledSIG 7 PM CCCC	7	8 SigMac 9:00 AM USUHS; LISA SIG after SigMac
9	10 PI SIG 8:00 PM Office	11	12 Executive Board 7:30 PM Office	13 Stock- SIG 8:00 PM Office; Apple /// 7:30PM UC	14	15 WAP Meeting 9:00 AM USUHS Garage Sale
16	17	18	19 Happy Holidays	20 Pascal SIG 7:30 PM Office-ink pen plotrs.	21	22
23	24 WAP Office Closed	25 WAP Office Closed - Happy Holidays	26	27**NEWSIG Help** Every Thurs 7:30-9:00PM Office	28	29
30	31 New Year's Eve-Have a good one!					

* January 1985 *

WAP

SUNDAY	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY
		1 Best Wishes for a Happy New Year!	2	3 SigMac 7:30PM-Lady of Lourdes; DisabledSIG 7 PM CCCC	4	5
6	7	8 Beginning Tutorial #1 7:30 PM Office	9 Executive Board 7:30 PM Office	10 Stock- SIG 8:00 PM Office; Apple /// 7:30PM WR	11	12 SigMac 9:00 AM USUHS; LISA SIG after SigMac
13	14 PI SIG 8:00 PM Office	15 Beginning Tutorial #2 7:30 PM Office	16	17 Pascal SIG 7:30 PM Office	18	19
20	21	22 Beginning Tutorial #3 7:30 PM Office	23	24	25	26 WAP Meeting 9:00 AM USUHS
27	28	29 Beginning Tutorial #4 7:30 PM Office	30	31		

SIGNEWS

APPLE /// SIG meets on the second Thursday of the month at 7:30 PM. The next meeting will be on December 13 at Universal Computers, 1710 Fern Street, Alexandria, VA.

Apple IIc SIG has been organized. Watch for further details in the Journal or call the Chairperson listed in "Officers & Staff".

APPLESEEDS is the special interest group for our younger members. They meet during the regular WAP meeting.

DISABLEDSIG - See the DisabledSIG column elsewhere in this issue. Call Jay Thal for details.

EDSIG - the education special interest group - see the EDSIG page elsewhere in this issue.

FORTHSIG will hold its next meeting on Saturday, December 15 at 1:00 PM in the WAP office.

GAMESIG is being revived. See Ron Wartow's article elsewhere in this issue.

LISA SIG meets after the SigMac meeting on the second Saturday of the month. See LISA SIG news elsewhere in this issue.

LOGOSIG - watch for further details in a later issue of the Journal.

NEWSIG will meet just after the regular Washington Apple Pi meeting and conducts a "drop-in" for new Apple owners on Thursday evenings from 7:30-9:00 PM in the office. They will answer questions and try to help new owners get their systems up and running.

PIG, the Pascal Interest Group, meets on the third Thursday of each month at 7:30 PM at the Club Office. The round-table discussions for the next few months are:

December 20 - p-System, Apple][(+,c,e) computers and ink pen plotters.

January 17 - p-system, Apple][(+,c,e) computers and input digitizers.

PI SIG (formerly ASMSIG) meets on the second Monday of each month at 8:00 PM in the WAP office. See Pi SIG News elsewhere in this issue. For further details, call Ray Hobbs at 490-7484.

SigMac meets on the 1st Thursday of each month at 7:30 PM at Our Lady of Lourdes School, 7500 Pearl Street, Bethesda, MD; and on the 2nd Saturday from 9:00 AM to 12:30 PM at USUHS, in the auditorium.

STOCKSIG meetings are on the second Thursday at 8:00 PM at the WAP office. See the STOCKSIG News elsewhere in this issue.

Telecomm SIG usually meets after the regular WAP meeting. See the Telecomm SIG News elsewhere in this issue.

DECEMBER MEETING DATE

Please note that the December meeting is on the third Saturday, December 15. Also note that this means an early, early deadline for Journal articles.

EDSIG NEWS

by Peter Combes

EDSIG Calendar

Tuesday, December 4, 1984 at 7:30 p.m.

"The Apple runs educational machines."

All EDSIG meetings are held in the Auditorium, Building B, of the Uniformed Services University of the Health Sciences, on the campus of the National Naval Medical Center, 4301 Jones Bridge Road, Bethesda, MD.

NOTES FROM THE OFFICE

PROGRAM COORDINATOR VOLUNTEER

The person who volunteered at the September meeting to assist in coordinating WAP programs in Cara Cira's absence should contact Tom Warrick at 656-4389.

DOWNLOADING JOURNAL ARTICLES

The WAP office is now capable of downloading articles for the Journal to the //e. These can be transmitted from either the Macintosh, Lisa or Apple][series. Please call the office to arrange a mutually convenient time. We can also handle articles on 5 1/4" or 3 1/2" diskettes which are mailed or hand delivered to us. All diskettes will be returned or a blank diskette substituted.

DISKETERIA TO PHASE OUT DOS 3.2 DISKS

The Disketeria staff is sad to announce that it will cease selling WAP disks Volumes 1 through 40, the old DOS 3.2 series, effective January 1, 1985. This is necessary because there is no longer sufficient demand for these disks, and most of the programs on these disks have been improved, combined and reissued as Volumes 70 - 79 (DOS 3.3).

However, as a service to its members, WAP will keep an "archive" set of disks 1-40 in the office. Should you need a copy of one of these early disks, the office staff will copy it on a "demand" basis for a cost of $5.00.

DEALER'S CORNER

CHRISTMAS COMPUTER SALE

Computer Depot, Wheaton Plaza will be having a "HUGE" sale during December. Mr. Joel Krautheim has notified the Washington Apple Pi office that:

"All my inventory will be offered at discount to your members. I am an authorized dealer of IBM, APPLE, COMPAQ and PANASONIC computers. I also carry printers, modems and other peripherals, as well as all the popular software programs. Your members can get their sale price by coming to my store, seeing me, and showing proof of their membership to your organization."

For more information and prices, call Joel Krauthheim at 942-8270.

WAP HOTLINE

Have a problem? The following club members have agreed to help. PLEASE, keep in mind that the people listed are VOLUNTEERS. Respect all telephone restrictions, where listed, and no calls after 10:00 PM except where indicated. Users of the Hotline are reminded that calls regarding commercial software packages should be limited to those you have purchased. Please do not call about copied software for which you have no documentation. If the person called has a telephone answering machine, and your call is not returned, don't assume that he did not try to return your call - perhaps you were not home. Try again.

General	John Day	(301) 672-1721
	Dave Harvey	(703) 527-2704
	Robert Martin	(301) 498-6074
Accounting Packages		
Accountant(Dec.Sup.)	Mark Pankin	(703) 524-0937
Home Accountant	Leon Raesly *	(301) 460-0754
APPLE SSC	Bernie Benson	(301) 951-5294
Apple TechNotes	Lance Bell	(703) 550-9064
	Shirley Weaver	(301) 761-2479
AppleWorks	Carl Eisen	(703) 354-4837
	J.J. Finkelstein	(301) 652-9375
	Jay Jones (Balt.)	(301) 969-1990
Communications Packages and Modems-Telecom.		
Anchor Mark 12	George Kinal(7-10)	(202) 546-7270
	Jeremy Parker	(301) 229-2578
Apple CAT II	Ben Acton	(301) 428-3605
ASCII Express	Dave Harvey	(703) 527-2704
BIZCOMP Modem	Jeremy Parker	(301) 229-2578
General	Tom Nebiker	(216) 867-7463
Hayes Smartmodem	Bernie Benson	(301) 951-5294
Omninet	Tom Vier (1-6 PM)	(202) 887-7588
VISITERM	Steve Wildstrom	(301) 564-0039
XTALK CP/M Comm.	Bernie Benson	(301) 951-5294
Computers, Specific		
Apple //c	John Day	(301) 672-1721
	Scott Rullman	(301) 779-5714
Apple //e	Scott Rullman	(301) 779-5714
Lisa	John Day	(301) 672-1721
	Don Kornreich	(301) 292-9225
Macintosh	Jay Heller	(301) 948-7440
	Scott Rullman	(301) 779-5714
	Tom Warrick	(301) 656-4389
	Donald Schmitt	(717) 334-3265
Corvus Hard Disk	Tom Vier (1-6 PM)	(202) 887-7588
Data Bases		
dBase II	Paul Bublitz	(301) 261-4124
	John Staples	(703) 759-3461
DB Master	Dave Einhorn	(301) 593-8420
Data Perfect	Leon Raesly *	(301) 460-0754
Data Factory	Bob Schmidt	(301) 736-4698
General Manager	Normand Bernache	(301) 935-5617
	Leon Raesly *	(301) 460-0754
List Handler	Jon Vaupel	(301) 977-3054
PFS	Bill Etue	(703) 620-2103
	Ben Ryan	(301) 469-6457
	Jenny Spevak	(202) 362-3887
QuickFile II	J.J. Finkelstein	(301) 652-9375
Expediter Compiler	Peter Rosden	(301) 229-2288
Graphics	Bill Schultheis	(703) 538-4575
Languages (A=Applesoft, I=Integer, P=Pascal, M=Machine		
A	Peter Combes	(301) 251-6369
A,I	Jeff Dillon	(301) 422-6458
A	Richard Langston	(301) 258-9865
A	Mark Pankin	(703) 524-0937
A	Leon Raesly *	(301) 460-0754
A,I,P,M	Bill Schultheis	(703) 538-4575
A,I,M	Richard Untied	(703) 241-8678
A,I,M	John Love	(703) 569-2294
M	Raymond Hobbs	(301) 490-7484
P	Dottie Acton	(301) 428-3605
P	Donn Hoffman *	(412) 578-8905
Languages, contd.		
Forth	Bruce Field	(301) 340-7038
LOGO	Ron Murray (eve.)	(202) 328-3553
LISP	Fred Naef	(703) 471-1479
Math/ O.R. Applns.	Mark Pankin	(703) 524-0937
Monitor, RGB Color	John Day	(301) 672-1721
Operating Systems		
Apple DOS	Richard Langston	(301) 258-9865
	Richard Untied	(703) 241-8678
CP/M	Robert Fretwell	(703) 971-2621
	Ray Hobbs	(301) 490-7484
ProDOS	Richard Langston	(301) 258-9865
Paddles	Tom Riley (eve.)	(301) 340-9432
Printers		
General	Walt Francis	(202) 966-5742
	Leon Raesly *	(301) 460-0754
Apple Color Plotter	John Day	(301) 672-1721
Apple Daisy Wheel	John Day	(301) 672-1721
Apple Dot Matrix	Joan B. Dunham *	(301) 585-0989
Daisywriter 2000	Bill Etue	(703) 620-2103
	Henry Greene	(202) 363-1797
IDS 460	Jeff Stetekluh	(703) 521-4882
Imagewriter	John Day	(301) 672-1721
	Scott Rullman	(703) 779-5714
MX-80	Jeff Dillon	(301) 434-0405
NEC 8023	Bill Mark	(301) 779-8938
Okidata	Fred Feer	(703) 978-7724
	Scott Rullman	(301) 779-5714
Silentype	Bruce Field	(301) 340-7038
Spreadsheets	Leon Raesly *	(301) 460-0754
	Walt Francis	(202) 966-5742
Lotus 1-2-3	Walt Francis	(202) 966-5742
	Roy Rosfeld	(301) 340-7962
Multiplan	Terry Prudden	(301) 933-3065
VisiCalc	Walt Francis	(202) 966-5742
	Leon Raesly *	(301) 460-0754
Spreadsheet 2.0 (MagiCalc)	Leon Raesly *	(301) 460-0754
Statistical Packages	Jim Carpenter	(301) 371-5263
	Mark Pankin	(703) 524-0937
Stock Market	Robert Wood	(703) 893-9591
Tax Preparer-H.Soft	Leon Raesly *	(301) 460-0754
Time-Sharing	Dave Harvey	(703) 527-2704
Word Processors	Walt Francis	(202) 966-5742
Apple Writer II	Dianne Lorenz	(301) 530-7881
	Leon Raesly *	(301) 460-0754
Executive Secretary	Louis Biggie	(301) 967-3977
Format II	Henry Donahoe	(202) 298-9107
Gutenberg	Neil Muncy Can.	(416) 298-3964
Letter Perfect	Cara Cira	(301) 468-6118
	Leon Raesly *	(301) 460-0754
Magic Window and II	Joyce C. Little	(301) 321-2989
Peach Text	Carl Eisen	(703) 354-4837
PIE Writer/Apple PIE	Jim Graham	(703) 643-1848
ScreenWriter II	Peter Combes	(301) 251-6369
	E. E. Carter	(202) 363-2342
Supertext II	Peter Rosden	(301) 229-2288
Word Handler	Jon Vaupel	(301) 977-3054
	Christopher Romero	(703) 471-1949
Work Juggler //e	Carl Eisen	(703) 354-4837
Word Star	Christopher Romero	(703) 471-1949

*Calls up until midnight are ok.

HARDWARE HELPERS

If you are having hardware problems with your Apple and/or peripheral equipment, the following persons have agreed to help. It will be at the discretion of the Hardware Helper just how involved he becomes. He may only suggest things for you to do, or he may actively assist in cleaning, removing or replacing parts.

Tilghman Broaddus
Rt 1, Box 246
Mechanicsville, VA 23001
(804) 779-2553 (till 10)

Gene Cartier
6026 Haverhill Court
Springfield, VA 22152
(703) 569-8450 (till 10)

J.T. (Tom) DeMay Jr.
4524 Tuckerman Street
Riverdale, MD 20737
(301) 779-4632 (till 11)

Bruce Field
1402 Grandin Avenue
Rockville, MD 20851
(301) 340-7038 (till 10)

Lyman Hewins
Route 2, Box 26
Leonardtown, MD 20650
(301) 475-9563 (till 11)

Pete Jones
1121 N. Arlington Blvd.
N. Arlington, VA 22209
(703) 430-1606 (7-10)

Bob Kosciesza
2301 Douglas Court
Silver Spring, MD 20902
(301) 933-1896 (till 10)

Mark Pankin
1018 North Cleveland St.
Arlington, VA 22201
(703) 524-0937 (till 10)

Richard Rowell
1906 Valley Stream Drive
Rockville, MD 20851
(301) 770-5260 (7-11)
(202) 651-5816 (9-4)

Jim Taylor
16821 Briardale Road
Derwood, MD 20855
(301) 926-7869 (till 10)

Ron Waynant
13101 Claxton Drive
Laurel, MD 20708
(301) 776-7760 (7-10:30)

Dave Weikert
17700 Mill Creek Drive
Derwood, MD 20855
(301) 926-4461 (7-10 except Thurs. and weekends)

GENERAL INFORMATION

Apple user groups may reprint without prior permission any portion of the contents herein, provided proper author, title and publication credits are given.

Membership dues for Washington Apple Pi are $25.00 for the first year and 18.00 per year thereafter, beginning in the month joined. If you would like to join, please call the club office or write to the office address. A membership application will be mailed to you. Subscriptions to the Washington Apple Pi Journal are not available. The Journal is distributed as a benefit of membership.

Current Office hours are:

Monday - Friday - 10 AM to 2:30 PM
Tues. & Thurs. - 7 to 9:30 PM
Saturday - 9:30 AM to 12:30 PM (except meeting Sat)
- 12:00 to 3:30 PM (meeting Sat only)

JOB MART

PROGRAMMERS WANTED - We need individuals who know the Apple][family cold. Individuals will be responsible for modifying software to run on our system. Assembly language and BASIC a must. Please call software development on 883-1522.

CLASSIFIEDS

NEEDED: Apple][+ and/or //e to evaluate software designed for persons with communications handicaps. This project is funded under a U.S. Dept. of Education grant; however a loan or donation of at least one computer is needed. Donations are tax deductible. If you can help contact: Jim Gelatt, American Speech and Language Hearing Foundation. (301) 897-5700.

FOR SALE: 64K Apple][+ with Integer ROM card, Paymar Lower Case Adapter, shift key mod., game port extender. $485.00. Disk drives, controller card, 80 column card, Wildcard available at your option for additional dollars. Call Rich Wasserstrom at (H) 893-9147 or (O) 797-5860.

FOR SALE: Apple Writer //e with manuals, in excellent condition. Dick Gray, (703) 437-0197.

!!!! $1295 !!!!
PERCOM HARD DISK
for Apple II,IIe
Apple and CPM OS
10,20,30mb versions

PANASONIC PRINTERS
KXP 1090 (80 cps)
KXP 1091 (120 cps)
KXP 1092 (180 cps)

In Stock -
At Great Prices!!

MICRODRIVE II from TDS

FOR APPLE II
single $219.
w/ controller $259.
double $470.
w/ controller $509.

9 1/2 x 11 Clean Edge Paper
18 and 20# (2700/3000 sheets)
only 32.95 box.

MICRODRIVE IIC by TDS.

For APPLE IIC.
800% Faster.
100% Compatible.
140 kilobytes per disk.
Single unit only $252.

MAC 3.5" DISKS 4.25 ea.
MAC DISK BOXES 3.50 ea.
MAC DISK FLIP FILES 21.95 ea. (holds 50)
IMAGEWRITER Ribbon + EPSON Ribbons for MX,RX,FX 80 4.75 ea. reloads 1.95

ROBINS
MICRO SUPPLIES
GREAT VOLUME DISCOUNTS!

8304 Hilltop Road
Fairfax, Virginia 22031
(Merrifield Area)
UPS Shipping
FOB Fairfax, VA

VISA MasterCard
560-5900

MODIFICATION TO THE "MORSE KEY" FOR THE HANDICAPPED

by Boris Levine

This article describes a change to the Morse Code key to be used with the Serial Key program developed by Mr. Wolfger Schneider. The program was discussed in the December, 1983 issue of the WAP Journal, and one version of the key that I built appeared in my article in the same issue.

The program used the game port and Pushbuttons 0 and 1 to recognize the dots and dashes of the Morse code, as entered by the Morse key. The key, as originally constructed, worked well on the Apple][+ but had problems on the //e. It turned out that these were due to the fact that Apple had brought PB0 and PB1 to the keyboard and connected them to the 'open Apple' and 'closed Apple' keys. The result was that, as originally designed, the Morse key's wiring was not compatible with the //e.

To make the key work with both models, Mr.Schneider suggested the use of a 'pull-up' circuit. To do this I changed the connections on the 16 point DIP header to those shown in the sketch.

a. One end of each resistor (2.2 kOhms, 1/4 watt) was connected to terminal 8, 'ground' (changed from connection to terminal 1, +5 volts).

b. The switch was connected to terminal 1, +5 volts.

c. The legs of the switch were connected to terminals 2 and 3, PB0 and PB1, respectively.

The effect of this change is to increase the voltage at one or the other terminal, when the switch is closed. This is what those who know call a 'pull-up' circuit; when the switch is closed, it 'pulls-up' the voltage at the terminal to + 5 volts. (The circuit described in last year's issue had - would you guess? - a pull-down circuit and was not the way the //e was wired.)

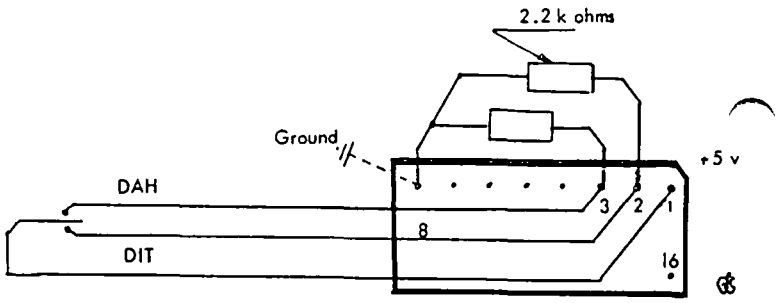
This electrical change required a corresponding change in the program. It repeatedly tests the two pushbuttons and the keyboard for signs of action, and goes into action when one of them is touched. Under the new connections,'the key instructions look like this:

```
 515 POKE 49168,0:REM RESET KEYBOARD
 750 IF PEEK(49249)>127 OR PEEK(49250)>127 OR
     PEEK(49152)>127 THEN GOTO 1010
 760 GOTO 750
1010 REM take some action
```

The addresses are those of PB0, PB1 and the keyboard. In the stand-by state, the PEEKed-at values are 127 or less: when one is energized the value exceeds 127 and the program takes over from there. (For your information, with the pull-down circuit, the energized values for PB0 and PB1 go to <128; however, the keyboard is not affected by the revised connection and the PEEK value remains >127.)

A fuller discussion can be found in such places as the APPLE II reference manual.

More on the use of the program will appear in another article, Watch for it.



NEW BOOKS FOR THE WAP LIBRARY

by Walton Francis

We now have over 200 books in the WAP library, covering every imaginable subject in computerdom. The table below shows the latest acquisitions, just added to the library. They range from the history of microcomputers in general (and Apple in particular) to in-depth advice on using your Macintosh. I particularly recommend:

- Fire in the Valley, a highly readable account of the emergence of microcomputers, with a focus on the bad business decisions that led to the demise of Apple's early competitors.
- Jerry Pournelle's book, The Users Guide to Small Computers, is mainly a (somewhat edited and annotated) compilation of his Byte columns. In spite of his anti-Apple prejudices, Pournelle is the best analyst/critic of computer languages and systems to wield a pen.
- The Whole Earth Software Catalog is not a compilation of all software, but of the best software. The reviews are not as thorough as some, but the judgments are excellent. I use a lot of applications software, and agree with these recommendations in almost every case.

Because the library is now so complete, we plan to concentrate in the future on selective acquisitions and multiple copies of popular books, so that loans can be initiated.

NEW BOOKS

Washington Apple Pi Library Book Listing — Nov 10, 1984

Subject	Title	Author	Year	No. Cop
Apple //	The Apple Almanac	Goez	1983	1
Apple //	InfoWorld's Essential Guide to the Apple	Hogan	1984	1
BASIC	The Apple Program Factory	Stewart	1984	1
DOS	Inside Apple's ProDOS	Campbell	1984	1
Data Base	Minute Manual for DB Master	Einhorn	1983	1
Data Base	Minute Manual for PFS	Lesho	1984	1
Directory	Whole Earth Software Catalog	Brand	1984	2
Gen Directory	OMNI Complete Catalog of Computer Software	Davies	1984	1
General	Fire in the Valley	Freiberger	1984	3
General	The Little Kingdom (Private Story of Apple)	Moritz	1984	1
General	The Users Guide to Small Computers	Pournelle	1984	2
Graphics	Applied Apple Graphics	Forer	1984	1
Hardware	The Epson Connection: Apple	Darnall	1984	1
Hardware	Hardware Interfacing with the Apple II Plus	Uffenbeck	1983	1
Mac/Lisa	Using MacWrite and MacPaint	Field	1984	2
Mac/Lisa	Presentation Graphics on the Apple Macintosh	Lambert	1984	1
Mac/Lisa	MacWork MacPlay	Poole	1984	2
Spreadsheet	Mastering Multiplan	Bolocan	1984	2
Spreadsheet	Multiplan Made Easy	Ettlin	1984	1
Spreadsheet	Managing Your Business with Multiplan	Witkin	1984	1

DISABLEDSIG NEWS
by Jay M. Thal

* *

DISABLEDSIG DECEMBER MEETING
Thursday, December 6, 1984, 7:00 P.M.
Chevy Chase Community Center
Connecticut Ave. & McKinley St.,NW, D.C.

SPEAKER: Dr. Lawrence Scadden

SUBJECT: Future Applications of Microcomputers for the Handicapped. Techniques which will enhance both educational and employment opportunities of not only the physically, but the cognitively, impaired.

* *

The November 1, meeting enjoyed a discussion of the symbiotic relationship between the handicapped and inmates of the Patuxent Institution at Jessup, MD. Patuxent now has over 10 microcomputers to assist its inmates in education and skills improvement.

The link between the inmates and the handicapped first started when MENSA began to help inmates learn reading skills through recitation into tape recorders. That project evolved into the inmates using their readings to transcribe books to tape for the blind. Later, the inmates became radio announcers - reading the daily newspapers over the air to be picked up by special radio receivers. From there it was only a step to repairing the receivers and tape recorders of the blind.

Now the inmates are repairing hearing aids and are learning to fit people for them. Members of the DISABLEDSIG and the Volunteers for Medical Engineering, led by John Staehlin, are now developing approaches which may lead to the inmates turning their newly acquired microcomputer technology, and mechanical skills, towards building adaptive devices for the handicapped.

This effort is made more poignant by the fact that many of the inmates may not have turned to crime but for their own learning disabilities providing limited opportunities. The National Council for Juvenile Court Judges estimates that somewhere between half and three-quarters of delinquent children are learning disabled.

* * * * *

FUTURE MEETINGS:

JANUARY 3, 1985 we expect to hear from Bud Rizer, of Maryland Rehabilitation. Mr. Rizer is a well known expert in the development of adaptive devices for the handicapped.

FEBRUARY 7, 1985 will be devoted to applications for the hearing impaired.

COMPUTER & ELECTRONIC
FURNITURE SALE
FOR CLUB MEMBERS ONLY
SAVE 20% OFF LIST PRICE
Just bring this ad or your club ID to qualify.

OUR SHOWROOM IS LOCATED AT
1042 ROCKVILLE PIKE ROCKVILLE, MD.

FUTURE FURNITURE
The Complete Electronics Furniture Center

Call 340-1106

VISA MasterCard CHOICE

Q & A

by Bruce F. Field

A couple of folks wrote in to correct one of my answers on adding a switch to your disk drive to allow writing on protected diskettes (diskettes that have the little write protect sticker over the notch). An additional switch should be wired in parallel with the existing switch. Don't disconnect either of the wires going to the existing microswitch in the drive, but bring the two wires out to an additional switch. When this extra switch is closed you will be able to write on any diskette whether it is write protected or not. I apologize for getting my original information from the manual rather than disassembling a drive and testing the switch.

Mark IV Designs, 2315 S. Canterbury, Lincoln, NE 68512, (402) 423-0363 sells a switch for the mechanically disinclined ($44.95) that requires no cutting or soldering.

Adding a switch to defeat the write protection will also permit you to write on the reverse side of a diskette (insert it in the drive upside down) without cutting an extra notch in the diskette. There has been considerable discussion on whether the reverse side of a disk should be used, so I report this only in the interest of completeness without including my opinion that you shouldn't use the back side of a diskette anyway. (Atta boy, Bruce - Ed.)

Murphy Sewall in Storrs, Connecticut wrote to report that there is a serious problem in using the //e 80-column card with a modem program.

"The problem, quite simply, is that the standard ready to receive communication code (X-ON or Ctrl-Q) causes the 80-column card to display 40 columns. The X-ON/X-OFF sequence is exchanged rather frequently in communication, especially with mainframe computers. Any communications program that is going to operate properly with the //e's 80-column card must either bypass the card's firmware or filter (discard) Ctrl-Q before it reaches the screen driver.

I bought my Apple //e in large part to use at 1200 baud with a University mainframe. I was extremely frustrated in May '83 when I discovered that none of the communications software at local dealers would function with the //e 80-column card. I later discovered that Softronic in Memphis had solved the problem in February 1983 by bypassing the card's firmware; other software vendors didn't get satisfactory software written until August 1983 or later."

As Murph points out, the moral of this story is to make sure your communications software takes care of this problem if you want to use 80-columns.

Terry Prudden called to say that Microsoft's Multiplan version 1.06 will not print properly with the Apple //c although it works fine with the //e. Version 1.07 has been released to correct this problem.

Q. We have both an Apple][+ and a //e. We use an NEC 8023 printer for all. We would like to add a mouse and Mousepaint to our system. However, I was told that Mousepaint only supports the Imagewriter. How can I get Mousepaint to dump to my printer? We have an SSM APIC parallel card and accompanying screen dump. Can you help?

A. It is true that Mousepaint was designed to work only with the Imagewriter. There is a way, albeit clumsy, around this problem. After you have finished creating your picture, save it to disk. Exit Mousepaint and BLOAD your picture back into memory with BLOAD YOUR PICTURE NAME.PIC,A$2000. Mousepaint automatically appends .PIC to the name you specify for your picture. Now type PR#1 (if your printer interface card is in slot 1) and type the command your card requires to dump Hi-res screen 1 to the printer. The Hi-res dump command can be found in the printer interface card manual, the SSM APPIC manual.

If you are somewhat adept at disassembling machine code it should be possible to modify Mousepaint to work directly with the NEC 8023 or the C. Itoh 8510 Prowriter. Pretty much the only difference between these printers and the Imagewriter is that the NEC and C. Itoh require ESC S before graphics characters and the Imagewriter requires ESC G. A friend of mine went through Mousepaint and found a table of escape sequences near the end. He changed the ESC G to an ESC S and found he could then print directly to the NEC printer from Mousepaint. He also noticed that after printing, the menus in Mousepaint were all messed up. This might be worth some further study.

Q. I have an Apple //c. How can I dump the Hi-res graphics screens to the new Apple Scribe printer?

A. The //c doesn't have software built-in to dump the Hi-res screens so you will have to purchase a program to do this. I believe the latest version of the Printographer from Roger Wagner Software (aka Southwestern Data Systems) supports the Scribe and //c. Other programs will probably be available by the time you read this.

Q. With Visicalc is there a way of replicating a column of numbers without replicating the formulas that produce these numbers? I want to be able to change the original column of numbers without affecting the copy.

A. One way is to save the column of numbers as a DIF file on disk, and then load it back in at a different column. The DIF file saves only the numbers and not the underlying formulas. If some Visicalc whizzes have a better solution I'd be interested in hearing about it.

Q. Plugging some peripherals into the game port in a //e seems to mess up some Apple operations in a different way than the][+. For example, Ctrl-RESET causes a re-boot of the system and messes up AppleWorks. What are the differences between the game ports of the][+ and //e that cause this?

A. The difference is that two additional keys on the //e keyboard, the OPEN-APPLE and SOLID-APPLE keys are connected to pushbutton 0 and 1 inputs on the game I/O port. They are do not send a code to the Apple when they are pressed as the other keys do. When Ctrl-RESET is pressed the Apple checks to see if either of these two keys are pressed. If the

contd.

Checkmate Technology presents 3 new peripheral cards for the Apple

The MultiView 80/160 Card™– the intelligent video card for the II, II+, //e, or Franklin. Perfect for Spreadsheets and Word Processing.

7 Screen Sizes Choose from among these highly readable screen sizes with a simple command 80x24 80x32 80x48 96x24 132x24 132x30 160x24. Use with any monitor!!!*

Wide Angle Screens Stretch the screen with one command. An 80-column screen becomes as easy to read as a 40-column screen and a 132 column screen becomes as easy to read as a normal 80-column display. We call it the "Eye-Saver".

12 Attribute Sets Choose any one of 12 attribute sets for a screen display. Bold, underlined, inverse, and normal characters combine with normal and inverse backgrounds to provide 12 exciting new ways to view the Apple screen.

//e Compatible Works in slot 3 of //e. And if an extended 80-column card is in the auxiliary slot, use MultiView to program the auxiliary 64K memory with simple BASIC commands.

Software Support Compatible with all languages and operating systems used on the Apple.

Programmer's Aids Bring back into view your program or catalog listing that scrolled off screen with our hardware REVERSE SCROLLING command . . . or add a 25th, 26th, or 27th line to the screen with the PROMPT LINE command!

Money Back Guarantee 10 day money back policy if not delighted with MultiView.

Reg. Price $349.95
SPECIAL CLUB PRICE: $225.00

*12MHz or better

Applewriter //e Preboot™

Word processing in any of MultiView's sizes and attributes. See almost a full page on screen with an 80x48 screen size! Applewriter II and ProDOS version preboots available January 1985.

Reg. Price $19.00
SPECIAL CLUB PRICE $12.00

InfoBank][™

Simplified "user friendly" data base that even a child can do! 132 column format calendar functions and auto dialing with your modem. Special 240 column field for detailed comments. **Reg. Price $89.95**

SPECIAL CLUB PRICE $59.00

//e 80RAM CARD™

64K of memory + 80 columns for your //e!! Features double hi-resolution graphics!! Can be used with MultiView with no screen flicker on any monitor!!

Reg. Price $129.95
SPECIAL CLUB PRICE $85.00

Superworks PROM ™

20 Function keys, a program line-editor complete with auto-numbering, and a screen dump print command make your Apple a programmer's dream!

Reg. Price $39.95
SPECIAL CLUB PRICE: $25.00

Language Factory Character PROMS ™

See Engineering/Math or Foreign Language or Graphic characters on the same screen with standard ASCII characters using the appropriate Language Factory PROM™

Reg. Price $44.95
SPECIAL CLUB PRICE $30.00

Universal Spreadsheet Preeboot™

See any MultiView screen size with VisiCalc, Multiplan, Magicalc! THE Spreadsheet, Acecalc, and IACALC. Use any attributes including underlining and bold face with your spreadsheet. Designed for use with MultiView only!

Reg. Price $49.95
SPECIAL CLUB PRICE $29.00

Zee 80A Card™

For II, II+, //e, and Franklin. Needed for CP/M programs like Wordstar, dBase II, Multiplan and other business programs. Software not included.

Reg. Price $89.95
SPECIAL CLUB PRICE $59.00

All products made in the U.S.A. featuring a 5 YEAR WARRANTY.

Manufactured by

Checkmate Technology, Inc.
509 South Rockford Drive - Tempe, Arizona 85281

™

Acecalc is the trademark of Artsci Inc.
Apple is the trademark of Apple Computer Inc.
AppleWorks and AppleWriter are the trademarks of Apple Computer Inc.
CP/M is the trademark of Digital Research.
Franklin is the trademark of Franklin Computer Inc.
IACcalc is the trademark of the International Apple Corp.
Magicalc is the trademark of Artsci Inc.
Multiplan is the trademark of Microsoft Corp.
THE Spreadsheet is the trademark of CALL A.P.P.L.E.
VisiCalc is the trademark of Software Arts.

CALL the Club office at 654-8060 to arrange purchases or for more information.

OPEN-APPLE key is pressed at the same time then the Apple first destroys any program or data in memory by writing two bytes of arbitrary data into each page of the memory. Then a normal cold-start reboot is performed (as if you had turned the computer off and then on again). If the SOLID-APPLE is pressed during RESET the Apple runs a self-test program to check memory.

The game I/O pushbuttons are wired in more or less in parallel with these keys, so if you have a peripheral where the buttons are held down, whenever you want to do a Ctrl-RESET you will get either a self-test or a cold-start reboot. Regular Apple paddles can be connected and won't interfere with RESET as long as the buttons aren't pressed, but don't let your paddles get buried under a lot of books or you could accidentally have a button pressed and do a reboot when you try to RESET.

Q. I am using Kraft paddles with my Apple][+. The readout from both paddles is unstable and they interact with each other. Without rotating the knob a paddle may read 150 one second then 148 the next and 151 the next. When both paddles are called from a program the readouts are not only unstable but also interact with each other so rotating one control not only affects its readout but also the readout of the other paddle.

I have used the paddles with many programs and my own which contain lines which read as follows:

```
10 PRINT PDL(0)
20 FOR I=0 TO 10:NEXT I
30 PRINT PDL(1)
40 GOTO 10
```

Would you tell me how to get the paddles to operate properly?

A. There may be a couple of problems here. Any time you take a continuously variable device (a paddle) and convert it's output to a digital number there is an uncertainty of plus or minus one in the last digit. The paddle may be set exactly halfway in between the two numbers and noise in the system will cause it to flip back and forth between them. You can expect the numbers to jump around some but the question is how much, and I think that 3 steps in a short period of time is too much. It could be that your paddles simply are getting old and noisy. If you can borrow some paddles from a friend, try them and see if they also jump around. Alternatively you could have a problem with the paddle timer chip in your Apple. This is a relatively inexpensive chip that is available from most electronics supply houses. What you want is a 558 Quad timer chip, and it is located at the rear of the motherboard on the right hand side (location H13).

The second problem, of paddle interaction, may be due to defective paddles or a timer chip, but may also be a software problem. When you use the PDL function of Applesoft the Apple sends a strobe to the timer chip which starts it timing ALL the paddle inputs. Four outputs (one for each of the possible paddles that can be connected) are turned on and read by the Apple. After a length of time that depends exactly on the setting of the paddle knob the signal for that knob turns off. The Apple counts the length of time that signal is on for the paddle you are trying to measure. If the first paddle we read returns a low number the time is very short and the signals from the other paddles may still be turned on. If we then try and read one of the other paddles we will get a completely wrong number. You need to wait a short period of time to let all the paddles "time out" before reading a second one. Three or four milliseconds should be enough time. The FOR - NEXT loop in your program above takes about 17 milliseconds to execute and should provide more than enough time. Thus I think you should look at your hardware, either the paddles or the timer chip.

Q. I have trouble in using my Apple][+ with the UltraTerm 80-column card. I purchased Magic Office from Artsci because it said in the ad that it would run on an Apple][with an 80-column card. When this software arrived, I tried it on my Apple][+ and it did not work. If I take the UltraTerm out of my machine Magic Office shows something strange on the screen. I checked Magic Office with my friend's Apple //e with Apple's 80-column card and everything works okay. Do you know of any way to overcome this problem?

A. This type of problem is very common and should serve as a warning to Apple][and][+ owners considering buying an 80-column card. The 80-column cards available for the][and][+ work differently from each other and differently from the //e 80-column card. It is important to make sure that any software you buy is compatible with your particular 80-column card, not just any 80-column card.

Having said that (which doesn't help our questioner who lives in Taiwan) there may be two solutions to the problem. The manufacturer of the 80-column card (Videx in this case) may have written a pogram to modify your Magic Office software to work with the Ultraterm. These kinds of programs are usually called "preboot" because you boot up on them, and then run the application program. A second possible solution is to contact the manufacturer of Magic Office (Artsci, Inc., 5547 Satsuma Ave., North Hollywood, CA 91601) and see if they have a version of Magic Office that will run with your Ultraterm. Finally if any readers have another solution and let me know, I'll pass it along.

Q. I have an Apple][+, Epson MX-80 printer, and Apple Writer II word processor. I'd like to include in my glossary a definition which will turn on enlarged print that will not automatically turn off at the end of each line. So far I have not been able to do this. Can you help?

A. You don't say, but I suspect that you have a Pkaso printer interface card. The Pkaso card intercepts the printer commands and turns off the enhanced features at the end of each line. To get around this you need to use the special Pkaso commands instead of the commands listed in your printer manual.

BEST OF WAP BBS

by =Alexander-

Ed. Note: This is our first attempt to capture some of the exchange of timely information and tips which appears daily on our Apple bulletin board system. Interestingly enough, some of our long time members and others newly on the scene who seem reluctant to contribute articles to the Journal do so frequently in this mode. Apparently, spontaneous short, punchy responses to calls for help are easier and more fun to generate than are lengthy articles. We lose some of the flavor of the BBS by correcting obvious typos and setting everything to upper and lower case - our format will change over time. The Editor's comments below are those of Alexander Barnes (WAP#3041) who wishes to be called by his handle, =Alexander-.

{In case you want to know how this all started.-Ed.}

From WAP538 to:[YOU] Date=09/19 Subject: BEST OF ABBS EDITOR

Thanks for your offer to be the "BEST OF THE ABBS" Editor. Do you have ScreenWriter or Apple Writer II or //e? That will make Gena Urban's job much easier. Also, I presume you do have some kind of terminal software such as AE Pro, Data Capture or Remember][to save messages as text on an Apple 5 1/4" disk? Tom Warrick

From WP4795 to:WAP538 Date=09/22 Subject: Apple's Own BBS

According to a letter from Apple (Charlotte, NC) Apple has a BBS for questions regarding Apple products. Operational 24 hours a day and at 300 and 1200 baud. Number is 704-523-7630. Tom, please add to the BBS listing. Letter is dated Sep 14, 84, in reply to a letter with Apple and non-Apple HW problem I had.

From WP4795 to:WAP538 Date=10/05 Subject: Apple BBS

Tom: The ltr says "For those who have modems, we have established a BBS which may answer many of your questions and is oper 24 hrs per day. Currently, the BB can be reached at (704)-523-7630 for 300 & 1200 baud." Except for abbrev that I used that is what they said. Seems, at least for NC Apple, they have made a change in policy. Hope that it will help. Bob Wood

From WAP208 to:WP4772 Date=09/26 Subject: Mouse & 80 cols.

The reason you probably haven't seen a Mouse version of AppleWorks is the apparent conflict between the extended 80-col card in the //e and the Mouse. I haven't seen anyone who has been a able to use both at the same time. Theron Fuller

From WAP884 to ALL Date=09/26 Subject: Mouse/80 columns

There is a new version of Bank Street Writer that has been released. It uses the eighty column memory, and uses the double Hires to produce 80 columns. It also uses the mouse. So much for the no mouse/80 columns theory. Andrew

From WP3274 to:WAP208 Date=09/27 Subject: Mouse & 80-Col

Dollars & Sense uses both 80-columns & the AppleMouse with no problems. I've never heard of any incompatibility problem between the two. BUT there is a serious incompatibility problem between the Mouse card and nearly every Z-80 or CP/M card on the market. Apple concedes this, but accuses the Z-80 makers of failing to follow Apple specs years ago. -- Walt Mossberg

From WAP208 to ALL Date=10/05 Subject: Mouse & 80 Col Card

You can get 80 columns with the Apple Mouse by using the //e-//c double hi-res graphics. But, when you do this, you use the extra 64K on the 80 col. card. Therefore, a program like AppleWorks which uses the extra memory for a 'desk top' can't have 80 cols. Anybody who can get the Apple Mouse and the //e 80 col. card to work simultaneously please let me know how you do it.

From WAP243 to:WAP208 Date=10/07 Subject: (r)mouse /80cols

It may be a problem from a Basic program, but there are certainly ways around it in machine language. (For one, you can ignore the ROM routine for BASIC control of the Mouse). There certainly would be no trouble on a //e. Davis Lee

From WAP208 to:WAP243 Date=10/07 Subject: Mouse & 80 Col. Card

Using Apple's documentation, I have not been able to initialize the Apple Mouse when the Apple 80 column card is in control of the screen. The Apple just goes away and contemplates its navel, or something. Nor have I seen any program which uses the Mouse & firmware 80 cols. All the programs I have seen use hi-res letters. And graphics 80 cols. is s-l-o-w for things like word processing. Just look at Screenwriter in 70 col mode. I would greatly appreciate someone telling me how to use the Mouse & Apple 80 col. board at the same time. I have some neat stuff I would like to do in 80 cols. Theron Fuller

{Even with brevity, there can be too much of a good thing.-Ed.}

From WP5390 to ALL Date=09/26 Subject: APPLE //c Experience

I've had the //c since 5/23/84 and have some gruesome anecdotes for those who need convincing never to buy a new machine 'til it/s at least 6 months old. Lv msg for WP5390 to hear the woe. Also, a recent conversation with Apple Marketing Manager in Atlanta indicates the flat panel display is not going to make it until early '85 and could be $600 to $900 retail list !!!

From [YOU] to ALL Date=10/06 Subject: //c or //e?

Why would one prefer a //c to a //e, or vice versa? =Alexander-

From WP4795 to:[YOU] Date=10/06 Subject: IIc vs //e

Without the flat screen for the IIc I sure can not see buying a IIc. Have a //e with a full board and can not see using a IIc like I do this //e. But I would consider a IIc when it has a flat screen (and I can afford it) in addition to the //e.

contd.

From WP4772 to:[YOU] Date=10/06 Subject: Apple //e or IIc?

I know the problem well. I chose the //e for its expandability. It seems that it will never be obsolete. You can always add a card to do whatever you wish to do. At this point the IIc does not have that capability. (That meand no CP/M board for the IIc at this posint, as an example.)

From WAP261 to [YOU] Date=10/07 Subject: //e vs IIc

The //e offers greater flexibility in adding peripherals. It has 7 empty slot which hardware designers can dream up any application they choose to put into these beauties. The IIc offers a solid set of features, built in at no extra charge. It will also offer portability once a flat panel monitor and portable power supply become available. Walt.

From WP2243 to:[YOU] Date=10/08 Subject: To c, or not to c

That is the question! //c would be good for anyone who wanted something simple, with limited capabilities, that they could carry with them. But, if you want -any- expansion possiblities at all, and you do not -need- to carry the machine with you, the //e is the better bet. If I were -given- a //c, I would sell it and buy a //e. Brett

From WAP438 to ALL Date=10/11 Subject: //c

I got my //c the first week because I needed a portable. What other machine can run a 128K version of Multiplan and fits in a small bag for $1300, (plus $100 for a second monitor to keep at work)...Although I think Apple exaggerated its compatability & comparability to a standard //e it is a nifty machine. A lower profile makes it easier to type on, and the new version of Bank Street Writer w/mouse is fun. If anyone needs more than 2 disks they can get the Quark 10MEG. The only reason to get a //e is for coprocessors, A/D, and other relatively unusual applications. I have yet to run into modem problems as I use my old II+ for comm.

From WP2645 to ALL Date=10/13 Subject: //c + modems

I just read a magazine article about the //c and modems. The //c's clock is a little off so every once in a while a character will be missed on 300 baud but on 1200 baud it's just terrible. Kitt Diebold

From WAP538 to:WAP805 Date=10/06 Subject: Old Apples

The ABBS Apple is serial number 6401 or close to it. Yours (3026) is the oldest I've known. John Moon purchased what is now the ABBS Apple in 1979. My personal Apple is serial number 13xxx, purchased in January 1979. How about it, folks? Any other venerable Apples out there? This would make a great story for the Journal! Tom Warrick

From WP4010 to ALL Date=10/08 Subject: Elderly Apples

I bought an Apple][+ in Aug 79 and it's #1999. Only thing ever to go wrong is two keys quit making contact within a month. Since then....nothing. Jed

From WAP187 to:WAP538 Date=10/09 Subject: Old Apples

Tom: My Apple is Serial No. 1437 - Have since replaced the motherboard and power supply.

From WP1755 to ALL Date=10/09 Subject: Our SYSOP

Our SYSOP, Tom Warrick, is author of a nice article on "Legal Aspects of Purchasing Microcomputer Software" in the August issue of "The Bulletin of the American Society for Information Science". Nice job, Tom!

{With this inclusion I get to keep this job for another month.-Ed.}

From WP5146 to ALL Date=10/16 Subject: Apple][+ Repair

To everyone who responded to my pleas for help with color TV and disk drive incompatability - Thanks!!! The solution turned out to be proximity of drives and TV. I moved the TV to the other side of the table and all now works super. Thanks for the great response. Lary.

From WAP243 to ALL Date=10/17 Subject: Apple 1984 sequel

"1984", voted best commercial of 1984 by Ad Week magazine, now has a sequel. Handled by Chiat/Day, the commercial has already been filmed in Sept. in London. Directed by Tony Scott (brother of Ridley). It doesn't focus on any product, but is surreal (like "1984"). Primary vehicle ... The Super Bowl. Cost? Maybe 10,000 Macs. Ad Age 10/1/84 - Davis Lee

From WP1095 to:WP5152 Date=10/17 Subject: Modem prices

Modem prices are dropping FAST. I understand there is now a Promodem for the Apple][family (in the slot type) offered via Call A.P.P.L.E., for $299 (300/1200 baud auto-everything). The Mark 12 is available mail order for $230 - check the Computer Shopper. Expect lots more price erosion in time for holiday buying.

From WP3274 to ALL Date=10/24 Subject: 16-bits for Apple

According to new INFOWORLD, two firms will release in 2-3 months add-on hardware to turn Apple //e into 16-bit computer able to address DIRECTLY up to 16 MG of internal RAM (vs 64K now, and 700K on IBM PC). Secret is long-rumored new chip, the 65816, which can emulate the old 8-bit 6502 chip we all have, and thus run ALL existing Apple][software, but is also a true 16-bit chip (not an 8/16 as in IBM PC) that is far faster and can handle much bigger programs. One company plans two new cards, each with the 65816 and some extra ROM and RAM. One, with added 64K, will cost $350. Other, with an extra 256K, will be $795. Article says that these cards will make direct use of any RAM you already have in addition to extra RAM cards contain. This implies that, if you had 128K already and got new 65816 card with another 256K your computer would run faster and would be able to directly (no bank-switches) use 384K. A second company will sell not a card but a modified 65816, called a 65802, which just plugs into your //e motherboard in place of 6502. It will still directly read just 64K of RAM, because it's a 16/8 bit chip. But it will run faster. A third firm, not mentioned in INFOWORLD but discussed on CompuServe, promises its own 65816 card in late '84 for $300-$500 that can run MicroSoft WINDOWS, a la IBM PC. And Apple is rumored to be using the new 16-bit 65816 in an all-new Apple // model called the "//X". APPLE][FOREVER! - Walt Mossberg.

APPLES ABROAD

by John F. Day

Having recently returned from almost five years in Europe, I wanted to pass on my Apple experiences to the Washington Apple Pi members.

Perhaps the biggest area of mystery confronting the computer user bound for a foreign country is the subject of AC voltage and hertz (cycles). Most of the world uses 220 volts, 50 hertz as standard wall outlet power, while the US has standardized at 110 volts, 60 hertz. If you are the owner of a 110 volt system, as most of you are, you will need to understand how to work around this difference before you can get back on-line in most foreign countries.

Some of the earliest Apples came with a dual voltage power supply that allowed the owner to just flip a switch to change between 110 and 220 volts. This feature, however, went away with the increased FCC requirements for shielding against radio frequency interference (RFI), leaving most Apple owners with a 110 volt computer and a requirement to be able to run it on 220 volts if they are to compute overseas.

There are several ways to overcome this problem. You can buy a 220 volt power supply and solve the whole problem. The price for one from Apple is about $150, although several independent companies sell "switchable" replacement power supply models that provide both voltages, and as a bonus, are usually cheaper. If you have the money, this is a permanent solution for your computer, and is probably the best answer. It does not, however, do anything for your monitor, printer, or any other peripheral needing wall power. I made several phone calls to peripheral manufacturers, and found that all had 220 volt power supplies, but that each additional supply was another $100-150. Altogether, a very good, but very expensive way to solve the voltage problem.

For those who don't have that kind of money, there is another way that works just as well, and requires no changes to your computer or any of its associated peripherals. My recommended solution involves purchasing a step-down transformer of sufficient wattage to power all your equipment. Such transformers reduce the 220 volts at the wall socket to 110 volts for use by your system. You merely plug the transformer into the wall, plug your system into the transformer, and you're in business!

Step-down transformers are available from most large electronics parts houses. The transformer that you need is a box about the size of a disk drive, and can weigh 10-15 pounds. The box contains a wire wound transformer, a cord to plug into the wall, and one or several sockets into which you plug your hardware. I do not recommend the smaller kind of transformers, such as the ones normally sold in most Radio Shack stores, because they are not made for sustained use.

To calculate the wattage needed, add the watts used by each piece of equipment together and buy a transformer with a rating equal to, or greater than the wattage of all your gear. Don't forget to add the power needed by that piece you plan to buy next Christmas. A transformer of 750 watts is sufficient for most people.

With the voltage situation solved, we move to the 50 versus 60 hertz problem. What I am about to say will no doubt raise the eyebrows of some technicians who will read this, but is based on over three years experience by not only myself, but many other users I knew in Europe. My recommendation is to ignore the hertz issue completely. Most of the power supplies made today are capable of being run at either 50 or 60 hertz without problems. Your 60 hertz power supply will operate a little bit hotter at 50 hertz, but otherwise normally. A fan such as a "System Saver" will take care of any extra heat. Even the power supplies that are designed to work properly only at 60 hertz have been no problem.

Some specific pieces of 110 volt, 60 hertz gear I have seen in extended use without problem are the Apple][, Apple][+, Apple //e, Macintosh, LISA 2/5, Franklin Ace, the line of Epson printers, Brother printers, NEC printers, Apple monitors, Amdek monitors, Taxan monitors, and virtually every accessory card know to man. In fact I don't remember a single piece of equipment that didn't operate properly, save several very cheap video monitors that worked, but suffered from a flickering picture.

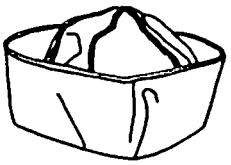
The reverse situation is also not a problem. My own Apple][+ is a "Europlus" made in Ireland, and along with my Apple Monitor ///, Taxan RGB Vision 1, and Apple Daisywheel printer, are all still 220 volts, 50 hertz. To operate them here in the US, I made an extension cord that plugs into my 220 volt wall air-conditioner outlet and have been using them on a daily basis for four months without any problem.

Several final tips are in order. Don't expect to find any bargain prices in Europe. The market is still predominately business and professional, and the prices are actually higher there than retail prices here because of import duties and shipping costs. In the Far East, there are many good bargains, but make sure they are genuine brand name products, and not a cheap pirated clone. If you do buy a clone of some kind, be prepared to have it seized by US Customs when you return to the U.S. If you buy software, be sure that the documentation is in English, and make sure the English is American English. I knew of several people who ordered and paid for software only to find out that it was set for British pounds rather then American dollars. Let the buyer beware!!! At the time I left Germany, Apple Germany refused to carry any 110 volt products with the exception of Apple//e and the Macintosh, which interestingly enough was only available in 110 volts. This meant that if you wanted 110 volt equipment you had a problem. Documentation was also a big problem, since it was easy to get in German, but almost impossible to get in English. Additionally you should know that standard European printer paper is 12" long, not 11". You will need to take this into consideration when buying paper and formating documents for printing.

Don't hesitate to take your Apple with you "over there." Follow these simple recommendations and you will have fewer problems making the adjustment. If you have specific questions, feel free to call me at (301) 672-1721.

OH . . . BY THE WAY

by John A. Love III



Between vacation, getting distracted by dissecting Applesoft and getting further distracted by learning more about DOS it seems that three months have slipped by. At the risk of driving our illustrious editor "bonkers", I've submitted five articles this time.

This first one isn't particularly spectacular - at least the portion of the program I wrote isn't. Simply speaking, it is a utility to go back and forth between decimal, binary and hexadecimal numbers. In the context of the user, "binary" is really "pseudo-binary" - a string of up-to-16 zeros and ones. My primary References include:

a) "All About Applesoft", CALL A.P.P.L.E. In Depth #1. I submit this as an absolute must for Assembly Language programmers, together with these other References.

b) "Disassembly Lines" by Sandy Mossberg, a series of extremely informative articles that has been continuing in Nibble.

c) The profusely commented disassembly of Applesoft ($D000 - $F7FF) written by Glen Bredon, the author of "Merlin" / "BIG MAC". This disassembly can be accessed via these two programs.

After entering my Assembly code, the user can implement the desired conversion by BRUNning the resultant Machine code, followed by one of the three "recipes" described in the top-most set of comments. As it is, the existing code "robs" the top 49 bytes of the Input Buffer ($0200 - $02FF).

Sandy Mossberg presents an alternative approach to switching back and forth between DECIMAL and HEX numbers. Instead of addressing Applesoft ROM as Val Golding does, Sandy addresses DOS directly via its routine at $A1B9 which is "called when the starting location (A) and length (L) of a binary program must be evaluated". Coming out of this routine will be the low order byte of the converted # in the X-register and the high order byte in the A-register. Furthermore, the first character in the Input Buffer (after the "&") will be the "heads-up" dollar sign for HEX input, percent sign for "BINARY" input or a # for DECIMAL input. To make a long story short, using Sandy's approach, I can save 12 bytes. Therefore, Line #64 could read ORG $02DB without having to defile $03D0 at the very end.

Just one more "tid bit" from Sandy Mossberg before I present the Assembly code. Look at Line #111. Why does it enable a skip over the following two bytes? The reason is that the 6502 CPU sees "2C A9 B1" as the next three bytes following Line #110. These bytes are interpreted as "BIT $B1A9" which affects or effects nothing in my code. As a direct result, the instruction labelled "ONE" is skipped. Neat, huh!!! Not only is it neat, but it saves one byte. Face it, "neat and useful" is an unbeatable combination.

Now, onto the main event -- ENJOY!!!

Assembly Code:

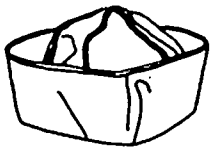
```
1               EXP  OFF
2     *  Notes: "EXPand OFF" for all MACROs via Line #1.
3     *         All code is Relocatable!!!
4
5
6     ************************************
7     *                                  *
8     *  DECIMAL--HEX--BINARY CONVERTER  *
9     *                                  *
10    *   CALL A.P.P.L.E. In Depth # 1   *
11    *               &                  *
12    *         John A. Love, III        *
13    *                                  *
14    ************************************
15    *                                  *
16    *    To convert HEX to DECIMAL:    *
17    *    enter " &$ " + HEX number.    *
18    *    (    e.g., &$FF = 255    )    *
19    *    ------------------------      *
20    *    This also gives 'BINARY':     *
21    *    (&$FF = %0000000011111111)    *
22    *                                  *
23    *    To convert DECIMAL to HEX:    *
24    *    enter " & " + DECIMAL nbr.    *
25    *    (   e.g., &100 = $0064   )    *
26    *    ------------------------      *
27    *    This also gives 'BINARY':     *
28    *    (&100 = %0000000001100100)    *
29    *                                  *
30    *    To convert 'BINARY'to HEX:    *
31    *    enter " % " +'BINARY' nbr.    *
32    *    ( e.g., &%101010 = $002A )    *
33    *    ------------------------      *
34    *    This also gives  DECIMAL :    *
35    *    (     &%101010 = !42     )    *
36    *                                  *
37    ************************************
38
39
40  *          EQUates follow:
41
42
43    JuMP      =    $4C       ; Machine code for "JMP".
44    AMPER     =    $3F5      ; Ampersand Vector.
45    DIGITS    =    $06       ; 16-bit registers used
46    SHIFT     =    $08       ;   by BINARY converter.
47    LENless8  =    $1F       ; Used by BINARY converter.
48    A2L       =    $3E       ; General purpose
49    LINNUM    =    $50       ;   16-bit registers.
50    CHRGET    =    $00B1     ; ASCII "GET" (Low byte).
51    INBUFF    =    $0200     ; Input Buffer.
52    FRMEVL    =    $DD7B
53    GETADR    =    $E752
54    LINPRT    =    $ED24
55    PRNTAX    =    $F941
56    GETNUM    =    $FFA7
57    ZMODE     =    $FFC7     ; Prepares "GETNUM" routine.
58    COUT      =    $FDED     ; Output to Monitor.
59    CROUT     =    $FD8E     ; Output <CR>.
60    SYNTAX    =    $DEC9     ; Executes "Syntax Error".
61
62
63    *-------------------------------------------------
64              ORG  $02CF     ; Execution Address uses top 49
65    *                        ;   bytes of the Input Buffer.
66    *-------------------------------------------------
67
68
69    SETUP     LDA #JuMP      ; SET UP Ampersand Vector
70              STA AMPER      ;   via "BRUNning" this code.
71              LDA #<START
72              STA AMPER+1
73              LDA #>START
74              STA AMPER+2
75              RTS
```

contd.

```
76
77
78  *  Look at incoming character following " & ".
79  *  NOTE:  CMP is with Positive ASCII of character.
80
81
82  START    CMP #'$'        ; If dollar sign, then
83           BEQ HEXin       ;   HEX input follows.
84           CMP #'%'        ; If percentage sign, then
85           BEQ BINin       ;   'BINARY' input follows.
86
87
88  *-------------------------------------------------
89  *  DECIMAL -to- HEX conversion ( by : Val Golding ).
90  *  Note:  "FAC" = Floating point ACcumulator.
91  *-------------------------------------------------
92
93
94  DECin    JSR FRMEVL      ; Evaluate input & put in FAC.
95           JSR GETADR      ; Get FAC & convert to integer.
96           LDA #"$"        ; Put into "LINNUM" in order
97           JSR COUT        ;   to print it.  "PRNTAX"
98           LDX LINNUM      ;   prints A, then X, via
99           LDA LINNUM+1    ;   "PRBYTE" @ $FDDA.
100 toHEX    JSR PRNTAX
101          JSR CROUT
102 *
103 toBIN    LDA #"%"        ; I added this trailing
104          JSR COUT        ;   conversion to 'BINARY'.
105 *
106 FAKEbin  LDY #9
107 Again    DEY             ; Y = 8 --> 1.
108 LEFT     ASL LINNUM+1    ; 1st LINNUM+1, then LINNUM.
109          BCS ONE         ; "0" or "1" ??
110 ZERO     LDA #"0"
111          HEX 2C          ; Trick to skip to "Display".
112 ONE      LDA #"1"
113 Display  JSR COUT        ; Saves Y-register.
114          CPY #1
115          BNE Again       ; Done after 8 bits.
116 *
117 Done     LDA LEFT+1      ; From "ASL LINNUM+1"
118          CMP #LINNUM+1   ;   to "ASL LINNUM".
119          BNE Restore
120          DEC LEFT+1
121          BNE FAKEbin     ; Re-cycle Y-register.
122 *
123 Restore  INC LEFT+1      ; Restore to "ASL LINNUM+1".
124          RTS             ; Finished with both bytes.
125
126
127 *-------------------------------------------------
128 *  HEX -to- DECIMAL conversion ( by : Val Golding ).
129 *-------------------------------------------------
130
131
132 HEXin    LDY #$FF        ; Get ASCII input.
133 ]BACK    INY             ; Y counts # of input char.
134          JSR CHRGET      ; Get ASCII char from "INBUFF".
135          BEQ DONEhex     ; If = #$00, then EOL or ":".
136          ORA #$80        ; Set Hi bit for output.
137 DONEhex  STA INBUFF,Y    ; Stuff back into Input Buffer.
138          BNE ]BACK
139 *
140          LDA #"!"
141          JSR COUT
142          JSR ZMODE       ; Convert ASCII to HEX
143          JSR GETNUM      ;   (2 char = 1 byte).
144          LDX A2L         ; Retrieve HEX bytes
145          TXA             ;   and save them.
146          PHA
147          LDA A2L+1
148          PHA
149 toDEC    JSR LINPRT      ; Decimal print routine.
150          JSR CROUT
151 *
152          PLA             ; A2L+1 = hi byte.
153          STA LINNUM+1
154          PLA             ; A2L = low byte.
155          STA LINNUM
156          LDA #0          ; Use Relocatable Code to
157 TObin    BEQ toBIN       ;   convert to 'BINARY'.
158
159
160 *-------------------------------------------------
161 * 'BINARY'-to-HEX-to-DECIMAL conversion (by:John Love ).
162 *-------------------------------------------------
163
164
165 **** MACRO to convert from pseudo to real 'BINARY'  ****
166
167
168          DO 0            ; Assembly off.
169
170 shift    MAC
171          LSR INBUFF+2,X ; Conditions Carry for "ROR".
172 *                        ;   Offset 2 to by-pass "&%".
173          ROR ]2          ; SHIFT or SHIFT+1.
174          CPX #0
175          BNE CONVERT
176 :more   LDA ]1          ; DIGITS or DIGITS+1 = # of
177          BEQ HEXout      ;   shifts to go.  If equal
178 *                        ;   to 0, then output HEX.
179          CLC             ; Mandatory for "ROR" to
180          ROR ]2          ;   produce leading "0"s.
181          DEC ]1
182          BNE :more
183          EOM
184
185          FIN             ; Assembly back on.
186
187
188 ******************  End of MACRO  ********************
189
190
191 BINin    LDA #0          ; Initialize just "SHIFT";
192          STA SHIFT       ;   "DIGITS" doesn't need
193          STA SHIFT+1     ;   to be initialized.
194 *
195          LDX #$FF        ; Get 'BINARY' input. X =
196 ]BACK    INX             ;   # of 'BINARY' digits.
197          JSR CHRGET      ; GETs #$30 or #$31 ('0','1').
198          BEQ DONEbin     ; If = #$00, then EOL or ":".
199          BNE ]BACK       ; Always!!
200 DONEbin  CPX #17         ; Error if > 16 bits input.
201          BCS ERROR
202          TXA             ; Accumulator = # of digits.
203
204 ***************************************************
205 *                                                 *
206 *  Here we determine DIGITS for both hi and lo bytes.  *
207 *  DIGITS is the amount of bit underflow in the byte;  *
208 *  for example, if there are only 5 bits in the byte,  *
209 *  then DIGITS = 3.  Note that this is accomplished by *
210 *  first subtracting 9, rather than 8.  This is done   *
211 *  in order to save memory on taking the absolute value*
212 *  of the Accumulator to store in DIGITS.  Normally,   *
213 *  with a negative Accumulator, its absolute value is  *
214 *  quantified by "EOR #$FF" followed by "ADC #$01";    *
215 *  in short, its "two's complement".  Subtracting 9    *
216 *  first, rather than 8, allows elimination of the     *
217 *  "ADC" instruction.                                  *
218 *                                                 *
219 ***************************************************
220
221 BytFUL?  SEC             ; Carry clear on entry.
222          SBC #9
223          STA LENless8    ; # of bits input less 8
224          INC LENless8    ;   (used later on).
225          BCS twoBYTs
226 oneBYT   EOR #$FF        ; Absolute value.
227          STA DIGITS
228          BCC CONVERT     ; Always!!
229 twoBYTs  SBC #8          ; Carry already set.
230          EOR #$FF        ; Absolute value.
231          STA DIGITS+1
232
233 ***************************************************
```

contd. on pg 27

OH . . . BY, BY THE WAY



by John A. Love III

In my investigations into the "sounds-Greek-to-me" world of Assembly language, I have accrued so far a rather hefty list of RAM and ROM addresses where some very "neat" pointers, registers and routines exist. For example, the following sequence displays a continuous row of the specified character from column (Y) over to the far right-hand edge of the screen:

```
LDY #column
LDA #character code (or, token)
JSR $FCA3
```

To make a long story short, for my own use I decided to make a single list consisting of the memory address, the corresponding label and a short descriptive comment or two. Clearly such a short comment would only serve as a "memory tickler" and could not possibly match the more comprehensive description available in other References such as "All About Applesoft" (CALL A.P.P.L.E. In Depth #1). The sole virtue of such a list is that it serves as a single source for a "quick and dirty look-see".

My list to follow is really rather short. The list for you more experienced programmers will, of course, be much, much longer. I recommend that you use your favorite Data Base program package to construct such a list so that you can later sort according to memory address, for example. Your own needs may dictate grouping according to memory-categories, such as "Zero Page Pointers", and then alphabetize within each category. Then you can use the "Report" section of your Data Base package to generate a printout.

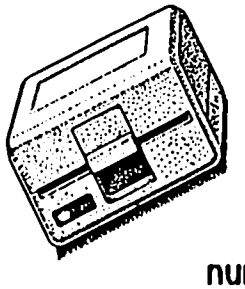
Well, ON WITH THE SHOW!!!

KEY ASSEMBLY CODE ADDRESSES

ADDRESS	LABEL	COMMENTS
$0024	CH	Cursor column position.
$0025	CV	Cursor row position.
$0028	BASL	Position at left margin.
$0032	INVFLG	= $FF (normal) ; = $7F (flashing) ; = $3F (inverse).
$0033	PROMPT	Prompt character.
$0036	CSW	Character output SWitch.
$0038	KSW	Keyboard input SWitch.
$003C	A1	Points to start of memory for MOVE.
$003E	A2	Points to end of memory for MOVE.
$0042	A4	Points to destination of memory for MOVE.
$0050	LINNUM	General purpose 16-bit register.
$0067	TXTTAB	Pointer to start of Applesoft program; defaults to $0801.
$0069	VARTAB	Pointer to LOMEM = simple variable space.
$006B	ARYTAB	Pointer to array variable space.
$006D	STREND	Pointer to end of current memory in use; start of free space.
$006F	FRETOP	Pointer to end of string storage; top of free space.
$0073	MEMSIZ	Ptr to HIMEM; string literals stored here down to FRETOP.
$0075	CURLIN	Pointer to current line #.
$0077	OLDLIN	Points to # of statement interrupted by CTRL-C, STOP, or END.
$0079	OLDTXT	Points to location of last byte (= 0) of the line just executed.
$007B	DATLIN	Points to current line # from which DATA are being read.
$007D	DATPTR	Points to address of first byte of DATA to be read.
$007F	INPPTR	Points to $0201 during INPUT; or current DATA stmt during READ.
$0081	VARNAM	Points to name of last referenced variable.
$0083	VARPNT	Points to value (or, length) of last referenced variable.
$009D	FAC	Floating point ACcumulator ($009D --> $00A2).
$00A5	ARG	ARGument register ($A5 --> $AA).
$00AF	PGMEND	Pointer to end of program in memory.
$00B1	CHRGET	Gets char fm BUFFER (Carry clear if #; A = 0 if EOL or :).
$00B7	CHRGOT	Previously GOTten character from CHRGET.
$00DA	ERRLIN	Points to line # in which error was trapped via ONERR GOTO.
$00DE	ERRNUM	Points to code of most recent error trapped via ONERR GOTO.
$0100	STACK	Stack (Page 1 of memory).
$0200	BUFFER	Input Buffer (Page 2 of memory).
$03D0	APPLE	Return to Applesoft via DOS warmstart (keeps HIMEM & BASIC).
$03EA	DOSHK	JuMPs to $A851 to reconnect DOS.
$03F2	RESET	Upon "CTRL--RESET", Monitor routine @ $FA62 JuMPs to ($03F2).
$03F5	AMPER	Ampersand vector.
$9D00	DOSBUFF	Points to DOS File Buffers.
$9D1E	DOSENTER	DOS Command Entry addresses (less 1).
$9E81	KBDincpt	JSRs to $9E81 to restore true I/O handlers to I/O hooks.
$9E8D	VIDincpt	Normal VIDeo intercept.
$A095	BLKfnBUF	Blank primary and secondary File Name Buffers.
$A1A4	FLUSHCMD	Flush blanks fm BUFFER; Z-flag set if "," or <CR> is 1st char.
$A1B9	CONVERT	HEX <---> DECIMAL; Carry set if there is an error.
$A851	IOINIT	Restore DOS : places your I/O hooks in $AA53-6.
$A884	DOSTEXT	DOS Command Text table.

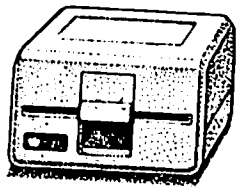
contd.

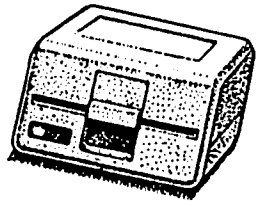
A DISK /// ON A // ?



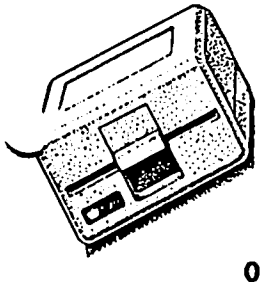
$199

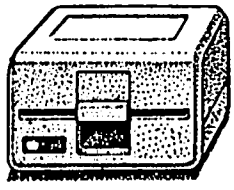
WOW!

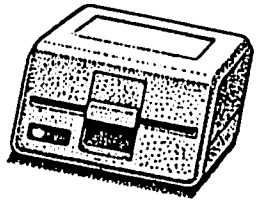




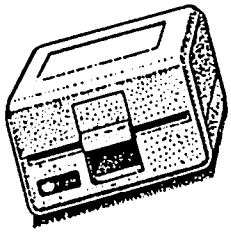
Clinton Computer had an opportunity to buy a limited number of brand new Apple Disk ///s at a very special price from Apple. We made this purchase because we have manufactured an adapter to make these Apple Disk ///s work on the Apple // Plus or Apple //e, with either DOS 3.3 or ProDOS. The adapter requires no modification to the computer, disk drive or cable. These drives will operate on an Apple // or an Apple ///.



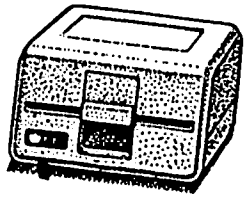


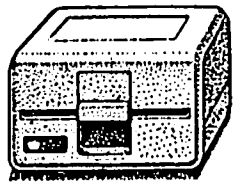


There are many third party drives for the Apple //. Most of them have a low initial price, but are more expensive over the life of the drive because they are not widely serviced or supported. The Apple Disk ///, on the other hand, was built as an improvement over the Disk //. The Disk /// works better and quieter, it looks nicer and it can be serviced by any authorized Apple dealer. A disk controller is also available for $69, if you need one.



Apple Disk ///s available only at our Clinton location. Call ahead for large quantities.





Clinton Computer

apple® Authorized Dealer

Branch Ave. (Rt. 5) at
6443 Old Alex. Ferry Rd.
Clinton, MD 20735
(301) 856-2500

$A971	DOSERROR	DOS Error Message table.
$AA53	cswTRUE	True CSW.
$AA55	kswTRUE	True KSW.
$AA5D	CMDlnIDX	Set = 0 for CONVERT.
$AA60	BINLNG	Length of most recently BLOADed program.
$AA72	BINSTART	Starting address of most recently BLOADed program.
$C000	KYBD	Input from KeYBoarD.
$C010	STROBE	Keyboard Strobe; clear via any access (LDA, STA, BIT etc.).
$C080	RAMrdWP2	Read and write-protect Bank 2 of RAM Card.
$C081	ROMrdWE2	ROM read and write-enable Bank 2 of RAM Card.
$C082	ROMrdWP2	ROM read and write-protect Bank 2 of RAM Card.
$C083	RAMrdWE2	Read and write-enable Bank 2 of RAM Card.
$C088	RAMrdWP1	Read and write-protect Bank 1 of RAM Card.
$C089	ROMrdWE1	ROM read and write-enable Bank 1 of RAM Card.
$C08A	ROMrdWP1	ROM read and write-protect Bank 1 of RAM Card.
$C08B	RAMrdWE1	Read and write-enable Bank 1 of RAM Card.
$D000	CMDTBL	Applesoft Command entry addresses (less 1).
$D0D0	TKNTBL	Applesoft Keyword Text table.
$D260	APPERROR	Applesoft Error Message table.
$D52C	INLIN	Main Applesoft input routine (no Prompt).
$D52E	INLIN1	Use X for Prompt, then proceed to INLIN.
$D539	GDBUFS	Put a "0" at the end of Input Buffer and mask each Hi bit.
$D61A	FNDLIN	Searches for line whose # is in (LINNUM).
$D66C	CLEARC	Clear variables and Stack.
$D6A5	LIST	List the whole program (the WHOLE thing!!!).
$D6DA	LSTLIN	List one line of the program.
$D86E	STOP	Stop the program.
$D870	END	Terminate execution of the program.
$DA0C	LINGET	Integer at TXTPTR to LINNUM.
$DAFB	CRDO	<CR> output.
$DB3A	STROUT	Print string at (Y,A); string must end with a quote or a 0.
$DB57	OUTSPC	Print a space (blank).
$DB5A	OUTQST	Prints a "?".
$DB5C	OUTDO	Print value in A.
$DD67	FRMEVL	Evaluate input at TXTPTR and put into FAC.
$DEC0	SYNCHR	Preceded by "LDA #$token", compares with character after "&".
$DEC9	SYNERR	Prints "Syntax Error".
$E000	BASICCLD	BASIC Cold start (= $20 for Integer & $4C for Applesoft).
$E003	BASICWRM	Warm entry to BASIC.
$E752	GETADR	FAC to LINNUM.
$E7A7	FSUB	(Y,A) --> ARG, then ARG - FAC.
$E7BE	FADD	(Y,A) --> ARG, then ARG + FAC.
$E97F	FMULT	(Y,A) --> ARG, then ARG * FAC.
$E9E3	CONUPK	(Y,A) --> ARG
$EA66	FDIV	(Y,A) --> ARG, then ARG / FAC.
$EAF9	MOVFM	(Y,A) --> FAC.
$EB2B	MOVMF	FAC --> (Y,X).
$EB53	MOVFA	ARG --> FAC.
$EB63	MOVAF	FAC --> ARG.
$EBB2	FCOMP	Compare FAC & (Y,A) : Acc = $01, $00, $FF for (Y,A) <, =, > FAC.
$ED24	LINPRT	Decimal printing routine for nbr. in X,A (Applesoft).
$EED0	NEGOP	FAC --> -FAC.
$F28C	RSTptr	Reset HIMEM & other pointers.
$F7E7	HTAB	Horizontal TAB # X spaces.
$F941	PRNTAX	Print A, then X, as HEX via PRBYTE.
$F94A	PRBL2	Output # X blanks.
$FB1E	PREAD	Read position of game controller: X = # of ctrlr ; Y = pos.
$FB5B	TABV	VTAB to value in A and HTAB to value in CH.
$FB6F	PowerUP	Set Power-up byte to prepare for RESET.
$FC22	VTAB	VTAB to CV.
$FC58	HOME	Clear the entire screen.
$FC9C	CLREOL	Clear to end of line.
$FCA3	LINE	LDY #start + LDA #char + JSR LINE --> prt char across scrn.
$FD0C	RDKEY	ReaD KEYboard ($C000) and put into A.
$FD1B	KEYIN	Normal true input handler; get KEYboard character INto A.
$FD35	RDCHAR	Get an input character or an ESCape code via RDKEY.
$FD67	GETLNZ	Print <CR>, then proceed to GETLN.
$FD6A	GETLN	Input line via RDCHAR; store in BUFFER; X = line length.
$FD8E	CROUT	<CR> output.
$FDDA	PRBYTE	Prints A as HEX.
$FDED	COUT	Output to Monitor.
$FDF0	COUT1	Normal true output handler.
$FE2C	MOVE	MOVE memory, using A1, A2 and A4.
$FE80	SETINV	Set INVerse video mode for COUT1.
$FE84	SETNORM	Set NORMal video mode for COUT1.
$FF3A	BELL	Ring-a-ding-ding!!!
$FF3F	IOREST	Restore all registers (A, X, Y, P, S) from $45 --> $49.
$FF4A	IOSAVE	Save all registers (A, X, Y, P, S) in $45 --> $49.
$FFA7	GETNUM	Convert ASCII to HEX.
$FFC7	ZMODE	Prepare for GETNUM by setting Y = MODE ($31) = 0.

DISCOUNTS!

5¼ DISKETTES & STORAGE

- SS/DD BOX OF 10 — $12.00 *
- SS/DD 10 BOXES — $115.00 *
- DOUBLE-NOTCHED DS/DD EACH — $1.35 *
- DOUBLE-NOTCHED DS/DD 100 — $125.00 *
- HARD PLASTIC STAND-UP DISKETTE LIBRARY CASES — $2.75 EACH / 4 FOR $10.00
  (specify color choices: beige, black, blue, green, grey, red, yellow)
- SMOKED PLASTIC JUMBO SIZE FLIP TOP 75 DISKETTE FILE CASES — $18.00
- 70-DISKETTE FILE CASES — $16.00
- 140-DISKETTE LOCKING WOOD FILE CABINET — $33.00

PRINTERS

- PANASONIC P1090 — $239.00 *
- PANASONIC 1091 — $299.00 *
- EPSON RX-80 F/T — $289.00
- EPSON RX-100 — $389.00
- EPSON FX-80 — $379.00 *
- OKI DATA MICROLINE 92A DOT MATRIX — $369.00
- SILVER REED 400 LETTER QUALITY — $269.00 *
- STARWRITER A-10 25CPS LETTER QUALITY — $495.00
- TOSHIBA 1340 DOT MATRIX AND LETTER QUALITY COMBINED — $795.00

PRINTER INTERFACES AND ACCESSORIES

- STANDARD PARALLEL INTERFACE CARD — $49.00
- GRAPHICS PARALLEL INTERFACE CARD — $75.00
- FINGERPRINT PUSH BUTTON GRAPHICS CARD — $119.00 *
- MICROFAZER GENERAL PRINT BUFFER — $149.00
- PRINTER STAND — $14.00
- SWITCH BOX 3 PARALLEL PORTS — $129.00
- SWITCH BOX 3 SERIAL PORTS — $79.00

FLOPPY DISK DRIVES

- FOURTH DIMENSION (FULL OR SLIMLINE) — $179.00
- DISTAR — $159.00 *
- LASER SLIMLINE — $154.00
- IIc DRIVES — $173.00 *
- DISK CONTROLLER — $59.00
- DOUBLE-SIDED DRIVE — $199.00 *
- 650K RANA DRIVE — $439.00

HARD DISK DRIVES

- 5 MEGABYTE WITH CONTROLLER AND SOFTWARE — $749.00
- 10 MEGABYTE — $1175.00

MONITORS

- GORILLA 12-INCH GREEN — $84.00
- GORILLA 12-INCH AMBER — $89.00
- SYNCO 12-INCH AMBER — $74.00
- 12 INCH GREEN WITH INVERSE — $99.00
- INTRA 14-INCH COMPOSITE COLOR/80 COLUMN — $239.00 *

MODEMS

- ZOOM TELEPHONICS 300 BAUD — $109.00
- CENTAURI 300 BAUD — $179.00
- PRO-MODEM 1200 — $349.00
- PRO MODEM 1200A INTERNAL — $279.00 *
- SINGALMAN MARK XII — $239.00 *

GRAPHICS DEVICES

- POWER PAD & STARTER KIT — $99.00 *

VIDEO & DISPLAY EQUIPMENT

- DIGITIZER — $299.00
- B & W CAMERA — $195.00
- COLOR PROCESSOR — $99.00
- COLOR PROCESSOR/ENHANCER STABILIZER/SYNTHESIZER — $279.00

GENERAL ITEMS

- 6 OUTLET POWER STRIP — $19.00
- SURGE PROTECTOR — $11.00
- RF MODULATOR — $49.00
- COMPUTER STAND — $24.00

GAME I/O DEVICES

- CH MACH II JOYSTICK — $37.00
- CH MACH III JOYSTICK — $45.00

SLOT EXPANSION

- 16 RAM CARD — $49.00
- 64K RAM & 80 COLUMN CARD FOR IIe — $109.00 *
- MEMORY MASTER IIE 64K + RAM & 80 COLUMN CARD — $145.00
- MEMORY MASTER IIE 128K RAM & 80 COLUMN CARD — $195.00
- MICROTEK II + 128 K VISICALC AND MEMORY EXPANSION — $219.00
- MODEM ELIMINATOR CABLE — $21.00
- SERI-ALL SERIAL INTERFACE CARD — $119.00
- 80-COLUMN CARD (VIEWMASTER) WITH SOFT SWITCH — $129.00 *
- CENTAURI APS Z-80 CARD — $59.00
- Z-80 PLUS CARD (CPM FOR APPLE) — $115.00
- FAST Z-80 CARD APPLICARD — $175.00 *
- TIMEMASTER II CLOCK/CALENDAR CARD — $109.00
- QUICK-LOADER PROM BOARD — $149.00
- ANALOG/DIGITAL BOARD — $99.00
- SUPER I/O BOARD — $49.00
- MULTIPLE-SLOT EXPANSION CHASSIS — $149.00
- SINGLE SLOT EXTENDER — $29.00

SPECIAL PERIPHERALS

- COOLING FAN WITH SURGE PROTECTOR — $39.00
- TITAN KEYBOARD — $159.00
- LIFETIME EXTERNAL POWER SUPPLY — $179.00
- SHIFT KEY MOD KIT — $8.00
- SCREEN SWITCHER/ DRIVE STEPPER — $74.00

APPLE SOFTWARE

- WORD STAR — $195.00
- MAIL MERGE/SPELL STAR/ STAR INDEX — $125.00 *

* DENOTES NEW PRICE OR ITEM

LONG DISTANCE CALL TOLL FREE WITH TOUCH TONE PHONE FROM ANY CITY!
DIAL 950-1088
WAIT FOR TONE
DIAL 363-1313

VISIT OUR NEW STORE LOCATION
8231 WOODMONT AVE
(AT BATTERY LANE) IN BETHESDA

ORDERS & CALL BACK MESSAGES (ANSWERING MACHINE)
(202) 362-9176

UPS shipping, $4.00 per order plus $6.00 per printer or monitor

SPECIAL SALE
APPLE IIc DRIVES WITH CABLE
$173.00*

(202) 363-1313
VF ASSOCIATES

6327 WESTERN AVE NW WASHINGTON DC 20015 (MAIL ORDER)

OH . . . BY, BY, BY THE WAY

by John A. Love III

In many of my previous articles I developed relatively short Assembly language subroutines such as my first one presented in this month's Journal. As can be seen, I set the ORG or execution address at or shortly below the beginning of Page 3 of RAM. Sandy Mossberg, in his continuing "Disassembly Lines" series published in Nibble, has presented other locations for short routines, for example, unused small blocks of memory inside DOS. These other locations aside, the sometimes "lazy" approach is simply to stuff my little routine inside Page 3 of RAM. If I'm willing to wait and worry later about conflict between multiple routines all having ORG addresses of $0300, Page 3 of RAM is okay under one and only one condition. This "condition" is precisely the subject of this present article.

This single condition is that my or your small Assembly language routine must not go beyond $03CF when assembled, that is, it must not "touch" $03D0 through $03FF. This area must be reserved for the DOS Vector Table. According to Don Worth and Pieter Lechner, authors of Beneath Apple DOS:

"These vectors allow access to certain places within the DOS collection of routines via a fixed location ($03D0, for instance). Because DOS may be loaded in various locations . . . . . the addresses of the externally callable subroutines within DOS will change. By putting the addresses of these routines in a vector at a fixed location, dependencies on DOS's location in memory are eliminated . . . . . "

I have disassembled this DOS Vector Table, with the results presented below. Clearly such a disassembly is not a big deal for two significant reasons: first, Beneath Apple DOS presents the contents of the DOS Vector Table in narrative form; second, disassembling 48 bytes of Machine code shouldn't be terribly difficult (Wanna bet ?!*?). The sole reason for my presenting this disassembly is that the DOS Vector Table is truly significant for any Assembly language programmer. Many of the vectors, such as $03D0 and $03EA, should be instantly recognizable; however, many were new to me. As a direct result, I wanted to share this new-found knowledge. Remember one key variable - you, the programmer, are free to change many of these vectors and therein lies their beauty.

And now, ON WITH THE SHOW!!!

```
:ASM
                          1  ********************
                          2  *                  *
                          3  *  DOS VECTOR TABLE  *
                          4  *                  *
                          5  *    ($3D0--$3FF)    *
                          6  *                  *
                          7  ********************
                          8
                          9
                         10           ORG  $3D0
                         11
                         12
                         13  WARM     =    $9DBF      ; DOS warmstart.
                       * 14  COLD     =    $9D84      ; DOS coldstart.
                         15  FM       =    $AAFD      ; DOS's File Manager
                         16  *                        ;   external entry point.
                         17  RWTS     =    $B785      ; Read/Write Track/Sector
                         18  *                        ;   external entry point.
                         19  FMPARM   =    $9D0E      ; 16-bit register has
                         20  *                        ;   the location of the
                         21  *                        ;   DOS File Manager
                         22  *                        ;   parameter list.
                         23  RWTSPARM =    $AAC1      ; 16-bit register has
                         24  *                        ;   the location of the
                         25  *                        ;   RWTS parameter list.
                         26  IOINIT   =    $A851      ; Reset I/O vectors.
                         27  OLDBRK   =    $FA59      ; Displays all the 6502
                         28  *                        ;   Registers upon "BRK".
                         29  RTS      =    $FF58      ; Default for "&" vector
                         30  *                        ;   contains an "RTS".
                         31  MON      =    $FF65      ; Clears decimal flag,
                         32  *                        ;   rings the bell and
                         33  *                        ;   "CALLs -151".
                         34
                         35
                         36  *-------------------------------------------
                         37  *  DOS warmstart : keeps HIMEM & BASIC program;
                         38  *  sets CSW State #0 and finally JuMPs to the
                         39  *  main Applesoft Command Loop @ $D43C that
                         40  *  handles all input from the immediate mode.
                         41  *-------------------------------------------
                         42
03D0: 4C BF 9D           43  DOSwarmV JMP  WARM
                         44
                         45  *-------------------------------------------
                         46  *  DOS cold start : clears BASIC program and
                         47  *  resets HIMEM; sets Slot # = 6 & Drive # = 1;
                         48  *  sets up Applesoft ROM entry point vector table;
                         49  *  sets up I/O hooks via "IOINIT" and finally
                         50  *  JuMPs to the main Applesoft Command Loop @
                         51  *  $D43C that handles all immediate mode input.
                         52  *-------------------------------------------
                         53
03D3: 4C 84 9D           54  DOScoldV JMP  COLD
                         55
                         56  *-------------------------------------------
                         57  *  Calls File Manager by JuMPing to the entry
                         58  *  points in the File Manager Function Handler
                         59  *  table @ $AAC9 - $AAE4.  The File Manager
                         60  *  occupies $AAC9 - $B600, essentially the middle
                         61  *  third of DOS that passes DOS program flow to
                         62  *  any one of 12 Function Handlers.
                         63  *-------------------------------------------
                         64
03D6: 4C FD AA           65  DOSfmV   JMP  FM
                         66
                         67  *-------------------------------------------
                         68  *  Disables Interrupts and calls the main entry
                         69  *  to RWTS @ $BD00.  RWTS occupies $B600 - $BFFF,
                         70  *  essentially the top one-third of DOS.
                         71  *-------------------------------------------
                         72
03D9: 4C B5 B7           73  DOSrwtsV JMP  RWTS
                         74
                         75  *-------------------------------------------
                         76  *  Locates the address of the input parameter
                         77  *  list for DOS' File Manager @ $B5BB - $B5D0.
                         78  *-------------------------------------------
                         79
03DC: AD 0F 9D           80  FMinV    LDA  FMPARM+1
03DF: AC 0E 9D           81           LDY  FMPARM
03E2: 60                 82           RTS
                         83
                         84  *-------------------------------------------
                         85  *  Locates the address of the input parameter
                         86  *  list for RWTS @ $B7E8.
                         87  *-------------------------------------------
                         88
03E3: AD C2 AA           89  RWTSinV  LDA  RWTSPARM+1
03E6: AC C1 AA           90           LDY  RWTSPARM
03E9: 60                 91           RTS
                         92
                         93  *-------------------------------------------
                         94  *  Re-connects DOS : places your I/O hooks in
                         95  *  $AA53 - $AA56 & replaces $9E81 into KSW
                         96  *  and $9EBD into CSW.
                         97  *-------------------------------------------
                         98
03EA: 4C 51 A8           99  DOSHK    JMP  IOINIT
03ED: EA                100           NOP             ; 2-byte filler.
03EE: EA                101           NOP
                        102
                        103  *-------------------------------------------
                        104  *  Keeps address where "BRK" was encountered;
                        105  *  displays 6502 Registers and then JuMPs to
                        106  *  "MON".  This is supported only by the
                        107  *  Autostart ROM.
                        108  *-------------------------------------------
                        109
03EF: 4C 59 FA          110  BRK      JMP  OLDBRK
                        111
                        112  *-------------------------------------------
                        113  *  Upon "RESET" or "CTRL-RESET", Monitor routine
                        114  *  @ $FA62 JuMPs to ($03F2), the Warm Start
                        115  *  soft entry vector.
                        116  *-------------------------------------------
                        117
03F2: BF 9D             118  SOFTEV   DA   WARM
                        119
                        120  *-------------------------------------------
                        121  *  Routine @ $FB6F takes complement of byte @
                        122  *  $03F3 with #$A5 in order to effect "RESET"
                        123  *  or "CTRL-RESET" JuMPing to your routine.
                        124  *-------------------------------------------
                        125
03F4: 38                126  PRWEDUP  HEX  38         ; #$9D + EOR #$A5 = #$38.
```

contd. on pg 27

WELL, I'LL BE . . . IT REALLY WORKS

by John A. Love III

First, I would like to thank Sandy Mossberg for the "Try it, it just might work!!!" encouragement that he continuously fosters in "Disassembly Lines", his monthly tutorial published by Nibble.

Second, I feel obligated to present some lengthy background material in addition to the equally lengthy comments that are part of the code presented at the end of this article.

I can hear the "cat calls" now -- "All right, Love, get to the point!!!" Well, the point is how to add your own favorite Applesoft commands to Apple memory. Sandy presented a generalized technique for adding customized DOS commands. Since the four parts of DOS - the File Buffers, the Command Interpreter, the File Manager and RWTS - all reside in $9600-->$BFFF, Sandy executed all his changes within RAM. But Applesoft is in ROM, $D000-->$F7FF, and how in the world do I write to ROM??? Aha, move Applesoft to your RAM Card in Slot 0, make all the changes you wish and then execute your Applesoft program totally on the RAM Card!!! What if you don't have a RAM Card? Well, that solution will have to wait for another issue of the Journal. Besides, I've already got an armload of fundamentals to cover for you kind readers, fundamentals of such extreme importance that I don't want to "muddle" my already-senile brain with more facts.

Let's jump right into the fundamentals I've mentioned. During November 1983 I presented in the Journal an article that really only scratched the surface in explaining how your keypress got moved to Apple memory and, further down the chain, got presented on the video monitor. At that time, I stated that:

"1. the KEYBOARD DECODER translates your keypress to its high byte binary equivalent.

2. this high byte result is stored in the INPUT/OUTPUT (I/O) BUFFER.

3. the BASIC INTERPRETER steps in to 'understand' this stored data and subsequently changes the data to low byte, that is, with Bit 7 = 0 or positive ASCII.

4. the 6502 Central Processing Unit (CPU) then calls the Character OUTput (COUT) routine at $FDED to send the stored low byte character to the user-defined output device, for example, the video monitor, the printer or the disk drive."

Let's expand on this a little and, most importantly, make some corrections while we're at it.

The KEYBOARD DECODER is a piece of hardware physically located beneath your Apple keyboard. To quote from What's Where in the APPLE, authored by William F. Luebbert and published by Micro Ink, "the heart of the APPLE is the FETCH--EXECUTE cycle". Professor Luebbert goes on to say that "FETCHing the information brings the byte into circuitry where it decodes it as a code that specifies the . . . operation to be performed. Not surprisingly, this byte is called the Operation Code part of the computer instruction. The instruction will also contain information on finding the data that is to be used in the operation. Normally this is done by specifying the address of the memory location in which the data to be used may be found." Thus, we have the two parts of every instruction -- the Operation Code and the Operand. Although this applies in general to all of Apple memory, it does not apply precisely to the KEYBOARD DECODER. The primary functions of the KEYBOARD DECODER do include however:

a) FETCH your keypress and decode it into an equivalent negative ASCII code.

b) The I/O hardware occupies $C000-->$CFFF and it is this hardware that (under the direction of the 6502 CPU, the "band conductor") determines where all I/O is to go, for example, your keypress into $C000. The KEYBOARD DECODER, via its circuitry, stores or places the negative ASCII code of your keypress into location $C000.

Okay, I've FETCHed my keypress and decoded it. What now??? DOS to the rescue!!!

When the DOS 3.3 System Master diskette is booted, DOS is loaded into $9600-->$BFFF. Of particular importance is essentially the first third of DOS, the Command Interpreter which contains all of DOS' main routines. This first third resides at $9D00-->$AAC8. Part of this memory range encompasses the DOS keyboard and video output intercept routines whose code starts at $9D02. That genius Sandy Mossberg has exhaustively described these intercept routines in his already-mentioned "Disassembly Lines" series in Nibble. Suffice it for me to say that upon booting, DOS eventually accesses $9ED1 which, among other things, places the true I/O handlers into the so-called I/O switches. For input, the true handler is the System Monitor routine beginning at $FD1B which loads into the Accumulator the value in $C000. For output, the true handler is the System Monitor routine beginning at $FDF0 which outputs the value in the Accumulator to the video monitor via "VIDOUT" at $FBFD.

Very sketchily, I admit, I've progressed through the KEYBOARD DECODER to placement of the true I/O handlers into the I/O switches. How does the latter make I/O happen??? To answer this important question, let's continue with DOS' Command Interpreter. If DOS is cold-started, it accesses $9D84 and, if warm-started, $9DBF (see a previous article in this Journal). In either case, a final JuMP to $D43C (labelled "CMDLP") is executed. As Sandy Mossberg has explained, CMDLP is the ROM Applesoft routine that:

a) obtains input and determines if the input represents a direct command or a program line.

b) if a direct command is issued, CMDLP executes it.

c) if a program line is entered, CMDLP processes it by resetting critical Page 0 program pointers, deleting the old line and/or inserting the new one.

Although strictly speaking not a part of CMDLP, "INLIN" at $D52C serves as the main Applesoft input subroutine. It turns out that INLIN is accessed immediately upon entry to CMDLP. Upon entry to INLIN, "GETLN" at $FD6A reads the first keyboard press at $C000 (by accessing the true input handler mentioned earlier) and stores it in the Input Buffer ($0200-->$02FF). If not a <CR>, then GETLN displays your keypress on the screen by accessing the true output handler. The offset (X-register) inherent to GETLN is "bumped" one for each successive character in your

contd.

input line and your next keypress is read and displayed. In this manner, the X-register counts the # of characters in your input line. As soon as a <CR> is detected, the whole input line has been read, displayed and stored all via the above I/O handlers. The remaining portion of the video line is then cleared and a <CR> subsequently generated. At this point GETLN is finally exited, returning you to INLIN. It should be noted here that if the number of characters (X) retrieved by GETLN exceeds 239, then X is set = 239 (all this time GETLN had been displaying and storing up to 255 characters; however, the Input Buffer length is cut off after your return to INLIN).

INLIN continues with "GDBUFS" at $D539 which:

a) places a HEX 00 as an End-of-Line (EOL) marker at the end of your keypress-stream, now stored in the Input Buffer as negative ASCII.

b) converts the negative ASCII contents of the Input Buffer to positive ASCII, that is, GDBUFS clears the high Bit (sets Bit 7 = 0).

c) resets "TXTPTR" ($00B8,9) so "CHRGET" at $00B1 will FETCH the first character (now positive ASCII) in the Input Buffer. For a disassembly of CHRGET, see the September 1981 issue of Apple Assembly Line, a very excellent source for Assembly Language programmers.

Okay, I've covered the input/output of your successive keypresses : besides reading and displaying, GETLN stores your successive keypresses in the Input Buffer or Page 2 of RAM; GDBUFS then clears the high Bit of each Byte in this Input Buffer and initializes TXTPTR to the beginning of the Input Buffer. By the way, the Byte (preceding TXTPTR) at $00B7 is labelled "CHRGOT" because it loads the character already GOTten by GETLN.

Well, I seem to have reviewed some old material. The reason for this review is that the relatively greater depth I have provided here (vice in November 1983) is really mandatory to understanding how to add your own customized Applesoft commands. As I mentioned earlier, I chose here to utilize the RAM Card in Slot 0; even if I didn't, however, it makes no difference. This in-depth review is still necessary.

"Why?" Good question - stay tuned!!! Let's continue with CMDLP to get the answer. Whether CMDLP is executing an immediate or direct command or processing a program line, CMDLP accesses $D559, called "PARSE". PARSE, as the name perhaps implies, condenses (where it can) the command or program line. Oh boy!!! It looks like we're going to have to discuss (again, sketchily) how Applesoft stores your direct commands/program lines. Well, let's not delay the inevitable - let's do it!!!

This tangent is definitely going to be sketchy - I highly recommend that you review All About Applesoft, published by CALL A.P.P.L.E., for a thoroughly enjoyable and comprehensive journey through this direct command/program line storage process. Suffice it for me to say that Applesoft program lines are stored beginning at $0801 or wherever you have directed the program pointer at $0067,8 ("TXTTAB").

a) $0800 : HEX 00 to indicate the start of the direct command or first program line.

b) $0801,2 : address of the next Applesoft line.

c) $0803,4 : current line number.

d) $0805 : start of line content in ASCII.

e) $08XX : HEX 00 to indicate the end of the program line and the beginning of the next one.

Let's expand the content of $0805 which is the start of the first line's content in ASCII. As Sandy Mossberg explains, PARSE will "TOKENize" that portion of the Input Buffer between TXTPTR (after GDBUFS, pointed to $01FF) and the EOL marker placed by GDBUFS. The myriad of puzzle pieces appear to be forming a picture:

1. To begin with, the KEYBOARD DECODER stuffs negative ASCII code into $C000.

2. The System Monitor, via GETLN, eventually reads $C000 and displays the corresponding character on the video monitor. GETLN also stuffs this same negative ASCII code into the Input Buffer. This "read--display--store" process is repeated until you press <CR>, thus forcing an exit from GETLN.

3. GDBUFS converts everything in the Input Buffer to positive ASCII.

4. Now, PARSE enters to look at the contents of this Input Buffer. And PARSE begins with the beginning too because GDBUFS was smart enough to put TXTPTR there before PARSE crashed your party. PARSE even knows where the Buffer's stored line ends because GDBUFS was doubly-smart to place an EOL marker at the end. Gosh!!!

PARSE effects the TOKENization of the Input Buffer contents by "simultaneously" scanning the Input Buffer and the Keyword Text Table ("TKNTBL") of 107 key Applesoft words in order to determine if there is a match. Except for "&" and the several arithmetic operators, each keyword is stored in DCI format within this table beginning at $D0D0. The reason for the DCI format is so PARSE can recognize when each keyword ends (in DCI format all characters, except the last one, have their hi Bit clear - that is in positive ASCII). The already-mentioned exceptions are one byte long, stored in negative ASC format. The DCI-formatted words are at least two bytes long; in Neither case, the last byte in any keyword is in negative ASCII.

Each word in TKNTBL represents an Applesoft Command like 'GET' or a keyword like 'THEN'. By virtue of the ordering within this Text Table, each Command or keyword can be transposed to a one-byte equivalent representation of its position within TKNTBL. In short, if I know the position the keyword occupies, by default I know what the keyword is. This one-byte position descriptor (called a "TOKEN") is in negative ASCII; therefore, every Applesoft keyword has a one-byte equivalent TOKEN ranging from $80 to $EA.

It turns out that if a match develops between a word in the Input Buffer and a word in TKNTBL, then PARSE replaces the Buffer's word with its equivalent TOKEN. As a direct result, your line length shrinks. It also turns out that if part of the Applesoft line's content cannot be TOKENized, that part remains stored as positive ASCII. Brilliant folks, those ROM writers, because now we have a direct measure of whether the stored line byte is a TOKEN or not. Positively brilliant!!! Some "Hi - 5's" in the Endzone, please!!!

After PARSE effects the appropriate condensing of your typed line within the Input Buffer, we return to CMDLP. If a program line and not a direct command, CMDLP does the "dirty" work of storing your program line beginning at $0801 or wherever your previous line(s) left off. This transfer from the Input Buffer to $0801++ even includes the EOL (HEX 00). So, this is how that EOL marker in $08XX got there???

contd.

So far we've stored each program line in succession beginning at $0801, with all keywords TOKENized. We know where each line ends because of the EOL marker following each line. We also know if there is another line to follow because the current line being processed starts off with the memory location of the next line. If there is not one to follow, then at the specified memory location of the "next" line there will be successive HEX 00's. Now we're still back to CMDLP that called PARSE to begin with.

Wait!!! Hold it!!! Gotta rest a bit.

(Three days later. . . . .) Okay, we're back to CMDLP. Now what happens with a direct command??? We see after TOKENization of the line contents (if appropriate) that a JuMP to $D805 eventually JuMPs back to $D7D2 which Sandy Mossberg labels "NEWSTT". This latter routine deals with the execution addresses of the executable Applesoft keywords. These addresses are stored in another table beginning at $D000, commonly referred to as "CMDTBL" for "Applesoft Command Table". Each address is the entry point (less one) where the Applesoft command is executed. It turns out that there are 64 Applesoft commands. There are, then, an additional 43 Applesoft words in TKNTBL that are keywords only, not commands; therefore, these latter words are found only in the Text Table, not the Command Table. For further edification Sandy Mossberg points out that these additional 43 either are referenced by one of the commands (for example, 'THEN' is referenced by the routine that executes 'IF') or are contained within a third Applesoft table, the Function Table (for example, 'SIN' or 'COS'). Since we are concerned with execution of a direct command at this point, let's worry only about the set of 64.

Returning to the operation of NEWSTT, it also turns out that the above-mentioned 64 executable commands are the first 64 keywords stored within the Text Table starting at $D0D0. NEWSTT scans the input line looking for a TOKEN which can easily be differentiated from the rest of the line contents because a TOKEN is stored in negative ASCII (remember I told you that those ROM writers were clever!!!). So, NEWSTT sees a character whose high Bit is set, knows it's a TOKEN as a result and, then, subtracts #$80 to determine its relative position. After quantifying this relative position, NEWSTT accesses the entry address where the command is executed. Execution now begins!!! Tralah!!!

Okay, we've covered executing a direct command. How 'bout RUNning an entire Applesoft program??? RUN executes beginning at $D912 according to CMDTBL. Guess what??? $D912 eventually accesses NEWSTT. Well, that was easy!!!

Believe it or not, I think we're almost ready for the actual code that adds my favorite Applesoft command. My favorite happens to be the equivalent of "GOTO NAME", where "NAME" has been previously defined through a "LET" statement. As all of you know, those infamous ROM writers neglected this command; however, now you have all the requisite tools for implementing it. So, let's go for it!!!

The steps are few and quite simple looking. I have described them in the Assembly code that ends this article. A "few" more preparatory remarks need saying, however, before I present the code itself. Hang in there!!!

Not only do I move all of ROM (Applesoft as well as the System Monitor), but I also move the actual code of the new command to the RAM Card so that all of Applesoft (including my additions) can be executed on the RAM Card itself. By virtue of the machinations inherent to writing to the RAM Card, any new code pegged for movement to this Card must first be loaded into low RAM (below $C000) and then moved. When we finally get to the code itself you readers will see that the code for this new command was loaded into the much-too-often used area, $0300-->$03CF.

The last "aside" before the actual code is presented entails at least a sketchy description of how ROM Applesoft stores variables and, in particular, real numbers. In All About Applesoft Eric Goez presents his "Real Variable Study" that describes this phenomenon. Essentially, the address pointed to by the Page 0 vector at $0083,4 starts a string of five bytes that represent how real variables are actually stored in Apple memory. Let's press on by using a couple of examples. Let's look at the two real numbers, 362.5625 and -2016.75.

In either case, if we clear RAM by typing "NEW" followed by "A =" a value, we see that the above Page 0 vector called "VARPNT" points to $0806. Examination of the five bytes beginning with $0806 reveals:

1) 89 35 48 00 00

2) 8B FC 18 00 00

Realizing that these are HEX bytes, let's convert them to their binary equivalent:

3) 10001001 00110101 01001000 00000000 00000000

4) 10001011 11111100 00011000 00000000 00000000

Now, let's pretend that we don't know what's in $0806++ and look at the binary representation of each decimal number:

5) + 362.5625 = % 000101101010.10010000

6) +2016.7500 = % 011111100000.11000000

Next let's drop all leading zero's and replace the very first "1" with a "0" (after all, the very first "1" is always a "1"; so why retain it???):

7) + 362.5625 = % 001101010.1001

8) +2016.7500 = % 01111100000.1100

We're getting there - now let's convert to scientific notation by placing the decimal point to the far left, determine the appropriate exponent in binary and placing this exponent before the mantissa:

9) + 362.5625 = % 1001.0011010101001

10) +2016.7500 = % 1011.011111000001100

Oh yes, just for the sake of a little symmetry, just a little, let's put both leading and trailing zero's in sufficient quantity so there is one full byte before the decimal point to represent the exponent and four full bytes after to represent the mantissa. And, another thing . . . . . since we now have a string of 40 bits with one(1) byte before the decimal point, this decimal point is always eight bits in from the left. So, who needs it??? Away with it!!!

11) + 362.5625 = % 00001001 00110101 01001000 00000000 00000000

12) +2016.7500 = % 00001011 01111100 00011000 00000000 00000000

Oh, oh . . . . . I think I blew it?!*!? Isn't that second decimal number negative??? Okay, I've got

contd.

it!!! Do you remember that "very first '1'" I dropped several steps back. That single bit is the first bit in the mantissa and, in scientific notation, the sign of the number is associated with the mantissa. So, let's set that first mantissa bit to "1" to depict a negative number and to "0" for a positive number.

Finally, in a totally arbitrary manner, let's set the high bit of the exponent byte. As a result:

```
13) + 362.5625  = % 10001001  00110101  01001000
                    00000000  00000000

14) -2016.7500  = % 10001011  11111100  00011000
                    00000000  00000000
```

Converting back to HEX:

```
15) + 362.5625  = $ 89  35  48  00  00

16) -2016.7500  = $ 8B  FC  18  00  00
```

These definitely should look familiar!!! And now, ONTO THE MAIN EVENT!!!

First, the two moving vans come - the first moves all of ROM Applesoft and the System Monitor; the second moves the code for the new command I have called "GONAME".

Speaking of name-calling, I then store the new name in DCI format at the appropriate position within TOKTBL. Then, through the judicious use of labels, I let the Assembled code do the work for me of writing trailing zero's to fill in that part of the replaced code (SHLOAD, $F775-->$F7BB) I don't consume with my shorter GONAME code. Note that by letting labels do the work for me, I didn't have to manually add or subtract when I was building or correcting the code.

The actual execution code for GONAME finishes the Assembly listing. Line 146 is a "stop-gap" against the possibility that I may inadvertently have "GONAME A$" in my Applesoft program, where A$ is not a number. If not a #, then an error message is displayed on the video monitor.

What is a FAC??? It is the Floating point ACcumulator located between $009D-->$00A2 that can be used as an "intermediate depot" which enables one Applesoft routine to "talk" to one another. For example, since PTRGET places the address of the exponent in (Y,A) = (Hi,Lo), I must have a vehicle to get all five bytes (exponent and mantissa) into LINNUM so that the regular GOTO (a #) can operate on the whole number. As described in the listing itself, the FAC serves as this vehicle.

Note that my entry to GOTO (a #) on Line 151 skips the first three bytes of the normal entry. These three bytes are "JSR $DA0C". The routine beginning at $DA0C, commonly labelled "LINGET", places the specified line number (#) into LINNUM. But I've already done that, so the "JSR LINGET" needs to be skipped.

A Really Nifty Utility contd. from pg 35

```
95CB: C4 D2 C5 D3 D3 A0 8D A0
95D3: A4 00
95D5: C5 D8 C9  435  EXIT     ASC  "EXIT!!!"8D,8D,00
95D8: D4 A1 A1 A1 8D 8D 00
95DF: 8D 8D 8D  436  ANOTHER  HEX  8D,8D,8D,87,87,87,87
95E2: 87 87 87 87
95E6: C1 CE CF  437           ASC "ANOTHER OPTION (Y/N)?? = "00
95E9: D4 C8 C5 D2 A0 CF D0 D4
95F1: C9 CF CE A0 A8 D9 AF CE
95F9: A9 BF BF A0 8D A0 00
                438
                439
                440  * ----- ENDs just under DOS @ $95FF -----

--End assembly, 1021 bytes, Errors: 0
```

ASSEMBLY CODE:

```
1   ********************************
2   *                              *
3   *  ADD a New Applesoft COMMAND *
4   *                              *
5   ********************************
6
7
8           ORG  $0300      ; Not again ?!*?
9
10
11  *-------------------------------------------------
12  *       EQUates follow:
13  *-------------------------------------------------
14
15
16  A1L     =    $3C        ; Start (lo) of code to be moved.
17  A2L     =    $3E        ; End (lo) of code to be moved.
18  A4L     =    $42        ; Dest.(lo) addr of moved code.
19  ROMrdWE2 =   $C081      ; ROM rd & RAM wrt-enable (Bk 2).
20  RAMrdWE2 =   $C083      ; RAM read & wrt-enable (Bk 2).
21  MOVE    =    $FE2C      ; Sandy says "Call the movers!".
22  CMDTBL  =    $D000      ; BASIC Cmd addr entry pts (- 1).
23  TOPMON  =    $FFFF      ; Top of the Monitor.
24  TOKTBL  =    $D0D0      ; BASIC Command Text table.
25  SHLOAD  =    $F775      ; Replaced Applesoft Command.
26  CHKNUM  =    $DD6A      ; Check FAC for numeric contents.
27  PTRGET  =    $DFE3      ; Get variable name, etc.
28  MOVFM   =    $EAF9      ; From memory @ (Y,A) to the FAC.
29  GETADR  =    $E752      ; From FAC to LINNUM @ $51,50.
30  GOTO    =    $D93E      ; Implements "GOTO line #".
31  APPLE   =    $03D0      ; To Applesoft via DOS warmstart.
32
33
34  *-------------------------------------------------
35  *  1st, move all Applesoft & the Monitor to the RAM Card.
36  *-------------------------------------------------
37
38
39  MoveIt  BIT  ROMrdWE2   ; Must access twice to effect.
40          BIT  ROMrdWE2
41          LDY  #>CMDTBL   ; I deliberately wasted space
42          STY  A1L+1      ;   in order to illustrate the
43          STY  A4L+1      ;   general code for setting
44          LDY  #>TOPMON   ;   the move parameters :
45          STY  A2L+1
46          LDY  #<CMDTBL   ;   Start = $D000 (ROM).
47          STY  A1L        ;   End   = $FFFF (Monitor).
48          STY  A4L        ;   Dest. = $D000 (RAM).
49          LDY  #<TOPMON
50          STY  A2L
51          LDY  #0         ; Initialize $FE2C by
52          JSR  MOVE       ;   setting Y-register = 0.
53
54
55  *-------------------------------------------------
56  *  2nd, move code for "GONAME" to the RAM Card.
57  *-------------------------------------------------
58
59
60  Again   LDY  #>GONAME   ; Another moving van:
61          STY  A1L+1
62          LDY  #>ZERO-1   ;   Start = $GONAME
63          STY  A2L+1      ;   End   = $ZERO-1
64          LDY  #>SHLOAD   ;   Dest. = $SHLOAD
65          STY  A4L+1
66          LDY  #<GONAME
67          STY  A1L
68          LDY  #<ZERO-1
69          STY  A2L
70          LDY  #<SHLOAD
71          STY  A4L
72          LDY  #0         ; Initialize Y-register
73          JSR  MOVE       ;   for another MOVE.
74  *
75  ROMoff  BIT  RAMrdWE2   ; Turn Motherboard ROM off
76          BIT  RAMrdWE2   ;   & write-enable RAM Card.
77
```

contd.

```
78
79  *-------------------------------------------------
80  *  Now, "magically" add your favorite Applesoft Command :
81  *
82  *    1. Find an existing Command, one that you really use
83  *       "alot", for example, SHLOAD.
84  *
85  *    2. To substitute your new Command for SHLOAD, choose
86  *       an equal-length name (6 characters in this case)
87  *       for the new Command, like GONAME.
88  *
89  *    3. Go to TOKTBL @ $D0D0 where are stored all the
90  *       Command text names.  We see that the names are
91  *       stored in DCI format, with 'SHLOAD' beginning
92  *       @ $D137.  Now place DCI 'GONAME' also
93  *       beginning @ $D137.
94  *
95  *    4. Go to CMDTBL @ $D000 where are stored all the
96  *       Command entry point addresses (less 1).  We
97  *       see that the entry point for SHLOAD is $F775,
98  *       which is stored (again, less 1) @ $D034.  By
99  *       examining the code beginning @ $F775, we note
100 *       note that SHLOAD encompasses 71 bytes,
101 *       $F775 -- $F7BB, big enough to hold GONAME.
102 *
103 *    5. Finally, start the actual code for GONAME in
104 *       low RAM ( moved earlier to the RAM Card ).
105 *       Strictly speaking, it's not necessary,
106 *       however, I chose to then fill the unused
107 *       portion of the above 71 bytes with HEX 00
108 *       at their new RAM Card location.  No big deal,
109 *       but it's more clean that way ( to once again
110 *       quote Sandy Mossberg -- "Just call me a
111 *       senile person!" ).
112 *
113 *-------------------------------------------------
114
115
116 ADDname  LDA  NEWname,Y  ; Y still 0 after MOVE.
117         BEQ  ADDzero    ; End of name.
118         STA  TOKTBL+$67,Y ; $D137 -> $D13C.
119         INY
120         BNE  ADDname    ; Always!!
121 *
122 ADDzero  LDY  #70
123         LDA  #$00
124 :1      STA  SHLOAD,Y   ; #$00 in GONAME+70 --> ZERO.
125         DEY
126         CPY  #ZERO-GONAME ; Changes automatically
127         BCS  :1           ;   with routine length.
128 *
129 toAPPLE  JMP  APPLE      ; Ret to Applesoft (on RAM Card).
130 *
131 NEWname  DCI  'GONAME'00 ; Token equivalent = $9A.
132
133
134 GONAME   EQU  *          ; Start of code for new command.
135
136
137         JSR  PTRGET     ; Get variable name; set variable
138 *                          flags & pointers in Page 0;
139 *                          determine if a new variable
140 *                          and make room if required;
141 *                          then, point to the address
142 *                          of the 1st byte of the value
143 *                          or, if an array, to the addr
144 *                          of the 1st byte of the
145 *                          array descriptor.
146         JSR  CHKNUM     ; Chk for #, or "Type Mismatch".
147         JSR  MOVFM      ; Rout. to get packed # at (Y,A),
148 *                          unpack it & move it to FAC.
149         JSR  GETADR     ; Convert # in FAC to an integer
150 *                          and then move it to LINNUM.
151         JMP  GOTO+3     ; Now we're ready to "GOTO".
152 ZERO     EQU  *
153         LST OFF
154         END
```

Oh . . By the Way contd. from pg 17

```
234 *                                                 *
235 *  Here we convert pseudo - 'BINARY' to real - BINARY. *
236 *  Stored in the Input Buffer is an array of $30's and *
237 *  $31's.  First, we perform an LSR on each Buffer    *
238 *  element.  If the resulting Carry flag is clear, we *
239 *  have a '0'; if set, then a '1'.  In short, the LSR *
240 *  instruction conditions the Carry flag for the      *
241 *  ensuing ROR (which, after shifting, places the     *
242 *  former contents of the Carry bit in Bit #7 of SHIFT *
243 *  or SHIFT+1 ).  Now, we've got to ROR 8 times for   *
244 *  each byte.  DIGITS and DIGITS+1 come into play when *
245 *  we've input < 8 'binary' bits per byte.  Now, wasn't*
246 *  that simple %?!*!?  Now, ON WITH THE SHOW!!!       *
247 *                                                 *
248 ***************************************************
249
250 CONVERT   DEX            ; # bits-1 = pos. in "INBUFF".
251         CPX LENless8    ; Only right-most 8 bits go to
252         BCS LObyte      ;   LObyte.  2 tests since the
253         LDA LENless8    ;   computer can't tell signed
254         BMI LObyte      ;   from unsigned numbers.
255 *
256 HIbyte    PMC shift/DIGITS+1;SHIFT+1
257         BEQ HEXout      ; Always!!
258 *
259 LObyte    PMC shift/DIGITS;SHIFT
260 *
261 HEXout    LDA #"$"        ; First display "$"
262         JSR COUT        ;   followed by HEX
263         LDX SHIFT       ;   designation.
264         LDA SHIFT+1
265         JSR PRNTAX
266         JSR CROUT
267 *
268 TOdec     LDA #"!"        ; Conversion to DECIMAL.
269         JSR COUT
270         LDA SHIFT+1     ; A,X registers clobbered
271         LDX SHIFT       ;   by "PRNTAX".
272         JMP LINPRT
273
274
275 ERROR     JMP SYNTAX
276
277
278 *-------------------------------------------------
279 *  Must end before $03D0 (Whew!!  Barely, huh!!).
280 *-------------------------------------------------
281         SKP 5
282         LST OFF
283         END
```

Oh . . By, By the Way contd. from pg 22

```
                127
                128 *-------------------------------------------
                129 *  Applesoft's "&" JuMPs to "AMPERV".
                130 *-------------------------------------------
                131
03F5: 4C 58 FF  132 AMPERV   JMP  RTS
                133
                134 *-------------------------------------------
                135 *  The Monitor's "CTRL-Y" JuMPs to ($03F9).
                136 *  While in the Monitor, "CTRL-Y" changed to
                137 *  "CTRL-C" that returns you to Applesoft.
                138 *-------------------------------------------
                139
03F8: 4C 65 FF  140 CTRLyV   JMP  MON
                141
                142 *-------------------------------------------
                143 *  Contains JuMP address to be called when a
                144 *  Non-Maskable Interrupt occurs.
                145 *-------------------------------------------
                146
03FB: 4C 65 FF  147 NMI      JMP  MON
                148
                149 *-------------------------------------------
                150 *  Address called by Monitor routine @ $FA40 when
                151 *  a Maskable Interrupt occurs.
                152 *-------------------------------------------
                153
03FE: 65 FF     154 IRQLOC   DA   MON

--End Assembly, 48 bytes, Errors: 0
```

Paragon Technologies, Inc.

offers classes in our IBM PC laboratory and Apple IIe laboratory in McLean:

- Getting Started With The Micro Computer
- VISICALC, LOTUS 1, 2, 3
- d Base II
- WordStar
- Programming in BASIC
- Computer assisted SAT preparation

All classes are hands-on with one person per computer.

Call
556-9659
or write to us:
P.O. Box 6128
McLean, Virginia 22106

MINUTE MANUALS

Minute Manual For PFS: FILE/REPORT/GRAPH/WRITE - Each program explained, for those who have one or more and those who want to find out about them. Also contains Quick Guide to over 50 procedures. Tutorial. Printer codes for Epson, Gemini, Apple, Prowriter, NEC, & Okidata. $12.95.

Minute Manual For Apple Writer //e - A lot of support for $7.95. Beginners guide to word processing, step by step instructions for the basic and advanced procedures. Over 60 pages of problem solving procedures on formatting and printing. Specific procedures for Epson, Gemini, Apple, Prowriter, NEC, Okidata.

Glossary Disk For Apple Writer //e and][+ - Easy to use - Automatically load glossary from menu. Select Apple Writer //e or][+. Select Epson FX/MX/RX, Gemini 10/10X, Prowriter, Apple DMP/Imagewriter, NEC 8023A, or Okidata 92/83A. Then turn on or off any print style with a single keystroke. $14.95.

Minute Manual For Apple Writer][+	$7.95
Minute Manual For DB Master	$12.95

Available at COMM CENTER, B. Dalton, Sidney Kramers (H St.), VF Assoc., Logical Choice, Towson Computer, Computers Unlimited, Westminster Computers & more, or send check + $1 S/H to Minuteware, P.O. Box 2392, Columbia, MD 21045 (301) 995-1166.

A UNIQUE IDEA!

HOLIDAY GREETING DISK, FOR FRIENDS AND FAMILY

Contains the spirit of the Holidays:

- *flickering candle*
- *gentle snowfall*
- *holiday carols*
- *personalized message*
- *and more*

Your friend's computer has never been so full of good cheer.

To send a holiday disk with friend's name included:
Send your friend's name and address, a greeting name for your friend, your name and address, a signature name for yourself, plus $5.00 to Love Phase Computing, P.O. Box 68 Boston, MA 02117

Orders should be post dated by Dec. 15.

—— Operant Systems ——

-- HARDWARE --

PRINTERS—
- Epson FX-80 (160 cps, tractor/single sheet, graphics).. 435
- FX-100 (wide carriage version of the above)..... 625
- RX-80F/T (100 cps tractor/single sheet, graphics) 305
- LQ-1500 (200 cps, fantastic letter-quality mode) 1095
- Fingerprint (pushbutton font selection for RX/FX).... 39/49
- Okidata 92 (160 cps, graphics, BEST print for price)... 395
- 93 (132 column version of above)..... 629
- Toshiba 1340 (144 cps draft, BEST letter-qual matrix).. 799
- Texas Instruments TI-855 (150 cps draft, 35 cps NLQ)... 795
- IDS Prism-80 (200 cps, color/sheet-feed options avail). 999
- Gemini 10X (120 cps, tractor/single sheet, graphics)... 265
- 15X (wide-carriage, same features as Epson MX).. 375
- Powertype (18 cps daisywheel)..... 335
- C.Itoh F-10 Starwriter (40 cps daisywheel, best for $) 1049
- A-10 (18 cps version of the above)..... (call)
- Silver-Reed 550 (18 cps daisywheel, best at this speed) 439
- 500 (14 cps daisywheel, as above)..... 375
- NEC 7710 (55 cps daisywheel, built like a tank)..... 1795
- Diablo 620 API (25 cps daisywheel)..... 839
- Qume Sprint 11/40 (40 cps daisywheel)..... 1295

MODEMS--
- Hayes Micromodem IIe (tone dialing/speaker/Smartcom I). 239
- Smartmodem 1200 (1200 baud, RS-232, auto-dial)... 469
- Smartmodem 300 (300 baud, as above)..... 199
- Novation Apple-Cat II (w/ Cowure; 1200 baud capable).. 235
- Smart-Cat 300 (RS-232, direct-connect)..... 175
- Smart-Cat 300/1200 (as above)..... 389
- Prometheus Promodem (300/1200 baud, RS-232, auto-dial). 365
- Microcom ERA 2 (300/1200 baud card with software)..... 359
- Zoom Telephonics Networker (300 baud card w/software).. 149
- US Robotics Password (300/1200 baud, auto-dial/answer). 335
- Anchor Automation Mark 12 (300/1200 baud, RS-232)..... 269
- Volksmodem (300 baud, RS-232)..... 59

DISK DRIVES—
- MicroSci A2 drive (100% Apple-compatible Shugart 390).. 199
- Rana Systems Elite One drive (40 track, 163K)..... 259
- Elite Two (40 track, double side, 326K)... 389
- Elite Three (80 track, double side, 652K). 449
- 10-MEG Winchester drive..... 1149
- TEAC Thinline drive (40 track, 100% Apple compatible).. 219
- Axlon 320K Ramdisk..... 829
- Davong, Corvus, and Corona Winchester drives.....(call)

CP/M & 6502C SYSTEMS--
- Applicard (6 Mhz Z-80, 64K to 192K RAM, 70-col video).. 249
- Microsoft Softcard IIe (Z-80, 80 col & 64K on one card) 265
- Softcard (includes CP/M 2.2 and MBASIC)..... 225
- DR Gold Card (6 Mhz, 80-col, 64-192K, CP/M 3.0, CBASIC) 259
- Titan Systems Accelerator IC (3.6 Mhz 6502C processor). 425
- Speed Demon (6502C high-speed coprocessor)..... 219

MONITORS—
- Amdek 300G (12" green anti-glare screen, 18Mhz)..... 125
- 300A (12" amber anti-glare screen, 18Mhz)..... 145
- NEC JB-1201/1205 (green/amber anti-glare screen, 20Mhz) 155
- JB-1260 (12" green, 15Mhz, best value for money)... 109
- USI PI-2 (12" green anti-glare screen, 20 Mhz)..... 125
- PI-3 (12" amber anti-glare screen, 20 Mhz)..... 125

INTERFACES & BUFFERS & CLOCKS—
- Pkaso/U printer interface (superior graphics & more !!) 129
- ShuffleBuffer (32K--128K w/cut & paste/ser and par).... 269
- Grappler+ printer interface (parallel w/ graphics)..... 109
- Buffered Grappler+ (16K to 64K buffer plus graphics)... 169
- CCS 7711 Super Serial (for printers & modems)..... 95
- Wizard IPI (graphics, text dump, page formatting)..... 69
- Practical Peripherals ProClock (ProDos compatible)..... 109

VIDEO & KEYBOARD—
- Videx Ultraterm (160 column/48 row display !!)..... 249
- Videoterm (80-column w/ softswitch & inverse).... 209
- MicroSci 80-col card w/64K RAM for IIc..... 125
- Keytronic KB-200 low-profile detached keyboard..... 235

MEMORY EXPANSION—
- Wespercorp Wizard 16K RAM card (2-yr warranty)..... 59
- Titan Systems 128K RAM card..... 349

-- SOFTWARE --

WORD PROCESSING—
- Wordstar 3.3 (including 6 Mhz Z-80 Applicard !!!).. 299
- ScreenWriter II (70 col display, spooling, more!!).. 85
- Super-Text Professional..... 115
- Bank Street Writer or Speller..... 45
- pfs: Write IIc or Proof IIc..... 85/49
- Pie Writer..... 99
- Word Handler + List Handler + Spell Handler..... 109
- HomeWord or HomeWord Speller..... 45/35
- Sensible Speller IV (checks ANY file type)..... 85
- The Word Plus (super spelling checker for CP/M).... 99

SPREADSHEETS & FORTUNE-TELLERS--
- Multiplan (state-of-the-art spreadsheet)..... 129
- FlashCalc (VisiCalc gets a facelift)..... 75
- SuperCalc 2 (powerful CP/M spreadsheet)..... 179

INFORMATION MANAGEMENT—
- dBASE II (w/ ZIP screen generator)..... 299
- pfs: File, Report, or Graph..... 85
- The General Manager 2.0..... 149
- Thinktank (electronic thought organizer)..... 99
- OB Master 4.0 (latest version)..... 225

BUSINESS & ACCOUNTING—
- Dollars & Sense (accounting + graphics).....69 & 79
- Peachtree Peachpak (GL/AR/AP)..... all three: 229
- Back to Basics Accounting (GL/AR/AP)..... 129
- BPI Accounting (AR/AP/PAYROLL/INVENTORY) ea module: 249

COMMUNICATIONS—
- Ascii Express Professional (for DOS 3.3)..... 89
- Z-term Professional (for CP/M)..... 99
- pfs:Access (basic communications for the IIe)..... 49
- CompuServe Starter Kit (password & 5 free hours)... 29

LANGUAGES--
- Microsoft TASC Applesoft compiler..... 119
- Softech UCSD p-System..... 425
- Microsoft Fortran-80..... 129
- BDS C compiler..... 115
- Micromotion FORTH-83..... 79
- Terrapin Logo..... 69

ODDS & ENDS—
- Kensington System Saver Fan (w/surge protector).... 66
- Kraft Joystick / TG Joystick / Hayes III..... 35/35/40
- KoalaPad Graphics Tablet (with graphics software).. 85
- Maxell, Verbatim, Dysan, Memorex, Wabash disks.. (call)

MACINTOSH
- Filevision (database system)..... 129
- Odesta Helix (another database system)..... 249
- Main Street Filer (yes...yet another database)..... 145
- Dollars & Sense..... 99
- Microsoft Word (what MacWrite should have been).... 129
- Chart..... 85
- Mac the Knife (canned doodles)..... 29
- T/Maker Click Art (...more doodles)..... 39
- Kensington MacModem (300 baud)..... 99
- Surge Protector (replaces power cord)... 39
- Tecmar, Davong hard disk systems..... (call)

CALL FOR PRICES OF ITEMS NOT LISTED

—— Please Compare Our Prices ——
If you find a lower price, give us a chance to beat it.

Feel free to call for answers to technical questions.

TO ORDER: Call Jeff Dillon at (301) 434-0405
or
Write or visit:

OPERANT SYSTEMS, 7676 New Hampshire Ave, Suite 312
Langley Park, Md 20783

Md. sales add 5% tax. UPS shipping is available.
All items carry full manufacturer's warranties.

A REALLY NIFTY UTILITY

by John A. Love III

Several times a year I am prone to dust off my old issues of Nibble to look for programming tips and/or full-fledged programs that will assist my current endeavors. This last time I came across "NEWSAVER" authored by Gary Bond (Volume 4, Number 5, 1983). "NEWSAVER" purported to be an all-purpose utility with the following options:

1. Restore the Applesoft file that maintained its existence in memory even after typing NEW or FP.

2. Determine the length of the Applesoft file currently in memory.

3. Restore on Disk a DELETEd file.

4. Determine the length and the starting address of any binary file on Disk without having to BLOAD it. The code at the end of this article focuses on the most recently BLOADed or BRUNned binary file.

5. Exit from the "NEWSAVER" utility.

This Nibble article purported to present the Machine code of this entire utility. But, hold on - this Machine code was only 83 bytes long. Now, wait just a minute!! Not all the above in just 83 bytes - oh well, I disassembled it anyway. Sure enough - these 83 bytes pertained only to Option #1. Must have been an editorial oversight. Now what??

I have maintained for a very long time that I do not learn very much using someone else's software. So, why not do it myself?? Believe it or not, I did - and here are the results.

The sources of knowledge, without which the task would have been impossible, deserve special note:

**** Sandy Mossberg's "Disassembly Lines" series appearing over the years in Nibble.

**** All About Applesoft from Call A.P.P.L.E., especially Val Golding's article "Applesoft from Bottom to Top".

**** All About DOS from Call A.P.P.L.E., especially Rick Sutcliffe's article "Rick's RELETER".

**** Beneath Apple DOS, the "bible" by Worth & Lechner.

Please note that I have awarded each their well-deserved and highest possible FOUR STAR rating!!

My entire code for "NIFTY" is presented at the end of this article. I will start at the beginning and explain not only the critical and essential code, but also the relatively unimportant but still "neat" code. By the way, the ORG Address is designed so that the last byte of my 1021 byte-long utility appears at $95FF, just under HIMEM ($9600) for a 48K system.

We START off by simply presenting on your monitor the Menu of five choices. I use a generalized PRINT routine that pegs on the first byte of the message to be presented on the screen. As a matter of fact, I use this same PRINT routine anytime I wish to print a message. Every message byte is presented on the screen via access to the Monitor routine, COUT, at $FDED. The very last byte of each message is always a HEX 00 so that this PRINT routine recognizes when the message is complete. In the case of the Menu I initialize the starting position of the Cursor via an equivalent VTAB and HTAB. VTAB is effected by storing the appropriate value in CV ($0025) and JSR-ing to the Monitor routine, VTAB at $FC22. HTAB is more simply effected by just storing the correct quantity in CH ($0024) prior to PRINTing. The reason that there is no need for an equivalent HTAB Monitor routine is because, when COUT is accessed, eventually we end up at STORADV in the Monitor at $FBF0 which prints the very first character at location CH. In short, COUT does the equivalent HTAB for us!!

After the Menu is presented, I INPUT the operator's selection by accessing Applesoft's main input routine at $D52C (see an earlier article in this Journal for a thorough discussion of $D52C). I then have to reset the Input Buffer ($0200--$02FF) pointer in order to tell the CHaRacter GETting routine ($00B1) where in the Input Buffer to start scanning for my input, in this case, the beginning. As you recall from the same earlier article, the code inherent to the routine at $D52C places your keypress(es) in the Input Buffer to await your retrieval. By the way, $D52C eventually clears the hi bit of the Buffer's contents - so that is why all of my comparisons are with positive ASCII. Had I simply accessed RDKEY at $FD0C to GET the keypress, my comparisons would be with negative ASCII.

If the operator's keypress is in the proper range, 1--5, and isn't 1--4, obviously the operator has selected to EXIT from "NIFTY", which is Option #5. In short, this fifth Option is treated as the default condition if no other Option applies.

It is appropriate to mention at this time that, in every case, after the selected Option has been implemented, "NIFTY" JuMPs to "Again?" on Line 372. This subroutine asks the operator whether he or she wishes to exercise ANOTHER Option. Notice here that I do access RDKEY, with my comparisons being with negative ASCII as mentioned above. If I press "No", the code returns to Applesoft. If I press "Yes", my code re-displays the same Menu and I start all over. If I mess up and press a different, and therefore meaningless, key I reprint the same question at the same position. I effect this by setting CH to 0 for the left margin and DECrement CV three times. I have to back-up three lines in order to exactly compensate for the fact that the first three bytes of the ANOTHER message (Line 436) are three <CR>'s. Otherwise, exact over-print does not occur.

Since Gary Bond has already written an excellent description of how to effect Option #1, I see no need to repeat his words here (Gary: I did alter your code slightly). I will simply state that if you accidentally type "NEW" or "FP", your Applesoft code is still in memory. However, two serious blemishes now prevent the operator from LISTing his program. First, some key pointers are messed up and, second, the starting address of the second program line in your code is missing. Gary simply resets these pointers and re-determines this starting address. Voila!! With that, I'm finished with describing Option #1.

Once you review Gary's article in Nibble, Option #2 becomes a snap. The length of any Applesoft program/file in memory is quantified by subtracting the beginning address pointed to by ($0067,0068) from the

contd.

ending address pointed to by ($00AF,00B0). I simply use the ROM Applesoft routine LINPRT at $ED24 to print the difference in DECIMAL. See, that was easy!!

I'll deliberately save Option #3 for last and jump to Option #4 that prints the length and the starting address of the most recently BLOADed or BRUNned file. Let's see - well, I know that said length is stored in $AA60,1 with the low byte in $AA60 and the high byte in $AA61. I might as well use LINPRT again to display the length in decimal. One down, one to go. I also know the starting address is stored in $AA72 and $AA73, low and high byte respectively. Since I probably wish to see the starting address in HEX, I'll just use the Monitor routine PRNTAX at $F941. Done!!

And, now, the BIGGEE - UNdeleting a DELETEd Disk file. Obviously any meaningful explanation must begin with a hopefully terse summary of the damage that DELETE does. Thank goodness it does not really delete, just as NEW or FP does not actually eradicate the Applesoft file from memory. The damage done, however, must be reversed to be back in business. Let's go for it!! Unfortunately, even before I can begin to discuss this damage, I need to discuss a little bit how a file is stored on Disk. Beneath Apple DOS to the rescue!!

Each Track on a 35-Track Diskette consists of 16 Sectors, for a total of 560 Sectors. DOS 3.3 reserves most of the 48 Sectors on Tracks 0,1,2 whether the Diskette be a "Master" or a "Slave". The Volume Table of Contents (VTOC) consumes Sector 0 of Track 17 ($11) for an "unprotected" Diskette and the actual Catalog Sectors of files (names, locations, etc.) consumes the remaining 15 Sectors of Track 17.

The VTOC Sector information is loaded into RAM when a Diskette is first booted. Starting at $B3BB are 256 bytes of dynamite. Among these are the Track & Sector (T/S) of the very first Catalog Sector. Got to know where to start, right?? In addition, we see what is called the Bit Maps of Free Sectors beginning at $B3F3. This Bit Map is 200 bytes long, but the last 60 are not used. Anyway, what this Bit Map shows is which of the 560 total Sectors are available. Four bytes of the "global" Bit Map are used to describe which of the 16 Sectors per Track are available or free. Let's see - 4 bytes to completely describe one Track and 35 Tracks makes 140 bytes total. Add 60 unused bytes and we end with 200 bytes. Yup - the arithmetic checks!! Okay, the VTOC at least tells us where to start looking for file info and what Sectors have some sort of data in them.

The first Catalog Sector starts on Sector 15, with the remaining consuming Sectors 14 --> 1. Remember the VTOC already occupies Sector 0. Each Catalog Sector also contains 256 bytes of dynamite. The first Catalog Sector, upon booting the DOS 3.3 System Master, starts at $B4BB. First, it contains a pointer to the next Catalog Sector. That sounds reasonable. Second, it contains what Worth & Lechner call a "File Descriptive Entry", each 35 bytes long, with a total of 7 file Entries per Catalog Sector. Oh - 7 files per Catalog Sector times 15 Sectors equals 105 total theoretically possible files per Disk (depending on length of course). So that's where that figure of "105" comes from!!

Each File Descriptive Entry points to the first T/S List Sector that, in turn, points to your file's content. Also the file type (binary, Applesoft, etc.) as well as the file name of course (30 bytes maximum) are part and parcel of the same Entry.

This whole chain seems like a rolling snowball. But we all know that it works - and that is the "bottom line". It seems to me that this "chaining" is part cause of the relative slowness of DOS 3.3 operations. But we'll save that subject for another article (No, not this issue).

What damage to any of the above does DELETE do?? Let's take a look! You type "DELETE NAME<cr>". Had "DELETE" been an Applesoft command, Applesoft's Main Command Loop at $D43C would immediately access $D52C which places these bytes in the Input Buffer as explained earlier. $D43C then sees that we have typed an immediate command, parses (tokenizes) it and then executes our command. The latter is via a JuMP to $D805. The ROM Applesoft parsing routine scans the contents of the Input Buffer and compares them with the contents of the Applesoft Token Table that starts at $D0D0. Once again, had "DELETE" been an Applesoft command, the word "DELETE" would have been found in this Table and DELETE's tabular position would have been calculated. $D805, "EXECUTE", takes this calculated offset and looks at the same position within the Applesoft Command Table beginning at $D000. In this manner, DELETE's Command Entry Point is defined. At this point, $D805 would execute DELETE by jumping to this Entry Point.

In short, as the Input Buffer is scanned, execution occurs immediately. In still other words, the ROM Applesoft (interpreter) executes as she goes - interpret, execute - interpret another line, execute some more - etc. Thus, we have one of the chief differences between an interpreter and a compiler. The latter "interprets" the WHOLE program at once BEFORE execution, not just one line at a time.

Obviously, however, DELETE is strictly a DOS command. Why, then, the discussion about executing Applesoft commands?? Simple!! There is an ALMOST exact parallel in the DOS command world. Also, we will see that Applesoft's routine at $D43C plays a pivotal role even in the DOS world. Please check out Sandy Mossberg's "Disassembly Lines" in Volume 4, Numbers 6-->8 of Nibble for a thoroughly enjoyable rendition of the processing of valid DOS commands.

I truly hope I don't do Sandy a disservice when I repeat just some of the many, many important points. To quote Sandy, "After booting a diskette, DOS imposes its presence by replacing the contents of the normal I/O hooks with its own I/O handlers, termed the keyboard (input) and video (output) intercepts." If DOS is active at the time, then the keyboard intercept, KSW, is $9E81 and the video intercept, CSW, is $9EBD. $9E81 eventually finds its way to $FD1B ("KEYIN") and $9EBD eventually finds its way to $FDF0 ("COUT1").

Applesoft's main input routine at $D52C (called by $D43C) is still calling the shots, in this case by routing its input via the input intercept vector at (KSW) and outputting via the output intercept vector at (CSW).

HOWEVER, DOS ALWAYS STANDS IN LINE AHEAD OF APPLESOFT.

The reason for this is that when the innards of $D52C process input/output beginning with $9E81, the Accumulator input from the keyboard at $C000 is looked at way before the Accumulator even finds $FDF0, the output to the screen monitor. DOS is notoriously "piggy" about this "me first" attitude. Well, let me tell you that when the input keypress is processed by $9E81 a "CSW State" number of 3 is assigned. Sandy labels this State "Process INPUT Statement". Processing now continues to the output intercept at $9EBD which jumps to the State 3 Output Handler. Here, if your keypress is not a <CR>, then $9EBD "echos" your keypress to the screen and control returns to the Applesoft routine at $D52C which handily stores said keypress in the Input Buffer. Hold my hand, folks, so I don't stray - I think I see the light!!

contd.

You deftly press <CR>. The <CR> is processed via $9EBD and DOS, "Mr. DOS" to you, blocks your path. DOS sees that you just typed an immediate command and knocks you to kingdom come over to CSW State 1. Sandy labels this State "Collect DOS Command". This CSW State routine stuffs this trailing <CR> into the Input Buffer and finally, finally JuMPs to DOS's parsing routine at $9FCD. This routine spots "DELETE" in the DOS Command Text Table that begins at $A884. A match has now been found between the contents of the Input Buffer and this Table. Personally I think it rather interesting that Applesoft stuffs all characters except the <CR> into the Input Buffer whereas DOS reserves the "stuffing" to himself when it comes to the <CR>. Anyway, a match has been found and DOS branches to CALCINDX at $A01B. At this point DOS sees that DELETE is allowed in immediate mode and subsequently checks for your valid file name. Found you!! Now a final JuMP to DELETE's Command Handler Entry point at $A263. Mr. DOS now wrecks bloody murder!!

So, it is true that even though the ROM Applesoft routine $D43C starts out in control, DOS erects a detour of himself between your actual input keypress and when that keypress shows itself on the screen monitor. DOS sure is a sneaky devil, isn't he??

Speaking of detours, I'm through this one and am ready to describe this havoc that DELETE imposes on your Diskette. Worth & Lechner on my right! Sandy on my left! Go for it!!

Upon entry to $A263 we see that the File Manager Command Handler is primed to jump to the DELETE Function Handler subroutine at $AD2B. The very first act that $AD2B executes is to locate and open your file. This locating process begins, in part, by accessing the subroutine at $B1C9, labelled "GETDIRFL". GETDIRFL reads the VTOC Sector into a separate buffer area.

> The 256-byte VTOC Sector, starting at $B3BB, contains the Track/Sector (T/S) location of the first Catalog Sector. Each of the 15 Catalog Sectors, in turn, house a maximum of 7 File Descriptive Entries. Finally, each File Entry contains 35 bytes of information among which is the NAME of the file. The first Catalog Sector begins at $B4BB.

Remember, you typed "DELETE NAME<cr>". So DOS has to locate your NAMEd file on Disk. How much more reasonable can DOS be!! It makes sense, then, that DOS scan all 15 Catalog Sectors, or all 105 File Descriptive Entries to look for your precious NAME on Disk.

This scanning process is effected by reading in the contents of each Catalog Sector via the subroutine at $B011. The Track # field in the first File Entry is looked at. If this Track # field at $B4C6 has its high bit set, then the entire File Entry is skipped and DOS moves to the next Entry by adding 35 to the File Descriptive Entry offset. This makes sense, doesn't it, because each Entry is 35 bytes long!! (Sandy labels this offset "DIRECIX".) Is it possible that when a file is DELETEd, this Track # field is filled with something > #$7F ?? Stay tuned!!

By this time DOS has found a File Descriptive Entry whose Track # field contains a byte <= #$7F. So, the next step is to scan the 30 bytes of the NAME field of the Disk file to look for a match to your input file NAME. Gotcha!!

Now, we return to the DELETE Function Handler. We now know where the Disk file is located on the Disk. Why?? Simple - we've kept track of the above-mentioned offset, DIRECIX. Mr. DOS sure is smart!!

After making certain that the Disk file of note is unlocked, we place a #$FF in the appropriate Track # field of the File Entry. Well, I guess #$FF qualifies as something > #$7F. So, this is why any DOS Operation skips a so-called DELETEd file. Said file is really still on Disk; it's just skipped when DOS looks for it after deletion.

Oh! Oh! DOS still inflicts more damage!!

- Copies the prior contents of this Track # field to the last byte of the file NAME field.
- Copies the prior contents of the Track # and Sector # fields over to the File Manager Work Area.

Damage almost complete!! Now DOS writes back all this partial damage to Disk. To continue . . . . .

- After reading in all the data of your to-be-DELETEd file from its T/S List Sector(s), each Sector is "freed" by accessing "FREESEC" at $AD89. This "free-ing" process entails placing a "1" in the Sector Bit Map of the F/M Work Area. This "1" will designate that Sector containing some of your file data as "free". And guess what DOS can do to a "free" Sector - the dirty rat can WRITE on it. Good grief!!
- Once your file data Sector(s) is "freed" in the F/M Work Area, then this Work Area Sector Allocation Map is mirrored to the VTOC Bit Map that I said started at $B3F3.

Now, Mr. DOS has completed his havoc!! Boy, what a mess!!

My execution of Option #3 starts on Line 196. The first thing I effect is to CATALOG just the DELETEd files. Well, it turns out that the CATALOG Function Handler at $AD98 also scans the Track # field of each File Descriptive Entry. Ordinarily, upon seeing a #$FF here, the CATALOG will skip the DELETEd file and continue on. However, if I simply change the byte that executes the skipping phenomenon from "BMI" to "BPL", then only DELETEd files will be CATALOGed. Gotta be sure to change it back again, though, as accomplished via Lines 204,205.

Next, I use Applesoft's Main Input routine at $D52C to input which DELETEd file I wish to "un"delete. Remember, "INLIN" stuffs positive ASCII into the Input Buffer. So, immediately after retrieval (via "CHRGET"), I must convert these contents to negative ASCII before I re-stuff them into the Primary File Name buffer beginning at $AA75. This re-stuffing procedure is required because it is in this Name buffer that all your DOS commands look for the file name. (Some commands like RENAME look for two, don't they!!). The comments associated with the remainder of the Option #3 code make the latter very easy to follow. At this point I would highly recommend that you study and re-study what I've said so far. To quote Sandy's infamous one-liner:

"I'LL WAIT ----- zzzzz"

Wait a minute, John, you've added some stuff I don't recognize!! You're absolutely right, so I'll concentrate just on these added features.

In another key article in All About DOS Rick Sutcliffe talks about "ROGRAM and Other DOS Traps". The problem focuses on a conflict in the use of STATUS, the Monitor's Status Register at $0048. It seems that both the Monitor and DOS use this location. The "bottom line" inherent to resolving this conflict entails dis-connecting DOS before accessing DOS' RWTS

contd.

routines and then re-connecting afterwards just before accessing the Monitor (e.g., $FDED im my PRINT routine). For goodness' sake, read Rick's article for a positively brilliant explanation of this phenomenon.

By the time the code on Line 300 rolls around, I've managed to repair the damage that DELETE effects on the file's Descriptive Entry. From this point onward I follow the DELETE Function Handler code, in some instances altering said code to "un"delete.

Rick Sutcliffe's first article (which I referenced at the very beginning) addresses "un"doing the Sector de-allocation of the file's T/S List Sector(s) -- you remember, those that contain the actual data of your binary or Applesoft file, for example. My routine "UNdelete" is yanked in total from DELETE's Function Handler with two key exceptions:

1) Changing from Sector de-allocation to re-allocation.

2) By-passing the normal exit from the File Manager, converting it to a simple "RTS". If I don't, then I return to the Monitor rather than to DOS. It turns out that this so-called "normal" exit fiddles with the stack, thus playing havoc with my return address.

After I re-connect DOS ala "ROGRAM and Other Traps" I display the re-generated CATALOG that "magically" shows my DELETEd file as UNdeleted. And, bingo, I'm done!!!

Whoops!!! One more thing - the program code:

```
:ASM

1   ********************************************************
2   *                                                      *
3   *                  A "NIFTY" Utility                   *
4   *                                                      *
5   *                     5  Options:                      *
6   *                                                      *
7   *  1 = Restore Applesoft file currently in memory      *
8   *  2 = Length of Applesoft file currently in memory    *
9   *  3 = Restore deleted Disk file                       *
10  *  4 = Length and starting Address of Binary file      *
11  *  5 = Exit                                            *
12  *                                                      *
13  *  Note: Can execute another Option following each.    *
14  *                                                      *
15  ********************************************************
16
17
18  * ----- Start of EQUates -----
19
20  BEGapPGM =   $0800      ; Start Applesoft pgm storage.
21  APPLE    =   $03D0      ; Vectored return to Applesoft.
22  APPlnPTR =   $06        ; $06 -> 09 unused by DOS,
23  MSGptr   =   $08        ;   the Monitor or Applesoft.
24  CH       =   $24        ; Horizontal cursor position.
25  CV       =   $25        ; Cursor row position.
26  TXTTAB   =   $67        ; Ptr - start of Applesoft pgm.
27  VARTAB   =   $69        ; Ptr - start of simple var.
28  ARYTAB   =   $6B        ; Ptr - start of array var.
29  STREND   =   $6D        ; Ptr - end of current  memory
30  *                       ;   or start of Free Space.
31  PRGEND   =   $AF        ; Ptr - end of Applesoft pgm.
32  DOStrue  =   $9EE0      ; Disconnects your DOS hooks.
33  IOINIT   =   $A851      ; Restore your DOS hooks.
34  BINlen   =   $AA60      ; Length of Binary file.
35  BINaddr  =   $AA72      ; Starting Addr of Binary file.
36  LINPRT   =   $ED24      ; Decimal print routine.
37  VTAB     =   $FC22      ; Set VTAB.
38  HOME     =   $FC58      ; HOME cursor.
39  COUT     =   $FDED      ; Output to monitor screen.
40  CROUT    =   $FD8E      ; Output <CR>.
41  PRNTAX   =   $F941      ; Outputs 2-byte HEX number.
42  RDKEY    =   $FD0C      ; "GET" your keypress.
43  INLIN    =   $D52C      ; Places INPUT in Input Buffer
44  CHRGET   =   $00B1      ;   for retrieval by "CHRGET".
45  TXTPTR   =   $00B8      ; Text pointer to Input Buffer.
46  STATUS   =   $48        ; Monitor Status register.
47  CATdel   =   $AD09      ; CATALOG deleted files only.
48  CATALOG  =   $A56E      ; CATALOG Command Handler.
49  PfnBUF   =   $AA75      ; Primary file name buffer.
50  BLKfnBUF =   $A095      ; Blank all file name buffers.
51  FITSTRK  =   $B4C6      ; T/S List Track for 1st file.
52  INITFMW  =   $ABDC      ; Zero entire F/M Work Area.
53  SETWAPRM =   $ABA6      ; Set FM Work Area parameters.
54  RDVTOC   =   $AFF7      ; Read VTOC.
55  RDDIRSEC =   $B011      ; Put Directory Sector buffer
56  *                       ;   address in RWTS parmlist.
57  DIRSECIX =   $B5D8      ; Offset to Directory Sector.
58  DIRECIX  =   $B39C      ; Offset to Directory file.
59  ADD35    =   $B230      ; Add 35 to file offset.
60  ALLOC    =   $B2F5      ; Reconcile Work Area Allocation
61  *                       ;   Map with VTOC Bit Map.
62  RETALLOC =   $AD86      ; By-pass FM exit.
63  UNdelete =   $AD50      ; "Un"delete the file.
64  SAVFMW   =   $AE7E      ; Save File Manager Work Area.
65  FREEbuf  =   $A2EA      ; Free DELETE buffer.
66
67  * ----- End of EQUates -----
68
69
70          ORG  $9203      ; Immediately under DOS.
71
72
73  * ----- Menu on monitor screen -----
74
9203: 20 58 FC  75  START   JSR  HOME       ; HOME.
9206: A9 07     76          LDA  #7         ; VTAB 7.
9208: 85 25     77          STA  CV
920A: 20 22 FC  78          JSR  VTAB
920D: A9 12     79          LDA  #18        ; HTAB 18.
920F: 85 24     80          STA  CH
9211: A0 10     81          LDY  #<MENU1    ; Start of Menu.
9213: A9 94     82          LDA  #>MENU1
9215: 20 FF 93  83          JSR  PRINT
9218: A9 10     84          LDA  #16        ; HTAB 16.
921A: 85 24     85          STA  CH
921C: A0 19     86          LDY  #<MENU2    ; Menu continues.
921E: A9 94     87          LDA  #>MENU2
9220: 20 FF 93  88          JSR  PRINT
                89  *
9223: 20 2C D5  90          JSR  INLIN      ; Your Keyboard INPUT.
9226: 86 B8     91          STX  TXTPTR     ; Reset Buffer Text pointer
9228: 84 B9     92          STY  TXTPTR+1   ;   in prep. for retrieval.
922A: 20 B1 00  93          JSR  CHRGET     ; Now, get your input.
922D: C9 36     94          CMP  #'6'       ; Error if not 1-5, so
922F: B0 D2     95          BCS  START      ;   re-display Menu.
9231: C9 31     96          CMP  #'1'
9233: 90 CE     97          BCC  START
                98  *
9235: F0 1F     99          BEQ  LIST       ; Execute your selection.
9237: C9 32    100          CMP  #'2'       ;   Note comparisons are
9239: F0 6F    101          BEQ  APPlen     ;   with positive ASCII.
923B: C9 33    102          CMP  #'3'
923D: D0 03    103          BNE  :1
923F: 4C CA 92 104          JMP  DELETE
9242: C9 34    105  :1      CMP  #'4'
9244: D0 03    106          BNE  :2
9246: 4C 9C 93 107          JMP  BINARY
9249: 20 C8 93 108  :2      JSR  CURSOR0    ; ----- Option #5 -----
924C: A0 D5    109          LDY  #<EXIT
924E: A9 95    110          LDA  #>EXIT
9250: 20 FF 93 111          JSR  PRINT
9253: 4C D2 93 112          JMP  Again?
               113
               114  * ----- Option #1 -----
               115
               116  * ------------------------------------------
               117  * Locate first byte of 2nd program line.
               118  * ------------------------------------------
9256: A2 04    119  LIST    LDX  #4         ; 1st byte in 1st line starts
9258: E8       120  :3      INX             ;   @ $0805.  1st byte in 2nd
9259: BD 00 08 121          LDA  BEGapPGM,X ;   line starts after next #$00.
925C: D0 FA    122          BNE  :3
925E: E8       123          INX             ; Skip this next HEX 00.
925F: A9 08    124          LDA  #8         ; Store "$08XX" in $0802,1.
9261: 8D 02 08 125          STA  BEGapPGM+2
9264: 8E 01 08 126          STX  BEGapPGM+1
               127  * ------------------------------------------
               128  * Store "$0801" in $07,06 to serve as a Pointer.
               129  * ------------------------------------------
9267: A9 08    130          LDA  #8
9269: 85 07    131          STA  APPlnPTR+1
926B: A9 01    132          LDA  #1
926D: 85 06    133          STA  APPlnPTR
               134  * ------------------------------------------
               135  * Looks for 3 successive HEX 00's ==> EOF.
               136  * ------------------------------------------
926F: A2 00    137          LDX  #0
9271: A0 00    138  :4      LDY  #0         ; Counts HEX 00's.
9273: 20 A3 92 139  :5      JSR  PLUS1
9276: A1 06    140          LDA  (APPlnPTR,X) ; (X) = 0.
9278: D0 F7    141          BNE  :4
927A: C8       142          INY
927B: C0 03    143          CPY  #3
927D: D0 F4    144          BNE  :5
               145  * ------------------------------------------
               146  * Then, set all Pointers.
               147  * ------------------------------------------
927F: 20 A3 92 148          JSR  PLUS1      ; 1st byte after 3 HEX 00's.
9282: A5 07    149          LDA  APPlnPTR+1
9284: 85 B0    150          STA  PRGEND+1
9286: 85 6A    151          STA  VARTAB+1
9288: 85 6C    152          STA  ARYTAB+1
928A: 85 6E    153          STA  STREND+1
               154  *
928C: A5 06    155          LDA  APPlnPTR
928E: 85 AF    156          STA  PRGEND
9290: 85 69    157          STA  VARTAB
9292: 85 6B    158          STA  ARYTAB
9294: 85 6D    159          STA  STREND
               160  * ------------------------------------------
               161  * Finished!!!
               162  * ------------------------------------------
9296: 20 C8 93 163          JSR  CURSOR0
9299: A0 D6    164          LDY  #<RELIST   ; Message.
929B: A9 94    165          LDA  #>RELIST
929D: 20 FF 93 166          JSR  PRINT
```

```
92A0: 4C D2 93  167          JMP  Again?
                168  * ---------------------------------------
                169  * Routine to INCrement APPlnPTR.
                170  * ---------------------------------------
92A3: E6 06     171  PLUS1    INC  APPlnPTR   ; Two-byte INCrement.
92A5: D0 02     172          BNE  RET
92A7: E6 07     173          INC  APPlnPTR+1
92A9: 60        174  RET      RTS
                175
                176  * ----- Option #2 -----
                177
92AA: 20 C8 93  178  APPlen   JSR  CURSORO
92AD: A0 EA     179          LDY  #<APPlong  ; Message.
92AF: A9 94     180          LDA  #>APPlong
92B1: 20 FF 93  181          JSR  PRINT
                182  *
92B4: A5 AF     183          LDA  PRGEND     ; 2-byte subtraction with:
92B6: 38        184          SEC             ;   (X) = low byte &
92B7: E5 67     185          SBC  TXTTAB     ;   (A) = high byte
92B9: AA        186          TAX             ;   to prepare for "LINPRT".
92BA: A5 B0     187          LDA  PRGEND+1
92BC: E5 68     188          SBC  TXTTAB+1
92BE: 20 24 ED  189          JSR  LINPRT     ; Print length in decimal.
92C1: 20 8E FD  190          JSR  CROUT
92C4: 20 8E FD  191          JSR  CROUT
92C7: 4C D2 93  192          JMP  Again?
                193
                194  * ----- Option #3 -----
                195
92CA: A9 10     196  DELETE   LDA  #$10       ; Change "BMI" to "BPL" to
92CC: 8D D9 AD  197          STA  CATdel     ;   CATALOG deleted file(s).
92CF: 20 C8 93  198          JSR  CURSORO
92D2: A0 0D     199          LDY  #<DEL1     ; Message.
92D4: A9 95     200          LDA  #>DEL1
92D6: 20 FF 93  201          JSR  PRINT
92D9: 20 6E A5  202          JSR  CATALOG    ; Just DELETEd file(s).
                203  *
92DC: A9 30     204          LDA  #$30       ; Change back to a
92DE: 8D D9 AD  205          STA  CATdel     ;   normal CATALOG.
                206  *
92E1: A0 30     207          LDY  #<DEL2     ; Another message.
92E3: A9 95     208          LDA  #>DEL2
92E5: 20 FF 93  209          JSR  PRINT
                210  * ---------------------------------------
                211  * Input File Name for "Un"deletion.
                212  * ---------------------------------------
92E8: 20 95 A0  213          JSR  BLKfnBUF   ; Blank all file name buffers.
                214  *
92EB: 20 2C D5  215          JSR  INLIN      ; INPUT your file name.
92EE: 86 B8     216          STX  TXTPTR     ; Reset Buffer Text pointer
92F0: 84 B9     217          STY  TXTPTR+1   ;   in prep. for retrieval.
92F2: A2 00     218          LDX  #0
92F4: 20 B1 00  219  :6       JSR  CHRGET     ; Scan Input Buffer.
92F7: F0 08     220          BEQ  :7         ;   HEX 00 ends INPUT.
92F9: 09 80     221          ORA  #$80       ; Convert to negative ASCII.
92FB: 9D 75 AA  222          STA  PfnBUF,X   ; Store name of your file.
92FE: E8        223          INX
92FF: D0 F3     224          BNE  :6         ; Always!!
                225  :7       EQU  *
                226  * ---------------------------------------
                227  * Okay, now we've got our file name tucked away.
                228  * Next, we've got to re-trace some ground covered
                229  * by the CATALOG Function Handler @ $AD98.
                230  * ---------------------------------------
9301: 20 E0 9E  231          JSR  DOStrue    ; "ROGRAM & Other DOS Traps".
                232  *
9304: 20 DC AB  233          JSR  INITFMW    ; Zero-out FM Work Area.
9307: 20 F7 AF  234          JSR  RDVTOC     ; Read VTOC into buffer.
930A: A9 00     235          LDA  #0         ; Initialize for 1st Sector.
930C: 8D D8 B5  236          STA  DIRSECIX
930F: 18        237          CLC
                238  *
9310: EE D8 B5  239  :8       INC  DIRSECIX   ; Read Directory Sector
9313: 20 11 B0  240          JSR  RDDIRSEC   ;   into the buffer.
9316: 90 11     241          BCC  GETTRK
                242  *
9318: 20 51 A8  243  :9       JSR  IOINIT     ; Re-connect DOS hooks.
931B: A9 00     244          LDA  #0         ; Reset Monitor
931D: 85 48     245          STA  STATUS     ;   STATUS register.
931F: A0 8A     246          LDY  #<UNFOUND  ; Exit FM because
9321: A9 95     247          LDA  #>UNFOUND  ;   file not found!!
9323: 20 FF 93  248          JSR  PRINT
9326: 4C D2 93  249          JMP  Again?
                250  *
9329: A2 00     251  GETTRK   LDX  #0         ; Evaluate Track number.
932B: 8E 9C B3  252  :10      STX  DIRECIX    ; (X) = File offset.
932E: BD C6 B4  253          LDA  FITSTRK,X
9331: F0 E5     254          BEQ  :9         ; End of entries.
9333: 30 07     255          BMI  COMPfn     ; A DELETEd file has #$FF.
                256  *
9335: 20 30 B2  257  SKIPfile JSR  ADD35      ; Not a DELETEd file, so
9338: 90 F1     258          BCC  :10        ;   skip it & finish Sector.
933A: B0 D4     259          BCS  :8         ; Finished with Sector.
                260  * ---------------------------------------
                261  * Upon entry to the DELETE Function Handler @
                262  * $AD2B, the track # in the File's Descriptive
                263  * Entry is copied to the last byte of the File
                264  * Name field.  Therefore, skip this name byte
                265  * when doing the next step -- comparing your
                266  * INPUT file name and the name in the File
                267  * Descriptive Entry of a DELETEd disk file.
                268  * ---------------------------------------
933C: A0 FF     269  COMPfn   LDY  #-1         ; Index for your INPUT name.
933E: E8        270          INX             ; Disk name begins with
933F: E8        271          INX             ;   4th byte from start of
9340: E8        272  :11      INX             ;   File Descriptive Entry.
9341: C8        273          INY
9342: BD C6 B4  274          LDA  FITSTRK,X
9345: D9 75 AA  275          CMP  PfnBUF,Y
9348: D0 E8     276          BNE  SKIPfile
934A: C0 1C     277          CPY  #28        ; Skip 30th byte in name.
934C: 90 F2     278          BCC  :11
                279  * ---------------------------------------
                280  * At this point we've got a successful match
                281  * between your INPUT file name and the name on
                282  * disk  (Bytes 0 --> 28).
                283  *
                284  * Now what??? I hope the suspense is unbearable!!!
                285  *
                286  * Upon DELETE, we've seen that the track # in
                287  * the file's Descriptive Entry is copied to the
                288  * last byte of the File Name field.  Also, the
                289  * track byte is replaced by #$FF.  According to
                290  * Sandy Mossberg, any routine that "un"deletes
                291  * a file needs to transpose these bytes again.
                292  *
                293  * So, folks, let's not dilly dally!!!
                294  * ---------------------------------------
934E: AE 9C B3  295          LDX  DIRECIX    ; Load back File offset.
9351: BQ E6 B4  296          LDA  FITSTRK+32,X ; Restore Track # of
9354: 9D C6 B4  297          STA  FITSTRK,X  ;   file's T/S List.
9357: A0 1D     298          LDY  #29        ; Restore last byte of
9359: B9 75 AA  299          LDA  PfnBUF,Y   ;   filename in Catalog
935C: 9D E6 B4  300          STA  FITSTRK+32,X ;   Sector buffer.
                301  * ---------------------------------------
                302  * With the damage to the filename (in the file's
                303  * T/S List) repaired, now set FM Work Area
                304  * parameters.  Follow this by reading the first
                305  * T/S List Sector to the file buffer.
                306  * ---------------------------------------
935F: 20.A6 AB  307          JSR  SETWAPRM
                308  * ---------------------------------------
                309  * Finally, write both the Directory Sector buffer
                310  * and the VTOC Sector buffer back to diskette.
                311  * ---------------------------------------
9362: A9 59     312          LDA  #$59       ; Re-allocate all the
9364: 8D F5 B2  313          STA  ALLOC      ;   de-allocated Sectors.
9367: A9 60     314          LDA  #$60
9369: 8D 86 AD  315          STA  RETALLOC
                316  *
936C: 20 50 AD  317          JSR  UNdelete
936F: 20 7E AE  318          JSR  SAVFMW
                319  *
9372: A9 19     320          LDA  #$19       ; Return code to
9374: 8D F5 B2  321          STA  ALLOC      ;   original state.
9377: A9 4C     322          LDA  #$4C
9379: 8D 86 AD  323          STA  RETALLOC
                324  *
937C: 20 EA A2  325          JSR  FREEbuf    ; Free buff used by DELETE.
                326  * ---------------------------------------
                327  * Display new CATALOG with DELETEd file present.
                328  * ---------------------------------------
937F: 20 51 A8  329          JSR  IOINIT     ; First, reconnect DOS &
9382: A9 00     330          LDA  #0         ;   reset the Monitor
9384: 85 48     331          STA  STATUS     ;   STATUS register.
                332  *
9386: 20 C8 93  333          JSR  CURSORO
9389: A0 4C     334          LDY  #<DEL3     ; Final message.
938B: A9 95     335          LDA  #>DEL3
938D: 20 FF 93  336          JSR  PRINT
9390: 20 0C FD  337          JSR  RDKEY
9393: 20 6E A5  338          JSR  CATALOG
9396: 20 8E FD  339          JSR  CROUT
9399: 4C D2 93  340          JMP  Again?
                341
                342  * ----- Option #4 -----
                343
939C: 20 C8 93  344  BINARY   JSR  CURSORO
939F: A0 A1     345          LDY  #<BIN1     ; Message in 2 parts.
93A1: A9 95     346          LDA  #>BIN1
93A3: 20 FF 93  347          JSR  PRINT
                348  *
93A6: AE 60 AA  349          LDX  BINlen     ; First the length and then
93A9: AD 61 AA  350          LDA  BINlen+1   ;   the starting address of
93AC: 20 24 ED  351          JSR  LINPRT     ;   most recently BLOADed file.
                352  *
93AF: A0 BE     353          LDY  #<BIN2     ; Length in DECIMAL &
93B1: A9 95     354          LDA  #>BIN2     ;   address in HEX.
93B3: 20 FF 93  355          JSR  PRINT
93B6: AE 72 AA  356          LDX  BINaddr    ; For each case :
93B9: AD 73 AA  357          LDA  BINaddr+1  ;   (X) = low byte &
93BC: 20 41 F9  358          JSR  PRNTAX     ;   (A) = high byte.
93BF: 20 8E FD  359          JSR  CROUT
93C2: 20 8E FD  360          JSR  CROUT
93C5: 4C D2 93  361          JMP  Again?
                362
                363  * ----- Routine for starting position of Cursor -----
                364
93C8: 20 58 FC  365  CURSORO  JSR  HOME       ; HOME.
93CB: A9 0C     366          LDA  #12        ; VTAB 12.
93CD: 85 25     367          STA  CV
93CF: 4C 22 FC  368          JMP  VTAB
                369
                370  * ----- Routine for selecting another Option -----
                371
93D2: A0 DF     372  Again?   LDY  #<ANOTHER
93D4: A9 95     373          LDA  #>ANOTHER
93D6: 20 FF 93  374          JSR  PRINT
93D9: 20 0C FD  375          JSR  RDKEY      ; GET response.
93DC: C9 CE     376          CMP  #"N"
93DE: F0 16     377          BEQ  OUT
93E0: C9 D9     378          CMP  #"Y"
93E2: F0 0F     379          BEQ  BEGIN
                380  * ---------------------------------------
                381  * Wrong keypress.  Print same message at same spot.
                382  * ---------------------------------------
93E4: C6 25     383          DEC  CV         ; Back-up 3 lines.
```

```
93E6: C6 25     384          DEC  CV
93E8: C6 25     385          DEC  CV
93EA: 20 22 FC  386          JSR  VTAB
93ED: A9 00     387          LDA  #0          ; HTAB 0
93EF: 85 24     388          STA  CH
93F1: F0 DF     389          BEQ  Again?      ; Always!!
                390  * ---------------------------------------
93F3: 4C 03 92  391  BEGIN   JMP  START       ; Re-display Menu & start over!
93F6: 20 8E FD  392  OUT     JSR  CROUT
93F9: 20 8E FD  393          JSR  CROUT
93FC: 4C 00 03  394          JMP  APPLE       ; Back to Applesoft!!
                395
                396  * ----- Routine to print message on monitor screen -----
                397
93FF: 84 08     398  PRINT   STY  MSGptr
9401: 85 09     399          STA  MSGptr+1
9403: A0 00     400          LDY  #0
9405: B1 08     401  :12     LDA  (MSGptr),Y
9407: F0 06     402          BEQ  :13
9409: 20 ED FD  403          JSR  COUT        ; Places 1st character @ CH.
940C: C8        404          INY
940D: D0 F6     405          BNE  :12         ; Always!!
940F: 60        406  :13     RTS
                407
                408  * ----- Various Messages -----
                409
                410
                411  * Note : $8D = <CR> & $87 = CTRL-G.
                412
9410: 0E 09 06  413  MENU1   INV  "NIFTY"8D,8D,8D,00
9413: 14 19 8D 8D 8D 00
9419: 85 A0 CF  414  MENU2   ASC  "5 OPTIONS:"8D,8D,8D
941C: D0 D4 C9 CF CE D3 BA 8D
9424: 8D 8D
9426: B1 A0 BD  415          ASC  "1 = RESTORE APPLESOFT PGM IN MEMORY"8D
9429: A0 D2 C5 D3 D4 CF D2 C5
9431: A0 C1 D0 D0 CC C5 D3 CF
9439: C6 D4 A0 D0 C7 CD A0 C9
9441: CE A0 CD C5 CD CF D2 D9
9449: 8D
944A: B2 A0 BD  416          ASC  "2 = LENGTH OF APPLESOFT PGM IN MEMORY"8D
944D: A0 CC C5 CE C7 D4 C8 A0
9455: CF C6 A0 C1 D0 D0 CC C5
945D: D3 CF C6 D4 A0 D0 C7 CD
9465: A0 C9 CE A0 CD C5 CD CF
946D: D2 D9 8D
9470: B3 A0 BD  417          ASC  "3 = RESTORE DELETED DISK FILE"8D
9473: A0 D2 C5 D3 D4 CF D2 C5
947B: A0 C4 C5 CC C5 D4 C5 C4
9483: A0 C4 C9 D3 CB A0 C6 C9
948B: CC C5 8D
948E: B4 A0 BD  418          ASC  "4 = LENGTH & START ADDR OF BINARY FILE"8D
9491: A0 CC C5 CE C7 D4 C8 A0
9499: A6 A0 D3 D4 C1 D2 D4 A0
94A1: C1 C4 C4 D2 A0 CF C6 A0
94A9: C2 C9 CE C1 D2 D9 A0 C6
94B1: C9 CC C5 8D
94B5: B5 A0 BD  419          ASC  "5 = EXIT"8D,8D,8D
94B8: A0 C5 D8 C9 D4 8D 8D 8D
94C0: D9 CF D5  420          ASC  "YOUR SELECTION?? "87,87,87,87,00
94C3: D2 A0 D3 C5 CC C5 C3 D4
94CB: C9 CF CE BF BF A0 87 87
94D3: 87 87 00
94D6: D2 C5 C1  421  RELIST  ASC  "READY TO RELIST!!"8D,8D,00
94D9: C4 D9 A0 D4 CF A0 D2 C5
94E1: CC C9 D3 D4 A1 A1 8D 8D
94E9: 00
94EA: CC C5 CE  422  APPlong ASC  "LENGTH OF APPLESOFT PGM (BYTES) = "00
94ED: C7 D4 C8 A0 CF C6 A0 C1
94F5: D0 D0 CC C5 D3 CF C6 D4
94FD: A0 D0 C7 CD A0 A8 C2 D9
9505: D4 C5 D3 A9 A0 BD A0 00
950D: C3 C1 D4  423  DEL1    ASC  "CATALOG OF DELETED FILE(S) :"8D,8D
9510: C1 CC CF C7 A0 CF C6 A0
9518: C4 C5 CC C5 D4 C5 C4 A0
9520: C6 C9 CC C5 A8 D3 A9 A0
9528: BA 8D 8D
952B: 87 87 87  424          HEX  87,87,87,87,00
952E: 87 00
9530: 8D 8D 8D  425  DEL2    HEX  8D,8D,8D
9533: C6 C9 CC  426          ASC  "FILE TO BE UN-DELETED : "00
9536: C5 A0 D4 CF A0 C2 C5 A0
953E: D5 CE AD C4 C5 CC C5 D4
9546: C5 C4 A0 BA A0 00
954C: C4 C9 D3  427  DEL3    ASC  "DISK FILE NOW UNDELETED!!"8D,8D
954F: CB A0 C6 C9 CC C5 A0 CE
9557: CF D7 A0 D5 CE C4 C5 CC
955F: C5 D4 C5 C4 A1 A1 8D 8D
9567: 87 87 87  428          HEX  87,87,87,87
956A: 87
956B: D0 D2 C5  429          ASC  "PRESS <CR> FOR NEW CATALOG :"8D,8D,00
956E: D3 D3 A0 BC C3 D2 BE A0
9576: C6 CF D2 A0 CE C5 D7 A0
957E: C3 C1 D4 C1 CC CF C7 A0
9586: BA 8D 8D 00
958A: 8D 8D     430  UNFOUND HEX  8D,8D
958C: C3 C1 CE  431          ASC  "CANNOT FIND FILE!!"8D,8D,00
958F: CE CF D4 A0 C6 C9 CE C4
9597: A0 C6 C9 CC C5 A1 A1 8D
959F: 8D 00
95A1: CC C5 CE  432  BIN1    ASC  "LENGTH IN BYTES (DECIMAL) = "00
95A4: C7 D4 C8 A0 C9 CE A0 C2
95AC: D9 D4 C5 D3 A0 A8 C4 C5
95B4: C3 C9 CD C1 CC A9 A0 8D
95BC: A0 00
95BE: 8D 8D     433  BIN 2   HEX  8D,8D
95C0: D3 D4 C1  434          ASC  "STARTING ADDRESS = $"00
95C3: D2 D4 C9 CE C7 A0 C1 C4
```


FREE SHIPPING

PRINTER CLEARANCE
FREE PRINTER I/F
CARD WITH PURCHASE
OF EPSON PRINTER

EPSON MX 80 F/T PRINTER	359
EPSON FX 80 PRINTER	459
MITAC DISK DRIVE (100% APPLE COMPATIBLE, SHUGART 390)	159
MITAC DISK DRIVE CONTROLLER	45
16K RAM CARD	40
Z-80A CARD	70
PROFESSIONAL JOYSTICK(AUTO CENTERING) FOR APPLE	19
REGULAR JOYSTICK FOR APPLE	12
SYNCO 12" AMBER MONITOR (18 MHZ)	90
VERBATIM DATALIFE SS/DD DISK (10 PACKS IN SOFT BOX)	20
DIABLO 620 PRINTER(FLOOR SAMPLE)	CALL
MEMORY CHIPS FOR IBM (4164, 200 NS, SET OF 9)	44

MICRO STAR COMPANY
P. O. BOX 2307
COLUMBIA, MD 21045
(301) 730-7172

Terms: Add $1.00 handling fee per order. MD residents add 5% tax. Personal or company checks allow 2 weeks to clear. COD accepted by M.O. or cashier check only(add $1.65 COD charge).Prices are subject to change without notice. Q'ty is subject to availability. MC & VISA customers add 3.5% surcharge. Manufacturer or MSC warranty provided.

contd. on pg 26

FORTH FLOATING POINT ARITHMETIC USING BCD NUMBERS

by Chester H. Page

A floating-point package is a welcome addition to Forth; it is even more welcome when decimal numbers are not changed to strings of 9's, or 0's followed by a 1, as is often done in Applesoft. This familiar annoyance is caused by the conversion of decimal fractions to binary fractions, with later reconversion. This conversion problem is avoided by performing all calculations in binary-coded decimal (BCD) format.

A byte holds two hexadecimal characters, 0 to F. If we use only 0 to 9 in each character position a byte can be a pictorial representation of decimal numbers from 0 to 99. This representation of numbers is called BCD. To perform addition, the CPU must set its carry bit when 9 is exceeded, rather than when F is exceeded. Fortunately, the 6502 CPU in the Apple][has this "decimal mode" available. To perform addition of two binary-coded numbers, the numbers may be "pictured" as ordinary decimal numbers; that is, the coded digits must appear in the same order as the ordinary written form of the number. Thus, if we have a ten-digit number, normally written as the sequence D1 D2 D3 D4 D5 D6 D7 D8 D9 D10, we want the coded form to "look" the same in its memory placement for arithmetic operations. In particular, I shall store a ten-digit number in memory locations 1 through 5, reserving address 0 for a sign byte. Negative numbers are stored, as usual, as complements, but complemented with 10's rather than 2's. These complements are formed by subtracting each coded byte for the (absolute value of the) number from $99. Since there is no decimal digit greater than 9, there is no question of needing a borrow in the subtraction. After all the bytes have been subtracted from $99, 1 is added to the least significant digit, with a possible carry propogating to the left. This converts the 9's complement to a 10's complement.

With 00 in the sign byte of a positive number, a negative number has $99 in its sign byte. A little experimenting shows that adding two numbers will yield a sign byte of 00, 01, 99, or 98 depending on the sign of the sum and on whether or not there is a "real" carry in the addition.

The mantissa of each number will thus be represented by 6 bytes (3 words). In memory, from lower address to higher address, we find

```
/sign/D1 D2/D3 D4/D5 D6/D7 D8/D9 D10/
```

When these data are fetched as words for the stack, the lower-address byte of a word becomes the higher-on-the-stack byte. Thus the first two bytes go on the stack as /D1 D2/sign/. To keep the sign byte from being buried in the middle of a floating-point number as held on the stack, we choose this word as the top word. The number on the stack becomes

```
W1 W2 W3  (W3 on top)
```

where

```
     W1                   W2                   W3

/D9 D10/D7 D8/       /D5 D6/D3 D4/        /D1 D2/SIGN/
```

For the complete floating-point number, an exponent must be placed on top of these mantissa words. The exponent word is

```
W4
/E1 E2/sign/
```

where E1 E2 is the "picture" of a decimal number from 0 to 99.

Note that when a word is fetched from the stack, its bytes do not interchange as they do going to or from storage, hence the sign is the low-byte of the word.

We need successive locations for three ten-digit numbers (with signs) : 0 to 5 for the first number (3 words), 6 to $B for the next number, and $C to $11 for the third. The next two bytes at $12/13 hold one (signed) exponent; $14/15 hold another. The first number is variously the ADDEND, SUM, DIVIDEND, and REMAINDER. The second holds the MULTIPLIER or the QUOTIENT, and the third is the ADDER, MULTIPLICAND or DIVISOR.

Before a floating-point number is collected from its 0/5 location in memory, TNORM shifts it to the left, one digit at a time, until the first digit is not zero. The exponent is adjusted with each shift.

Floating-point numbers are displayed with one digit preceding the decimal point, followed by an "E" and the value of the exponent. When the sign byte in address 0 is a 1 or $98, the arithmetic has produced a carry into the second digit before the point; the picture-sequence of digits must be shifted one digit to the right and the E-value increased by 1. Then the sign is changed to 00 or 99, respectively. The correct decimal point placement is adjusted by TNORM.

In the case of division, the first digit of the quotient can be a 0, the second cannot. When the first digit is a zero, TNORM produces a left-shift resulting in a fill-in 0 as the tenth digit. This situation is avoided by carrying out the division to the 11th digit in the quotient, followed by a right-shift if the first digit is not 0.

For addition of floating-point numbers, the number having the lesser exponent must be "denormalized" by shifting to the right until the greater exponent is appropriate. This "aligns the mantissas". The addition can now be performed, and the sum renormalized to allow for having had a digit carry.

The basic operations of shifting one digit (half a byte!) at a time, adding, complementing, and dividing are written in machine code and entered using CREATE. The remainder of the program is conventional FORTH.

Numbers are entered in floating-point format by entering the number in the usual way, but including a decimal point in the desired position (e.g., 1.23, .2, 4.) or by using "scientific" notation, i.e., no decimal point, but follow the number with E and the exponent. The + sign is not used but negative numbers start with -, and negative exponents have a - immediately after the E. The allowed exponent range is -99 to 99, and the number (mantissa) may not contain more than 10 digits. Numbers are displayed (by F.) as one digit, point, 9 digits, and exponent.

Floating-point numbers can be entered as above in either "executing" or "compiling" mode.

Operating words are analogs of the usual words: F! F@

contd.

```
FVARIABLE  FCONSTANT FDROP FDUP FSWAP F+ F* F- F/ etc.
```

Built-in polynomials, such as the approximation of SINE, are evaluated by a DO-LOOP of the form

```
: POLY ( FVAR---FRESULT) F0 -1 5 DO FOVER F* I  TABLE
F@ F+ -1
+LOOP FSWAP FDROP ;
```

which evaluates the polynomial

```
C0 + C1*X + C2*X^2 + C3*X^3 + C4*X^4 + C5*X^5
```

using the top-down routine

```
X(X(X(X(X(0 + C5) + C4) + C3) + C2) + C1) + C0.
```

The coefficients, each a four-word floating-point constant, are stored in a vector table created by

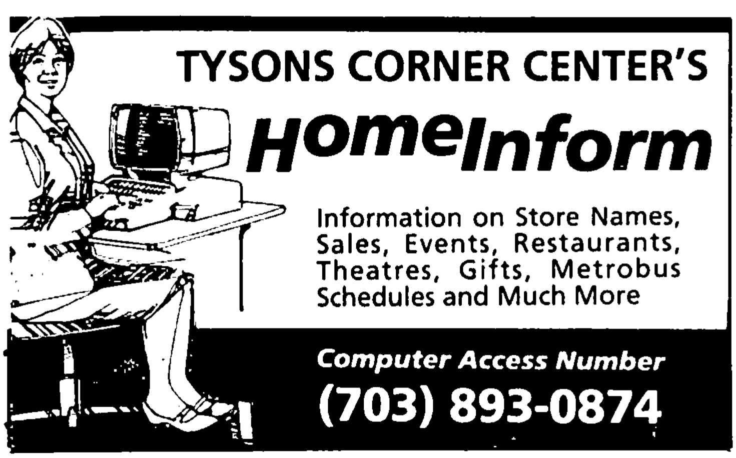
```
: FVECTOR (BUILDS 1+ 8 * ALLOT DOES> SWAP 8 * + ;

5 FVECTOR TABLE
C0 0 TABLE F!
C1 1 TABLE F!
C2 2 TABLE F!
etc.
```

where the actual values of the coefficients are entered (not the names!)

SINE is evaluated by using the 11th degree polynomial given in the NBS Applied Math Series No. 55. For the stated accuracy of 2*10^-9, argument must be between 0 and PI/2. (Unfortunately, a floating-point package published in Dr. Dobbs's Journal allows the argument to go up to PI.) The integer result of dividing the angle (in degrees) by 90 allows a CASE statement to handle each of the four quadrants in terms of a first-quadrant computation.

LOG and EXP (base 10) are also evaluated by using approximation polynomials given in AMS 55. Another two orders of accuracy are given by ACCULOG and ACCUEXP. These approximations use terminated continued fraction representations of (natural) log and exp, but with an initial value approximating the omitted "tail". The "tail" values are approximated by a linear fit through two accurate tail values.



COMPUTER DEN Ltd.

Call us for prices. We meet or beat most advertised prices.

Gemini 10X 275.00
Call for prices on all STAR printers
C. Itoh ProWriter II 530.00
Color, 180 CPS 550.00
Okidata Microline 92 .... 410.00
Epson RX-80 .... 250.00
RX-80 F/T+ 300.00
FX-80 F/T+............. 440.00
JX-80 .... 575.00
FX-100 .... 640.00
LQ-1500 (Parallel) ... 1225.00
Paper 20#, 2500 sheets 28.00
Grappler + ... 120.00
PKASO/U...................... 120.00
Print it! Interface 175.00
ShuffleBuffer (64K) 275.00
ShuffleBuffer (128K) 350.00
Printer Stand: short--20. long--25.
Disk holder for 3-1/2" 11.00
Disk holder for 5-1/4" 12.00
Hayes Smartmodem 1200 w/Terminal Program & CCS SERIAL card .. 630.00
Prometheus Promodem 1200 w/Terminal Program & CCS SERIAL card .. 520.00
CP/M CARD, CP/M 3.0, Guides . 270.00
SuperSprite Graphics Pkg. ... 320.00
Koala Pad Touch Tablet 80.00
Micro-Sci Disk Drive, A2 .... 225.00

--*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-

SPECIAL INVENTORY CLEARANCE SALE

Verbatim diskettes
SS/DD --18.00/10 --170.00/100
DS/DD --27.00/10 --250.00/100
3-1/2" -19.00/5 -- 36.00/10

Head Cleaning Kits 1.50
Visicalc for IIe 45.00
Trivia Fever 30.00
Infocom's SEASTALKER 37.50
Infocon's Enchanter 37.50
Wildcard 2 .... 104.25
FingerPrint (MX, RX Series) .. 45.00
(FX Series) 52.50
Record Master (Complete Data Base System) 45.00

--*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-

Call for Macintosh and Apple IIc software and peripherals.

This ad is written a month in advance. Since prices are subject to change, please call for current pricing. We attempt to meet all advertised prices.

COMPUTER DEN Ltd.
11716 KEMP MILL ROAD
SILVER SPRING, MARYLAND 20902
301-649-6868

Call to order
Sunday - Thursday, 10 A.M. - 9 P.M.
Free delivery to Crystal City
UPS shipping
Quantity Discounts Available

TRANSMITTING & SAVING FILES USING REMEMBER][

by Thomas S. Warrick

These instructions show step by step how to save files to disk using the public domain communications program REMEMBER][that is on Washington Apple Pi Disk 43. This program works only with the D.C. Hayes Micromodem II or IIe. In these instructions, things you should type into the computer are indented 10 spaces, so that

A

would mean that you should press the "A" key. Control characters are indicated as

<ctrl-A>

for control-A and

<RETURN>

for a carriage return. Where more than one line of instructions are shown together, you may need to pause for a second for another menu to be displayed or for a screen indicator to be updated.

1. Put a DOS 3.3 disk with REMEMBER][into the drive. If the Apple is not on, turn it on. Boot the disk if necessary. Once DOS 3.3 is in the Apple's memory, simply enter:

BRUN REMEMBER II<RETURN>

You should now see the REMEMBER][main menu.

2. Check the status of the program to make sure it is compatible with your hardware configuration. (If you have already used REMEMBER][successfully, it is not necessary to do anything except, perhaps, to turn lower-case translation off. See paragraph 4.) To see the status of the program, enter

R

and you will see the Status Report menu. First, check that the modem and printer slots are correct for your computer. Early versions of the REMEMBER][program had the modem slot set to 3, whereas many Apple owners now have their modems in slot 2. If you need to change the slot to 2, do this:

<RETURN>
Q
CALL -151<RETURN>
C87:2<RETURN>
803G<RETURN>
R

The first line exits the Status Report menu. The second quits from REMEMBER][. The third enters the monitor. The fourth sets the slot number, which is the number following the colon. The fifth line returns you to REMEMBER][. The last command returns you to the Status Report menu.

3. From the Status Report menu, make sure that REMEMBER][is in half-duplex so that you can see what you type. If the word "FULL" is in inverse (dark characters on a light background, rather than the light characters on a dark background in which most of the program's menus are displayed), enter

A

and "HALF" should become inverted.

4. If you want to have lower case letters appear and be saved as lower case, which you almost always will want to do, make sure that lower case translation, item D, is OFF. If it is not, enter

D

and the last word in the "D" line should change to OFF.

5. The other settings on the Status Report menus probably do not need to be changed unless the particular computer you will be communicating with requires them to be changed. You will almost certainly want the "SAVE BACKSPACED-OVER CHARACTERS" to be OFF, but as it will not have been turned on unless you specifically instructed it so, you should not need to change it. If SAVE BACKSPACED-OVER CHARACTERS is on and if either person types a character and backspaces over it while the memory is on, both the old character and the backspace will be saved in memory, along with the corrected character.

6. Return to the main menu by entering:

<RETURN>

7. You are now ready to establish the connection. If you are going to call the other party, read on. If you are going to receive the call from the other party, go to paragraph 8. To dial the desired phone number, enter

B

and REMEMBER][will prompt you for the phone number. Note that unlike the Micromodem II's native dialing command <ctrl-A> <ctrl-Q>, which dials each digit when you press the key, REMEMBER][does not dial the number until after you press <RETURN>. If, for example, you were dialing the Washington Apple Pi office to transfer an article for the Journal, you would enter:

654-8060<RETURN>

(Of course, if you want to transfer an article to the Journal, contact the editor to arrange a mutually-convenient time for transmission. Most of the time this number is answered by humans, not a modem. In addition, editorial and office staff sometimes are unable to handle file transfers due to other commitments.) If there is no answer to the call, REMEMBER][will return you to the main menu. If the connection is established, the Micromodem II will tell you so and REMEMBER][will put you into terminal mode. Go now to paragraph 9.

8. To receive a call, simply press:

C

A blinking cursor will appear but otherwise nothing will happen. This is normal and proper. When the

contd.

other side calls, REMEMBER][will answer the phone and put you immediately into terminal mode.

9. You should now be in terminal mode. Terminal mode is part of the firmware on the Micromodem, and operates as the Micromodem's terminal mode usually does. If you ever want to exit terminal mode and get back to the main REMEMBER][menu, enter:

<ctrl-A><ctrl-X><RETURN>

10. You can now "chat" over the line with the other party. If the other party cannot see what he or she is typing, remind them that they should be in half-duplex. The person sending the file should state, if it has not been discussed in advance, what the name of the file is, how long it is, and what should be done with the file after it is saved to disk. The mechanics of sending and receiving a file between two REMEMBER][users will be explained in the remainder of these instructions. The sender of the file will be referred to as "sender." You will have to guess what "recipient" or similar words mean in the instructions below. An overview of the process is:

(a) The person receiving the file should turn REMEMBER]['s memory on.

(b) The person sending the file should load it into REMEMBER]['s memory and transmit it.

(c) The person receiving the file should turn REMEMBER]['s memory off and save the file to disk. This completes the process.

11. If you are receiving the file, inform the person at the other end that you are about to turn REMEMBER]['s memory on by saying something like:

I AM NOW TURNING MY MEMORY ON. I WILL TYPE "OK" WHEN I AM READY TO RECEIVE THE FILE.

The recipient should then exit terminal mode and turn the REMEMBER][memory on:

<ctrl-A><ctrl-X><RETURN>
E

The "ON" after "E - MEMORY" should become inverted. REMEMBER]['s memory should be cleared, so that the first two hexadecimal digits on the E line should be the same. (Most of the time they will be $1901-$1901, with a $9600 after them.) If they are not the same, enter

I
Y

to clear memory. In any event, the recipient should next enter:

A

to return to terminal mode. Then type:

OK<RETURN>

to let the sender know you are ready to receive the file. Note that the "OK" will be saved as part of the received file and will need to be deleted from the disk file after the file is saved. (Once memory is on, both people should refrain from typing any unnecessary characters that would take up disk space or make cleanup of the transmitted text file more difficult.)

12. The sender of the file must now load it into memory. Use the

K

command, even though it is entitled "SEND FILE." Then enter:

A

to load a text file. Enter the name of the file you wish to load. (If you press <RETURN>, you will get back to the main menu.) Thus, if the file is named SENDTHIS, enter:

SENDTHIS<RETURN>

After the load is complete, REMEMBER][will return you to the Send File menu. Press

<RETURN>

to get back to the main menu.

13. If you wish to display the file while it is in memory, use the

G

command. Do not be alarmed if lower-case characters appear as inverse digits and special characters. If upper-case characters appear as anything but upper-case characters, become alarmed. This should not happen. To interrupt the file listing during display, enter:

<ctrl-C>

After the entire file has been displayed, press

<RETURN>

to return to the main menu.

14. The sender should now begin sending the file as ASCII characters, i.e., regular text. Press

K

to get back to the Send File menu and

D

and transmission will automatically begin. When file transmission is complete, the sender will automatically see the main menu. The sender should now press

A

to return to terminal mode, but the sender should not type anything else at this point. The sender should wait for the recipient to type something.

15. At this point, characters should have stopped coming over the recipient's screen. Unless the sender is sending a file with many non-printable control characters, the recipient should be safe in assuming that file transmission is now complete. The recipient should immediately turn off memory by entering:

<ctrl-A><ctrl-X><RETURN>
F

If you as recipient wish to examine the memory at this point to confirm that the document has been saved in memory, enter:

G

and the contents of memory will be displayed. The G command is further described in paragraph 13. Return to the main menu after checking the memory by pressing:

<RETURN>

16. The recipient should now re-enter terminal mode by pressing:

contd.

A

The recipient should then type something to the sender to let the latter person know that the transmission has been completed successfully:

THE FILE WAS RECEIVED SUCCESSFULLY. THANK YOU VERY MUCH.

If the file was not received correctly, the recipient should say so, and the file will have to be retransmitted.

17. The telephone connection may now be broken. Both the sender and the recipient should exit terminal mode and hang up the phone:

<ctrl-A><ctrl-X><RETURN>
D

18. The sender can now quit REMEMBER][by entering:

Q

19. If you are the recipient, you must now save REMEMBER][s memory to disk. This is done by entering:

J
A

You then enter the name of the file in which the contents will be saved:

RECEIVEDFILE<RETURN>

Note that the disk the file is saved on must be initialized and should have enough memory to save the file. After the file is saved to disk, you will be returned to the main menu.

20. You may now quit the program by entering:

Q

21. Use a word processor that can handle standard DOS 3.3 files, such as Apple Writer II or //e or Screen-Writer, to examine and manipulate the file. If you use a ProDOS-based word processor, simply CONVERT the saved text file to the ProDOS format. (A text editor such as comes with most assemblers will also work if the lines are not too long. Note that most word processors put carriage returns only at the end of a paragraph, so that in such cases text editors may not work.) You will have to delete the "OK<RETURN>" that you were told to enter back in paragraph 11. Make any other changes you wish and re-save the file to disk.

22. If the file you received was an Applesoft BASIC program, you will want to convert it from a text file to an Applesoft file using the EXEC command. If you saved the contents of memory under the file name RECEIVEDFILE, enter:

EXEC RECEIVEDFILE

You should see a series of characters, mostly]'s, go flying by. After these characters stop coming, try entering

LIST

and the program should now be in memory. Save it as an Applesoft file by entering:

SAVE NEWPROGRAM

If the program is a binary file rather than an Applesoft file, the EXEC command will also work, but you must know how to BSAVE the file. For further details, see the DOS 3.3 or ProDOS manuals, as appropriate for the operating system you are using.

TELECOMM SIGNEWS

by Dave Harvey

The Telecomm SIG was held after the WAP meeting on October 27 and was conducted by George Kinal. There were about 15 members attending.

There was a general discussion on problems that members were having. One problem was with the sending of a password to a DEC computer. The computer would accept the USERID but when a password was sent, it would not accept it. One suggestion was that maybe non-printing characters were being sent during the sequence which affected the terminal software.

Someone asked what full and half duplex transmission was and there followed a explanation of these two types of transmissions.

There was also a discussion of CP/M downloading. There are a number of local bulletin board systems which have CP/M programs on them. It was pointed out that for most of them a protocol transfer is required. An exception to this is the AJ BBS which allows regular transfer. DOS 3.3 can be used without any problem and a CP/M terminal program is not required for just downloading.

George pointed out that he had been receiving a lot of calls about the new serial interface cards. Two new ones, CCS 7711 and the Practical Peripherals Seri-all do not allow or support carrier detect.

Someone asked how long it takes to get on the WAPABBS after a member selects a password. The time is usually 5 days. One member stated it had been 3 months. A call to the office was suggested in that case. It should never take that long.

The next item of discussion was the problems new Apple //c owners have with communications. One problem is that at present only the Apple Modem works with the //c. This is a hardware defect within the Apple and Apple will swap motherboards if there is a problem. The baud rate generator within the Apple is the cause of the problem. It affects some printers as well as modems.

Someone asked what were some good books on telecommunications. One mentioned was "The Complete Handbook of Personal Computer Communications" by Alfred Glossbrenner. Another is "Joy of Telecommunications" by WAP's own Bill Cook. This book will probably be sold at the WAP office if enough interest is shown in carrying it.

There was a question about sending formatting codes to a host computer. Could it be done? The answer was that a lot depended on the host computer that you were sending to. One suggestion was that with some word processors, a formatted file can be sent to disk instead of to a printer and then when saved to disk, send this file to the host computer. All control characters will have been replaced with spaces. One caution was that with some hosts the sending of two carriage returns in a row indicates end of file. It was pointed out that one terminal program, ASCII Express Professional, allows you the option of inserting spaces in a file whenever this condition occurs so that the host continues to receive a file.

Someone asked if there were any programs that would emulate computer terminals. The program "Softerm 2" was mentioned as one of the most popular ones. This

contd. on pg 51

DECEMBER SPECIALS FOR ALL MACINTOSH OWNERS

FROM

MICROWARE - EAST

Post Office Box 612 Sterling, Virginia 22171 Phone: (703) 435 - 1380

Product	Company	List Price	OUR Price
DISKETTES:			
3 1/2" SS/DD Disketts (Box of 5)	BASF	$31.00	$20.00
SOFTWARE:			
PFS: File	Software Publishing	$125.00	$82.00
PFS: Report	Software Publishing	$125.00	$82.00
PFS: File/Report (Bundle)	Software Publishing	$195.00	$125.00
Friday	Ashton-Tate	$295.00	$205.00
D-Base II	Ashton-Tate	$495.00	$345.00
Think Tank	Living Videotext	$145.00	$99.00
Mega Filer	Megahaus	$195.00	$135.00
Mega Merge	Megahaus	$125.00	$85.00
Chart	Microsoft	$125.00	$85.00
File	Microsoft	$195.00	$129.00
Microsoft Basic	Microsoft	$150.00	$99.00
Multiplan	Microsoft	$195.00	$135.00
Word	Microsoft	$195.00	$135.00
Home Accountant	Continental	$99.95	$65.00
DB Master	Stoneware	$195.00	$135.00
Main Street Filer	Main Street Soft.	$199.95	$125.00
Filevision	Telos	$195.00	$135.00
Sargon	Hayden Software	$49.95	$34.00
Word Challenge	Hayden Software	$39.95	$29.00
DaVinci: Houses	Hayden Software	$49.95	$34.00
DaVinci: Landscapes	Hayden Software	$49.95	$34.00
DaVinci: Interiors	Hayden Software	$49.95	$34.00
DaVinci: Building Blocks	Hayden Software	$79.95	$55.00
DaVinci: Commer. Interior	Hayden Software	$199.95	$130.00
Speller	Hayden Software	$79.95	$55.00
Dollars & Sense	Monogram	$149.95	$98.00
Mac Pic Volume 1	Magnum Software	$49.95	$34.00
Mac Pic Volume 2	Magnum Software	$49.95	$34.00
Typing Tutor	Simon & Schuster	$59.95	$40.00
Mastertype	Scarborough	$49.95	$35.00
Sales Edge	Human Edge	$250.00	$155.00
GAMES:			
Millionaire	Blue Chip Software	$59.95	$38.00
Murder by the Dozen	CBS Softeware	$44.95	$35.00
Pensate	Penguin Software	$39.95	$26.00
Transylvania	Penguin Software	$39.95	$26.00
Run For The Money	Scarborough Systems	$49.95	$32.00
Frogger	Sierra On-Line	$39.95	$27.00

Product	Company	List Price	OUR Price
GAMES:			
Cutthroats	Infocom	$39.95	$27.00
Deadline	Infocom	$49.95	$39.00
Enchanter	Infocom	$39.95	$27.00
Hitchers Gde To Galaxy	Infocom	$39.95	$27.00
Infidel	Infocom	$44.95	$34.00
Planetfall	Infocom	$39.95	$27.00
Seastalker	Infocom	$39.95	$27.00
Sorceror	Infocom	$44.95	$34.00
Starcross	Infocom	$49.95	$38.00
Suspects	Infocom	$44.95	$34.00
Suspended	Infocom	$59.95	$39.00
Witness	Infocom	$39.95	$27.00
Zork I	Infocom	$39.95	$27.00
Zork II	Infocom	$44.95	$34.00
Zork III	Infocom	$44.95	$34.00
Trivia	Kastel Technology	$59.95	$38.00
Disque	Kastel Technology	$27.95	$20.00
Jocks	Kastel Technology	$27.95	$20.00
The Me Generation	Kastel Technology	$27.95	$20.00
Show Biz	Kastel Technology	$27.95	$20.00
What's That Tune	Kastel Technology	$29.95	$22.00
HARDWARE:			
MAC 11.1 MB Hard Disk Dr.	Corvus	$2,495.00	$1,895.00
MAC 5.5 MB Hard Disk Dr.	Corvus	$1,795.00	$1,395.00
MAC Starter Kit (Swivel/Dust Cover/Surge Protector)	Kensington Microwar	$90.00	$60.00
5 MB Mac Drive Hard Disk (5.5 MB Removeable Cartridge)	Tecmar, Inc.	$1,995.00	$1,495.00
10 MB Mac Drive Hard Disk (Fixed Disk)	Tecmar, Inc.	$1,995.00	$1,495.00
Mac Drive Upgrade Kit (2nd 5.5 MB Removeable Hard Disk)		$1,295.00	$995.00
Mac Pac Kit (includes cable & Software)	Prometheus	$150.00	$120.00
MAC PAC (Promodem/cable/software)		$495.00	$395.00

Terms: All prices subject to change & availability.
Corporate Accounts Welcome
VISA & Master Charge (add 3%)
Virginia Residents (add 4% Tax)
Add $2 For C.O.D.

Shipping: $2.50 per item UPS for Diskettes & Software
$10.00 per item for Hardware Shipping & Handling

** Allow 2 weeks for clearing of Personal & Corporate checks

* * * * *

Comparable Savings in Software and Diskettes for the Apple IIe and IIc. Please call for prices!!!

SIG MAC NEWS

by Ellen L. Bouwkamp
& Steve Hunt

The November 10 meeting was our best one yet. For those members who may have missed it, we have taken a couple of steps to improve the meeting format and ensure that everyone (new user and experienced hacker) benefits.

So that everyone can see what is happening, we meet in the auditorium (which holds over 350 people). Since we had over 220 members this month, we probably should start a contest to see when we will exceed the capacity of the auditorium?

Through the generosity of David Morganstein (President of the Pi), we have been able to use a large screen projector which gives a far better view of the Mac than we had before. (The Pi is in the process of buying its own display so we don't have to keep mooching.)

BUSINESS MEETING:

A quick glance around the Auditorium confirmed that attendance DOUBLED since last month's meeting. We have started advertising SigMac meetings in the Washington Post Computer Calendar (Monday's in the Business Section) and will continue to do so. Each member is also encouraged to spread the word about SigMac ... let's fill the Auditorium!

SigMac officers meet regularly to plan meetings and begin action to improve services provided to members. We need your ideas, advice and counsel! Call Steve Hunt or Don Landing to bring up any ideas you want the officers to consider.

Some people have been confused about the different SigMac meetings - so we offer the following clarifications.

Thursday meetings are intended for ADVANCED Mac users. They are not a substitute for the Saturday meetings. If you are interested in exploring the details of the Mac, attend the Thursday meetings. The next meeting will be December 6 from 7:30 pm to 9:30 pm at the Our Lady of Lourdes Parochial School, 7500 Pearl Street, Bethesda. [NOTE THIS NEW LOCATION.] Be prepared for about an hour of give and take and an hour of specific programming information. The next few meetings will explore Inside Macintosh.

Saturday meetings are intended for ALL Mac users. We will continue with our question and answer sessions, a short business meeting, a main program (including demos of new releases and rumors), then splitting into two groups (advanced users and new users). Unfortunately, we did not have ample room for the new user meeting this month but this approach will resume at the December meeting.

We are working hard to continue to improve the meetings. We need your help. The quality of services you receive from this club is in direct proportion to the amount of time and effort contributed by volunteers. No long term commitments are required - even an hour or two will be welcome.

In the works:

- At our next Saturday meeting, we will have a handout with the names and numbers of experts you can contact for Mac problems and questions, current version numbers of major software releases, a listing of the programs on the SigMac diskettes, and other useful information.
- Software/hardware problem sheet. We will be developing a form so that you can report current problems with Mac items. The form will be distributed on future SigMac disks. We want to collect the forms (either at the meetings or by mail) and use them to answer questions, develop articles for the Journal, and keep track of major trends.
- A Mac "new users" guide and disk. We got five volunteers to help and will be contacting them to get things really started. We will report back to the membership at the next meeting.
- Directory of SigMac members. We are still working on this project - resolving a few issues regarding data collection and privacy. In the interim, we will have the short list of who to contact for problems available at the next meeting.
- Tutorials. We are planning on a regular program of tutorials (1-3 hour sessions dedicated to a specific subject such as telecommunications, beginning basic programming, Forth, etc). Members attending this meeting were given a chance to indicate which tutorials they wanted.
- A special Mac Bulletin Board. Approximately half of the members at the meeting indicated they had a modem and were interested in having access to a bulletin board system oriented to the Mac. We will work on this issue and report back to the membership at the next meeting.

Washington Apple Pi consistently produces one of the finest users' journals anywhere in the U.S. Special thanks (and a warm round of applause) go to Bernie and Genevie Urban for their superb work. Our WAP Journal includes many valuable articles about the Mac - virtually all of which are written by WAP members! We solicit your articles - even a paragraph or two will do! Write them now - and send to Bernie Urban by the deadline shown on the calendar. If you are sending a long article, please send Bernie a diskette - it will be returned to you if you remember to include your address.

NEWS, RUMORS, AND LATE BREAKING GOSSIP:

A book called The Macintosh Buyers Guide is available in bookstores and computer dealers. It summarizes new products for the Mac. The list price is $7.50 but you should first ask because dealers have a limited supply of these to give free to customers (if you are nice to them).

The special election issue of Newsweek contains nothing but Macintosh advertising. You have to see it to appreciate it!

RUMORS: Rumors abound about what will be announced at Apple's January annual meeting. Apple usually takes the opportunity at that time to make BIG announcements. Rumors include: a color Mac (another upgrade kit??), an extra-large memory - 2 M bytes (another upgrade kit??), 20 MB internal hard disk drive supposedly being supplied by Tandon. There are also

contd.

rumors of a new 1-gigabyte (1 billion bytes) optical disk and networking capability intended for the business environment. Corvus announced networking software to be available next year.

Lotus is also expected to announce, on Nov 12th, their 512K Mac product - reportedly called "JAZZ". One SigMac member who has seen it was most impressed. It supposedly will be available before March 1, 1985.

Dennis Brothers has a disk backup program available on Compuserve. We will provide a copy on a future SigMac disk for those who are not members of CompuServe (or want to save the cost and headaches associated with downloading).

CompuServe also included a notice of a "Fast Finder" developed by a third-party software firm. It supposedly supports a hard disk drive, includes on-line help menus, automatically initiates an application by clicking on a file, and several other features. Members can contact the developer at Fast Finder, 2817 Sloat Rd, Pebble Beach, CA 93953 or at (408) 372-1722.

DEMONSTRATIONS:

MacDraw (demo by Tom Berilla): MacDraw is an graphics program with tremendous power. It is a companion to - not a substitute for - MacPaint. MacDraw allows you to develop large drawings (a maximum size of 96.0 by 48.0 inches) and extend objects across these page boundaries. A pallete of drawing tools (similar to MacPaint) is available down the left side of the screen. Individual objects can be grouped together (a feature which is not supported in MacPaint) to develop more sophisticated drawings. There are also three magnifications (normal, one-third normal, and one-ninth normal). The demo files shown by Tom include an architectural/drafting package. Tom moved and extended walls, moved furniture, and changed scales. This was a pre-release copy, the official release date has not been established when this was written.

MAGIC5 - the Macintosh Graphics Input Controller (demo by Tony Taylor and Mike Fritz): MAGIC digitizes graphic and print input from a video camera or a VCR tape and saves it in MacPaint format. You can take pictures of pages in magazines, three dimensional objects, as well as "real life" scenes. The camera captures 768 dots by 525 lines. You have control over where the camera picture will be placed on the Mac screen. A review comand allows you to look at the drawings you've saved. This is a very impressive system for digitizing images. The quality is the best yet. MAGIC costs $399 by itself; $549 for MAGIC and a video camera, and if you buy MAGIC and later need the camera, the camera will cost you $219. Contact Tony Taylor at Sterlingware Computer Products in Sterling or write New Image Technology, Inc. 10300 Greenbelt Road, Seabrook, MD 20706. Mike Fritz can be reached at (301) 464-3100. The system is available now.

Music Works (Demo by Withers Morgen): This new program by MacroMind of Chicago Ill generates superb music and drew a hearty round of applause from all attendees. Music comes from either the internal speaker or a stereo system connected to the back of the Mac. Music can be written directly on a score by pulling notes off a palette. The notes are automatically formatted according to the specified tempo and type of note. There are 10 possible instruments (you can use any 4 simultaneously). Menu options allow variations of sound (hard-loud, vibretto etc.). A "master score" shows the overall composition the notes are being played. The score can be printed. A future program will allow you to combine music with animated pictures. Hayden Software is the distributor $79.95. Available by 1 Dec 84.

MacProject (demo by Steve Hunt): This is a "PERT" network program that allows you to define events and interconnecting dependencies. It goes far beyond just providing a network schedule, however, because the program also has Gantt charts for both events and resources, cash flow charts and special displays which allow assignment of costs to individual resources. The program calculates costs for each event based on the resources which are identified as required to complete the event. The page layout is similar to MacDraw allowing a total size of 96" x 48". The project start and stop dates (as well as the number of hours in each work day) are specified with an easy-to-use calendar. There are many other options which provide a wide range of tools to a program manager. A project of about 2000 events can be handled in a 512K Mac (approximately 200 events in a 128K Mac). This program is one of the two free applications with the 512K upgrade. Apple Computer, $125.00. Available now.

Icon Editor (demo by Withers Morgan): This set of routines allows you to develop your own icons and assign them to an individual application. Withers showed a copy of one of the "Mac Stuff" disks from the Inside Macintosh Software Supplement that had been modified with all new icons. They were truely impressive and easily illustrated the purpose of the application. This capability is intended for the advanced user and is not yet ready for use by novice Macintosh users.

The meeting could have gone on for several more hours since there is now a rapidly growing collection of software available for our favorite computer. The range of applications and the extremely high quality of most applications is most welcome and very impressive.

Stay tuned . . . there will be even more to come next month.

$$$ DISCOUNT PRICES $$$

Amdek Color I	$270	D.C. Hayes 300 Baud Smartmodem	$215
Amdek Color II (RGB)	$410	D.C. Hayes 1200 Baud Smartmodem	$480
Amdek 300 A	$155	Koala Gibson Light Pen	$195
NEC 1260 (Green)	$120	Apple Dumpling w/16K	$165
NEC Color (Composite)	$270	Applicard (6 MHz)	$260
NEC RGB	$395	Stock Option Analysis Program (H & H Scientific)	$250
NEC 3510 Spinwriter	$1250	Stock Option Scanner (H & H Scientific)	$350
Okidata 82 A	$310	Money Decisions Vol. II	$180
Okidata 83 A	$560	Fox & Geller Quick Code	$210
Okidata 92	$435	BPI General Accounting IIe	$240
Okidata 93	$650		
Epson FX 80	$435		
Epson FX 100	$600		
Toshiba 1351	$1300		
D.C. Hayes Micromodem IIe	$230		

VISA MasterCard

VISA/MC (Add 3%), money order, certified check.
Government purchase orders accepted.
Prices subject to change. Shipping/handling $5
MD residents at 5% tax. Mail order only.

CALL FOR ITEMS NOT LISTED
We carry a complete line of hardware and software items.

RAMADA COMPUTER PRODUCTS
A Division of H & H Scientific
13507 Pendleton St.
Ft. Washington, MD 20744
Tel. (301) 292-2958

MAC Q & A

by Ellen Bouwkamp and Steve Hunt

The following are just a few of the questions and answers addressed at the SigMac meeting on Nov 10 including a few that have been sent in by mail.

Q. On several SIGMAC disks, I get an error message "application not found" - what am I doing wrong?

A. Many of the applications on the SigMac disks are actually MS-Basic programs. To run them you must transfer these programs to a disk with a copy of MS-Basic ON THAT DISK and then insert this new disk on the internal Mac disk drive. The program should run properly. The current versions on MS-BASIC (1.0 and 1.01) do not work properly if Basic is on one disk and the application itself is on another disk.

Q. When transferring a MacPaint document to MacWrite, can I transfer an entire MacPaint page to MacWrite?

A. No. With the current version of the program, you can only transfer a screen image (about 1/3 of a page) to MacWrite.

Q. Why do the worst errors come after 5 PM on Friday when the technical support staff has gone home?

A. Presumably there is a malicious spirit in many things - even computers. This is just one reason why we much prefer a few delays in releasing new software to try to minimize the errors. (Unfortunately, no one could resolve the specific problems being experienced by the individual asking the question.)

Q. Has anyone had problems with BASF disks?

A. No. The disks now on the market appear to be quite reliable - even the early problems with some Memorex disks are also gone.

Q. What mail order houses have been the most reliable and is there a relationship between the price of software and the quality?

A. The best mail order houses: (1) MacConnection provides good service though you may have trouble getting through, (2) Northeastern Software provided fine delivery of Chart for at least one member, (3) Programs Plus is very good and has a knowledgeable staff. PC-Network does NOT have a good reputation with Apple Pi members and should be avoided. Rising Sun in California (previously mentioned in the Journal) appears to be out of business.

Q. What protection do I have with mail order firms?

A. If you order by telephone, the FTC laws DO NOT apply; if you are concerned about mail fraud and feel the need for legal protection you should send your order by mail (or better still, deal with a reputable firm). Use a credit card (NOT a debit card) if you want the maximum recourse against failure to deliver software. Don't give out a bank "debit card" number to any mail order firm.

Q. I have had problems with Multiplan files larger than 30K bytes. Can't paste and link and I also have trouble printing. What is wrong?

A. There is a bug in Multiplan which causes this problem. The print problem does not occur on the 512K MAC but does occur on the 128K version.

Q. I have a copy of a program called "File Edit", how can I use it?

A. Very carefully. This program (intended for system developers) allows you to do dangerous things to the files on your disks. You can destroy your systems file. Novice users should not use this program.

Q. I have an intermittent problem with MacWrite - keep getting a system error 28. What am I doing wrong?

A. The disk could be bad or be too full. We suggest you get a fresh copy of the systems file and of MacWrite and try again. This error should not be occurring.

Q. There appears to be problems with a few of the Desk Accessories on a recent SigMac disk. Can you help?

A. There is a problem if you try to use the magnifying glass and the "bugs" at the same time. Both of these desk accessories want to be on top and this conflict crashes the system.

Q. Has anyone seen MacPaint v. 1.4 for the 512K Mac?

A. Yes, it was released with the 512K Mac and should be provided by your dealer with the 512K machine or when you get a 512K upgrade.

Q. On MacTep v. 1.87, I occasionally get an "out of memory" error while data is still coming in from a bulletin board. What is wrong?

A. This is a bug in MacTep. We don't know of any fix at this time.

Q. I see references to a "library" and a "disketeria". What is the difference between them?

A. The library is the collection of diskettes kept in the WAP office. You may use them in the office by appointment (because we only have one Mac in the office). They are not available for loan. The "disketeria" is the name given to the diskettes which have public-domain software on them and which are sold by the WAP at club meetings. [Some of the programs on these diskettes ask for extra payment (generally to the author) if you like and use them. If you intend to use these programs, you should honor any author's request for payment, which is usually less than $50. We think these payments will help keep the flow of public-domain, user-initiated software available.]

Q. What does "assembly-based" programming mean? Do I really have to know the difference between this and BASIC-based programming?

A. "Assembly-based" programs are written in a computer language built into the Mac. This means you don't have to buy any other language (such as BASIC or Pascal) to run the program - it will run on your Mac as is. BASIC is another language available for the Mac (which is far easier to use than assembly language because it uses more English-like words than the assembly language). If you want to run an

contd. on pg 45

PERSPECTIVES OF A NEW MAC USER

by Ellen Bouwkamp and Steve Hunt

"The computer for the rest of us" -- not just a catchy advertising slogan but a drawing card which brings a new group of users to the Washington Apple Pi.

Ralph Begleiter (a new member of Washington Apple Pi) wrote a lengthy letter to the club which really hit home. Ralph is both a new Mac owner and is totally new to computers.

He said it best --

"I never owned a computer before."

"I've bought one specifically because it requires very little knowledge about computers to operate."

"I don't want to know much about how computers operate."

"I want/need help using my machine and its software, without getting bogged down in 'programming'."

"I don't know the "lingo". . . . I recognize that many of your members are very experienced computer users . . . there are a small-but-growing number of us who are a different breed."

The Macintosh Special Interest Group (SigMac) is committed to meeting the needs of ALL members. Though many of us have been involved with computers for several years we really will try to remember what it was like when we began.

SigMac will offer the following extra hand to new users.

- NEW USER FORUMS. Though we have had some problems scheduling the rooms, we will have a "new user forum" from about 11:30 till 1:30 (following the main presentation) at the saturday SigMac meetings (2nd Saturday of each month).

- NEW MEMBERS GUIDE AND DISK. We are developing a special summary of Mac User Tips and suggestions for inclusion in the Washington Apple Pi New Members Guide. We will also have a new members disk. Look for them in January. (Sorry for the delay but it will be worth the wait.)

- TUTORIALS. We have already presented one tutorial on telecommunications with the Macintosh and will be getting ready for a more extensive schedule after the holiday rush.

- JOURNAL ARTICLES. We hope to see a steady flow of articles for this journal aimed for the new Mac user. Several people have already volunteered to help write articles and do the necessary editing. Stay tuned.

Each of you should consider the following actions so you get the most out of your membership:

- VOLUNTEER! This is a volunteer group. Any offer of help is welcome. We promise you plenty of opportunity to VOLUNTEER! The more you get involved, the faster you'll learn! You DON'T have to be an expert or programming whiz - just willing to help.

- COME TO MEETINGS! We cannot answer all your questions or concerns in the Journal. At meetings, you can speak up during the "question and answer sessions" or easily corner someone to resolve those nagging questions.

- CALL SOMEONE! If you need special assistance, call the individuals shown in the Journal (WAP notline) or call the office to get a name of someone who can help you.

We all want to share the joys of exploring this new machine and getting the most out of it - whether you are a novice or an experienced "hacker".

Mac Q & A contd. from pg 44

application written in BASIC, you have to buy a copy of the BASIC language (like MS-BASIC), load that software, and then load and run your BASIC program.

Q. Where can I find out more about surge protectors?

A. John Hardis remembered an article in the December 1983 issue of BYTE (Vol 8, No 12, p.36). This article contains instructions on how to make a good surge protector out of $5.00 worth of parts (if you like electronics). If you don't want to build one, see your dealer and ask for a comparison of the best models. We will also try to get more information for a future article in the Journal.

If you have other questions, send them in by mail to Ellen Bouwkamp or come to the Saturday SigMac meetings and get the answer first hand.

DISK DRIVES $150.
Direct drive, TEAC mechanism, 1/2 Ht.
Disk controller card $50.

MACRO CARD keyboard enhancer $75.
70 FIXED function keys, 120+ USER DEFINED,
128 KEY TYPE-AHEAD BUFFER, MACRO DISK FILES,
APPLESOFT AND CP/M; for Apple II & II+.

128k RAMCARDs $150.
GRAFIX-PRINT Cards + cable $80

SYSTEMS SERVICES COMPANY
1125 SHANNON PLACE, HERNDON VA 22070
(703) 435-3896 VA residents add 4% tax.

THE MAC UPGRADE

September 14, 1984

Mr. Steven Jobs
Apple Computer, Inc.
20525 Mariani Avenue
Cupertino, CA 95014

Dear Mr. Jobs,

Seeing Apple's full-page ads this week announcing the "FatMac" and the new, lower price for the Macintosh has prompted me to write this letter. Before I make my main point, however, let me say that I am a new Apple customer..in fact, a newcomer to the world of microcomputers. I am not a "hacker", but a journalist who became convinced when I saw the Mac that it was the kind of machine which could finally bring the power of computers into the hands of people who don't know how they work, don't know how to program them... and don't want to know those things. I am fully satisfied with my new computer so far.

But (you knew that was coming, didn't you?) seeing Apple's new policy on the "FatMac" and the regular Mac price touched a nerve with me. One of your ads (New York Times, 9/12/84) says:

> "We introduced Macintosh last January, and sold tens of thousands overnight. And continued to sell tens of thousands month after month. Hence achieving certain 'Economies of Scale'...Our costs went down...so we are passing the savings...along to you, the consumer."

Well, I'm one of those consumers whose purchases allowed you to achieve economies of scale. I'm one of those people who bought the new Mac knowing that it would have only 128K and one disk drive. Many of those "tens of thousands" of consumers paid list or close-to-list price for their Macs: $2495, plus Imagewriters (at around $500).

You've now lowered the basic Mac's price by about $300 (though dealers are beginning to offer discounts of up to $300 more). The new price is $2195. And you've begun selling the "FatMac" for about $3195. You are pricing the "upgrade" for basic Mac owners at nearly a thousand dollars. That means someone who paid original list must pay about $300 more for a "FatMac" than people who are buying their first Mac today.

I realize, of course, that Apple is in business to make money, not to make friends. But the pricing arrangements sure appear to us pioneers as a "penalty" for buying into the Macintosh when it was just a dream...just a hope for Apple. Have you considered the possibility of offering some sort of concessionary pricing or upgrade (if not the expensive 512K chip, how about a less costly external disk drive) for those "tens of thousands" of consumers who helped allow Apple to achieve those "economies of scale" which now make lower prices possible? In other words, how about passing along some of the reward to those who contributed to Apple's Mac success by buying before September 12, 1984?

Thank you for listening.

Sincerely,

Ralph J. Begleiter

apple computer inc.

20525 Mariani Avenue
Cupertino, California 95014-2094
(408) 996 1010

October 19, 1984

Mr. Ralph J. Begleiter
9820 Betteker Lane
Potomac, MD 20854

Dear Mr. Begleiter:

Thank you for taking the time to write to Apple. Several questions have come up recently regarding the Macintosh 512K Memory Expansion, and we would quickly like to explain why we priced the expansion kit at $995 and how the free software offer works.

WHY WE THINK $995 IS A GOOD DEAL. There are two reasons why the additional 384K bytes of memory cost $995.

First, please note that other major computer manufacturers (like IBM and Compaq) offer 384K bytes of memory in the $750-$1000 price range. These upgrades involve exchanging chips or adding additional modules. The Macintosh Memory Expansion, however, involves swapping the entire digital board. This approach is much more reliable since it eliminates chip sockets which can be mechanically defective and minimizes the amount of work and handling a service technician has to perform. The Macintosh Memory Expansion is so reliable that it's backed up by a 90-day warranty. Replacing the entire board instead of the chips costs us more, but we think it's worth it in terms of reliability in the long run.

Second, the Macintosh Memory Expansion uses 356 Kbit dynamic RAM chips which store four times as much information as the 64K chips used in Macintosh 128K. These chips are the most-advanced memory circuits in high volume production. Apple had to pay premium prices to get them to you many months earlier than both you and we had anticipated. Furthermore, we use 16 of these chips in a Macintosh 512K.

In summary, $995 is the most aggressive price we can possibly offer. The Macintosh 512K Memory Expansion is priced at the higher end of the industry standard because it is more reliable and because Apple pays a premium for the advanced memory chips.

THE FREE SOFTWARE OFFER. We realize that the people who bought Macintosh early are our most valuable customers. You people led the market and supported Apple when Macintosh was very new. We want to thank you and reward you for your business, so we're offering a free copy of MacDraw and MacProject when you expand to the 512K machine. Before we introduced Macintosh 512K we met with early Macintosh purchasers to understand who would buy the Memory Expansion kit and why. We learned two things: you wanted to be RECOGNIZED and REWARDED by Apple, and you wanted more business software.

So we put together a program that gives two powerful business tools (a $250 value) to those of you who bought Macintosh when there was only one machine to choose from.

In closing, we want to thank you for your business. We hope we've cleared up any confusion that may have existed, and that you will take advantage of our free software offer when you purchase a Memory Expansion kit.

Sincerely,

APPLE CUSTOMER RELATIONS

THE APPLE WRITER HOTLINE CALLS BACK: Part II - Customizing the Merge Program

by Dianne Lorenz

The following merge program has been made interactive and will accept the name of the base letter, the address file, and the date. You can expand its use to include a variable for drive # so that the documents you are working with can be in either drive. It is currently set for a one-drive computer. Type in the program exactly as is, especially observing all spaces and capital letters. Save to your disk as Merge and then practice by taking the tutorial that follows. This program can be used for any combination of base letters and address files.

```
START   pin Please put disk containing base letter and
addresses in drive 1 and press       RETURN.
        pin Enter name of base letter =$b
        pin Enter name of address file =$c
        pin Enter date =$d
        PSX 1
LOOP    NY
        L $b,D1
        B
        F/(Date)/$d/
        Y?
        B
        F/(Address)//
        Y?
        L$c,D1/<(X)>/</N
        PGO FOUND
        PGO QUIT
FOUND   PLS$c,D1/<(X)>/ /N=$A
        B
        F/(Name)/$A/A
        PNP
        PSX +1
        PGO LOOP
QUIT    PIN         Done at address (X) (press RETURN)
        NY
```

A FORMLETTER TUTORIAL

Developing a formletter to be sent to many recipients involves three steps: (1) creating a base letter, (2) creating an address file, and (3) running the MERGE program to produce the finished letters.

CREATING THE BASE LETTER

1. Clear memory, CTRL N,Y, use the spacebar or tab to position the cursor where you want the date to appear, and type (Date).

2. Leave several blank lines, and where you want the address, type (Address).

3. After the greeting (Dear, etc.), type (Name) for the Name variable.

4. Write the letter. Type (Name) any place in the letter where you want the first name to appear.

5. After the closing and space for the signature, type the following for the instructions that allow one letter at a time to be printed and advance the printer to the next sheet of stationery:

```
.inAddress (X) (Press Return)
.FF
```

Your screen should resemble the one below.

```
                                        (Date)

(Address)

Dear (Name),

          The letter goes here.....

Sincerely,

.inAddress (X) (Press Return)
.FF
```

Type CTRL S to save the base letter to your disk giving it an appropriate name.

CREATING AN ADDRESS FILE

Clear memory, CTRL N,Y, and follow the form below to create an address file. Be sure to use the <> symbols around the item numbers; they are important markers for the merge program. Address must be flush left.

```
<1>John Brown
401 Main Street
Harpers Ferry, VA 22201
<2>John Wilkes Booth
Ford's Theater
Washington, DC 20001
<3>Anne Boleyn
Tower of London
London, England   (No cutting comments, please - Ed.)
<
```

The final < symbol is not a typo, but an essential character to indicate that all addresses have been processed. Save the address file to the disk. You may have several different address files; therefore, you may want to name them in a way that will make it easy to recall their contents. For example, names like Friends or Creditors will be easier to identify than address1 or address2.

MERGING FILES

Because the MERGE program is written in WPL (Word Processing Language), it must be run from the Print/Program command. Follow the steps below to produce the final formletter.

1. Turn on the printer and check the paper supply. Set the printer to Friction Feed if you are using your own stationery.

2. Clear memory with CTRL N, Y.

contd. on pg 52

SOFTVIEWS

by David Morganstein

How About a Nice Game of Chess?, Larry Atkin. I recently wrote reviews about Sargon III (now available on the Macintosh) and Chess 7.0, authored by Larry Atkin (known for mainframe chess programs which have won international competitions). Larry is now releasing, through Odesta, a modified version of his Chess 7.0. Having lost many times to 7.0, I began my examination of "How About a Nice Game of Chess?" with equal enthusiasm.

"Chess" is well suited for the objective of teaching someone how to play the royal game. Like its predecessor, 7.0, Chess can suggest moves, show squares which can be attacked by any piece, show squares which are attacking any piece and several other handy options. Although 7.0 allows you to adjourn a game at any point and save it to disk for later use, you cannot do this with Chess. You can watch Chess think, ask its advice for your move, or get it to play both sides.

For the learner, the disk includes a series of "movies". You can learn how the pieces move and the rudiments of tactics by watching the movie. At the end of the sequences, short quizzes are provided which have you make decisions and let you see the consequences. One problem using the movie sequences is that they must be started at the beginning and cannot be exited without restarting. It would be nice to repeat a practice without having to replay the movie.

You indicate your moves by using the keyboard, the paddles or, a new features in Chess, the mouse, to select a piece which is to be moved. To select a piece by paddle or keyboard, Chess steps through all the pieces. After you choose one, Chess steps through the legal moves which can be made by that piece and allows you to select the one you want. This procedure is very awkward for someone who knows how to play. The mouse, on the other hand, is a very natural way to select a a piece and the place to put it. For those without a mouse, I would have liked the more traditional option of entering the algebraic notation designating a move. That is easier and faster. One advantage of this technique for the learner is that it shows all possible legal moves. When using the mouse, the legal squares "light up" as the piece is moved through them.

Chess plays a good game. You can select any of nine levels of play (Chess 7.0 has seventeen levels!). The levels are time limited from one second per move (level 0) to two to six minutes per move (level 9). Most of the newest chess programs, like this one, Chess 7.0 and Sargon III, think while you think. Thus, if you select the move the algorithm expected, the program is one step ahead of you!! Both Chess and Chess 7.0 have an "opening book" of 7000 positions. If you remain in the book, Chess will move automatically.

I have very few criticisms of the package or the accompanying manual. A nice option would have been to permit the game moves or a given position to be printed. Also, it would have been nice to adjourn a game and save it temporarily to disk for later completion.

The Program: How About a Nice Game of Chess, Odesta, 930 Pitner, Evanston, IL 60202. $34.95.

Documax. Many software authors would like to write the next VisiCalc or Lotus 1-2-3. Unfortunately, most new software products look like new versions of old ideas. Occasionaly, something new surfaces. Doxumax tries to solve an important problem untouched by any program I have seen. When using your Apple for word processing, you quickly accumulate a large number of files with various versions of text material under constant revision. Documax is designed to organize and examine a disk full of text files and to quickly search through them for key words or phrases.

The main menu presents options to: compress a disk into a special high density format, convert the compressed data back to a standard text or binary format, examine the first few lines of selected files and then review an entire file if desired, search through selected files for one or more phrases or sort a directory.

The compress option allows you to store your files in a special format permitting over twice the quantity of text on a diskette. Because the compressed disk is in DOS 3.3 format, it can be copied with any standard backup program. This compaction is done by data encryption. The good news is the extra capacity obtained; the bad news is that to use the data outside of Documax, you must first "uncompress" the file.

Documax allows you to easily examine files. You may begin by inspecting the first several lines of a selection of up to twelve files. Alternatively, you can examine the entire contents of a selected file, line by line. You can jump to the beginning or end of the file or print it out. A powerful option allows you to specify a single phrase or several phrases (which can be 'and'ed or 'or'ed together) to use as search strings. Documax will examine the contents of up to twelve files, stopping to display the selection if found. This feature can be quite handy in locating an appropriate document.

The program is so straightforward that little documentation is needed. The accompanying manual is more than adequate to teach you how to operate the program and to answer questions. The manual has both a table of contents and an index. A fold-out "quick guide" provides an overview of its operation that will enable you to use Documax without even reading the manual. A demonstration disk included in the package contains a number of sample "demo" files to illustrate various options.

The program performed as the manual described. On a few occasions when I did something "wrong", the program provided me with ample indication of the problem and how to correct it. The screen prompts are clear and understandable. One possible point of confusion deals with the examination of DOS 3.3 files. You must indicate on a set-up screen whether the files you are looking for are in text or in binary form. This could trip you up if you know the file is there but failed to indicate the proper file type. That is, if the set-up is for text files, the directory will not show any binary files which might be present.

The success of this program will depend greatly on the market for its novel function. I think that the $175.00 price tag will scare away many potential users. While the program expands the storage capacity of Apple floppies and makes looking for the right file easier, I don't think many people will be willing to pay the price. Signum Microsystem, 120 Mountain Ave., Bloomfield, CT 06002, (203) 726-1911.

BRINGING BACK GAMESIG

by Ronald Wartow

David Morganstein appealed for the revival of GAMESIG in the November Journal. After expressing interest, he asked that I issue a call to the membership for such a purpose.

First, a few preliminary notes. To be perfectly honest, my //e would collapse if anything other than a game was booted. I sold my 80-column card after several personal appeals by the card to be used. Furthermore, I joined WAP because of its public domain game library, particularly to get the Original Adventure for $10. The Journal is largely incomprehensible to me. The Journal article on how to alter your Ultima III characters several months ago piqued my interest but, really, a disk zap!?!?

On the other hand, I have solved many graphic and all-text adventures (particularly Infocom), 2 of the 3 Ultimas and the 3 Wizardrys. I have come to the conclusion that the challenge, excitement, strategic, tactical, and diversionary considerations are good for me. I have written a few articles on computer gaming under pseudonym. After all, I am an over-40 lawyer who supposedly should "know better."

From the attendance and number of games making the rounds at the June swap meet, what follows will be familiar to many of you. However, for those of you unfamiliar with the concept of a computer game other than Pac-Man or Defender, the following hopefully will whet your appetite.

This month's Personal Computing contains a section on computer gaming with articles on the best games, and several literate articles to demonstrate that games contribute to computer competency, enhanced personal skills, and psychological development. In the same magazine an article discussed how, despite Apple's image attempts to the contrary, the Mac may be the ultimate game machine. In fact, about 50 games have been or are about to be released for the Mac. A May Medical Economics magazine contained an article by a doctor entitled "How Zork Took Over My Life." Some magazines, books, and cottage industries are devoted exclusively to the genre, providing strategy, maps, and hints. Finally, according to the April Softalk, the number 1 selling Apple program of all time is Wizardry -- a game.

There are three basic gaming categories -- adventure (including D&D role-playing), strategy (wargames, chess, simulations), and arcade. Adventure games, usually in a fantasy contest, contain logic puzzles, riddles, and extensive mapping (Wizardry I has 4000 locations) can be all-text or hi-res graphics. They take from a few to hundreds of hours to solve. Several innovative new games involve no typing and are played entirely by joystick (SunDog, see review elsewhere in this issue, and Seven Cities of Gold, just released for the Apple). Strategy games let you play chess against a program that, in its advanced modes, can beat most U.S. players (Sargon III). Wargames recreate classic or possible future global encounters (Tigers in the Snow, Germany 1985). Sophisticated simulations let you play the 1927 Yankees against the 1979 Orioles (Computer Baseball) or make a fortune in the stock market (Millionaire). You can even fly a plane in a program so realistic and accurate that it is FAA-approved (Flight Simulator II). Even the much-maligned arcade games are no longer just "shoot-em-ups" or dot eaters. Witness Lode Runner, with 150 screens which require complex strategy and Championship Lode Runner, the sequel, which even needs a save-game feature. Or, One-on-One, where Dr. J. and Larry Bird can shatter a backboard with a vicious slam dunk. Or, relive the Olymics in Summer Games.

Programming advances in design, artificial intelligence and playability are resulting in innovative prospects. For example, Wizardry IV, to be called "The Return of Werdna" can be joystick or mouse run. Ultima IV will be 8 times larger than the formidable Ultima III. King's Quest will have the character appear on screen and respond to your commands.

Anyone interested in participating may contact me between 11:30 and 1 at 275-8276 and between 6:30 and 11 at 654-4439. If enough interest is expressed by a solid 6-10 people, an organizational meeting will be held in Janmuary (Remember, December is the swap meet). If we can get it off the ground the goals of GAMESIG will be to provide the membership with the following services:

1. monthly reviews and/or articles on what is available that is worth buying by weeding out the considerable amount of market junk, and informing you of little-known classics and current games.

2. games to look forward to.

3. tips and strategies on solving and playing games.

4. stores where software is discounted.

5. demonstrations

LET ME HEAR FROM YOU!!!

LISA SIGNEWS

by John F. Day

The LISA special interest group will meet on December 8th at the USUHS following the SigMac meeting (SigMac will start at 9 a.m.). All LISA owners and any other interested people are invited to attend. The main topic will be a demonstration of two LISA products: "Art Department," by BPS, and "LISA Desktop Calendar," by Videx. Both the products work on the LISA 2/5 or 2/10 under the LISA 7/7 Office System.

The LISA SIG is very interested in contacting all LISA users. If you are a LISA user, owner, or are just interested in the LISA system, please contact either John Day at (301) 672-1721 or Gordon Stubbs at (703) 750-0224. If you are a member of WAP, and are listed in the club records as a LISA owner you should have already received a letter from the LISA SIG. If you haven't received a letter, give us a call so that we may add you to our list of "LISA people." If you know of other LISA groups around the world, please give us the address where they can be reached.

Let's make the LISA SIG a forum for your interests. Give us your ideas for future SIG meetings. We hope to be able to present more software demonstrations and discuss other important topics in the future.

SUNDOG: FROZEN LEGACY - A Review

by Ronald Wartow

SUNDOG: FROZEN LEGACY by FTL Games, San Diego, Ca.

The sole purpose of this article is to apprise the membership of something special now available in the Apple gaming world. Assume the following: the top adventure, strategy, and fantasy-role-playing game programmers have come to you for advice on their imminent combined effort for a new, exciting and challenging game. I expect your "wish list" for such a game would be all of the best elements of the classics of those genres.

1. Deep personal involvement and specific character interaction (Wizardry).

2. Riddles and puzzles (Zorks).

3. Crisp and clean hi-res graphics (Ultimas II and III) with Lisa-type windows (Wizardry III).

4. Ease of playability (Seven Cities of Gold).

5. Repeat playability, i.e., even when stated goal is achieved, the "play is the thing" the second time around (Wizardry).

6. Logical thinking (Rocky's Boots, Robot Odyssey I).

7. Familiar commerce and trade tactics (Millionaire).

You would also want to pay a reasonable price and get lots of play for that price. Finally, you would like to sit back in your easy chair with joystick in hand to do all of the above.

As you probably expected, I personally believe that SUNDOG accomplishes all of the above and more (even arcaders will enjoy it to a small extent). I highly recommend this game, but don't take my word for it. All reviews I have seen (InfoWorld and Computer Gaming World) have heaped lavish praise on this game, or should I say, gaming system of a life simulation. This quote comes from Softalk's June 1984 issue:

> "Create a character to pilot the SunDog, the beat-up space freighter you inherited. Fix the ship and engage in intergalactic trade, all with only a joystick. Macintosh-style windows. Epic scope. Be prepared to spend a while."

To bring all of this into perspective, you can do the following within one hour of booting SUNDOG:

1. Create your character.

2. Explore one of 50 cities on 18 planets in 12 star systems on foot, in a ground shuttle, or via a teleportation booth.

3. Fight muggers.

4. Sleep in a hotel.

5. Battle space pirates.

6. Buy cadcams, droids, fruits, furs, or whatever, at a city's stock exchange, then go to another city's stock exchange and sell what you bought at hopefully higher prices.

7. Consummate a deal for an "autoslew" or "concentrator" at a bar by dickering with the seller.

8. Anger a storekeeper or bartender.

9. Enter a bank and transfer money to your pocket or another star system bank.

10. Discover a new colony, then bring goods and cryogenics to the colony to start fulfilling your contract (the game's goal).

11. Have a drink or eat in a tavern.

12. Fly the SunDog after outfitting and retrofitting your ship at equipment stores.

13. Be told a joke by a bar patron or give money to an orphan.

14. Loot adversaries you have conquered.

In conclusion, this game is clearly the wave of the future. The reason that I've only peripherally mentioned the plot is that, at the moment, the playing of this incredible gaming system is too much fun. One last note: SUNDOG 2.0, a greatly enhanced version of the original SUNDOG released a few months ago, has just been released.

APPLE DISK ///: A Cautionary Note

by Jay M. Thal

Clinton Computer has been offering Apple Disk ///s for sale at $199. These drives have been modified for use on Apple][+ and //e and represent an excellent buy.

Basically, the modification boils down to plugging two of the sockets in the drive's 26 lead cable connector so that it can only fit one way into the controller board's 20 pin connector.

The cautionary note applies more to //e owners rather than][+s. Given the shape of the rear of the][+ case, the added 1/4 inch depth caused by the larger connectors can be accommodated tightly with the cabling emerging through the normal V slots.

The //e accommodation is another story. The //e's back panel is designed so that a normal disk][cable will fit through holes 1 and 2. Disk ///'s cable and connector will not fit through holes 1 or 2. Rather, they must be passed through hole 7 which takes up a valuable 4 inches, or so, of cable length. Further, because of the thicker upper portion of the //e's back case, drive 1's connector must first be attached to the controller card before it is forcefully but carefully inserted into slot 6.

Nevertheless, these are minor problems given the cost and quality of the drives. Now, does anyone make an adapter so that a disk][or /// can be used with the Apple IIc's infernal and non-standard 19 pin D connector? (Ed. Note: Yes, they are available through A.P.P.L.E. and elsewhere for approximately $15.)

CARD TRICK

by Merle Block

"CARD TRICK" is based on an old card trick, where twenty-one cards were dealt in three hands (or 'columns'). You selected a card, and called the number of the hand where the card was located. The hands were dealt, and after the third time, you were told the value of the card that you had selected.

Part of the 'secret' is that the hand with the selected card was always put in the middle of the deck of 21 cards before re-dealing. The associated program does the same thing.

Now for the description of the program, which is listed below.

The 'ONERR GOTO' command in line 120 takes care of alphabetic responses to the 'GET A'. Lines 140 and 150 put the card suit names into array SU$(). Line 160 puts the string names 1 to 14 into the card name array CD$(). Lines 170 and 180 rename the cards 1, 10, 11, 12, 13 to A, T, J, Q, K. That retains the one character spacing used in the printout.

The 'flag' in line 200 will count the number of passes through the program and jump out at line 310. Line 200 puts the numbers 1 through 52 into array S(). Line 210 picks 21 random numbers 'K' between 1 and 52. If the card has been used (i.e., S(K) is zero) then a different K is slected. Line 220 puts the card number S(K) into array A() and shows that the card has been used by putting zero into S(K).

Lines 240 and 250 display the cards in 3 columns. Lines 260 and 270 put the number of the selected column into Z. Line 290 and subprogram 380 are the heart of the program. The method used is from G. M. Mugford's article in the June 1984 WAP Journal, page 43. 'S' is the number of cards that are skipped to get the 7 cards from the selected column into the middle of the rearranged deck, before the deck is redealt. Line 300 puts the rearranged array B() into A() and check the value of the 'flag' A. Lines 340 to 370 display the 'guessed card' and finish the program.

Listing

```
100  REM   * CARD TRICK, PROGRAMMED BY M.BLOCK *
110  HOME : PRINT "GUESS YOUR CARD": PRINT "THE
     COMPUTER WILL GUESS YOUR CARD": PRINT "IF YOU
     FOLLOW INSTRUCTIONS": PRINT " ": PRINT " ": PRINT
     "PRESS ANY KEY TO DISPLAY CARDS."
120  PRINT " ": PRINT "DO  NOT  PRESS <RETURN> AFTER
     PRESSING  KEY- OR SELECTING CARD!": ONERR  GOTO
     260: GET A$
130  DIM S(52),A(21),B(21),SU$(52),CD$(52)
140  REM   * S()=CARD NUMBERS. A()=NUMBERS IN FIRST
     DECK. B()=NUMBERS IN REARRANGED DECK. SU$()=SUIT
     NAMES. CD$()=CARD NAMES. *
150  FOR I = 1 TO 13:SU$(I) = "DIA": NEXT I: FOR I =
     14 TO 26:SU$(I) = "HTS": NEXT I: FOR I = 27 TO
     39:SU$(I) = "CLB": NEXT I: FOR I = 40 TO
     52:SU$(I) = "SPD": NEXT I
160  N = 1: FOR I = 1 TO 52:CD$(I) =  STR$ (N):N =
     N + 1: IF N = 14 THEN N = 1
170  NEXT I: FOR I = 1 TO 40 STEP 13:CD$(I) = "A":
     NEXT I: FOR I = 10 TO 49 STEP 13:CD$(I) = "T":
     NEXT I
180  FOR I = 11 TO 50 STEP 13:CD$(I) = "J": NEXT I:
     FOR I = 12 TO 51 STEP 13:CD$(I) = "Q": NEXT I:
     FOR I = 13 TO 52 STEP 13:CD$(I) = "K": NEXT I
190  REM   * PUT 21 UNIQUE CARDS IN A() *
200  FLAG = 0: FOR I = 1 TO 52:S(I) = I: NEXT I:
     FOR I = 1 TO 21
210  K =  INT (52 *  RND (1) + 1): IF S(K) = 0
     THEN 210
220  A(I) = S(K):S(K) = 0: NEXT I
230  REM   * DISPLAY CARDS IN 3 COLUMNS.
240  PRINT : PRINT : FOR N = 1 TO 3: PRINT "COL.#";N;
     SPC( 6);: NEXT N: PRINT " ":FLAG = FLAG + 1
250  I = 1: FOR R = 1 TO 7: FOR C = 1 TO 3: GOSUB
     400:I = I + 1: NEXT C: PRINT : NEXT R: PRINT
     CHR$ (7)
260  PRINT : PRINT : PRINT "WHICH COLUMN CONTAINS YOUR
     CARD ?  (1,2 OR 3)  ";: GET Z: IF Z < 1 OR Z > 3
     THEN  PRINT : PRINT "MUST BE 1,2,OR 3": PRINT
     "PLEASE REENTER": PRINT " ": GOTO 260
270  PRINT Z: FOR I = 1 TO 500: NEXT I
280  REM  * PUT COLUMN WITH SELECTED CARD IN THE
     MIDDLE *
290  T = 1:S = 1 + 2 * (Z <  > 3): GOSUB 380:S = Z:
     GOSUB 380:S = 2 - (Z = 2): GOSUB 380
300  REM  * PUT ARRAY B() INTO A()
310  FOR I = 1 TO 21:A(I) = B(I): NEXT I: IF FLAG = 3
     THEN 340
320  GOTO 230
330  REM  * DISPLAY GUESSED CARD *
340  PRINT " ": PRINT " ": PRINT "YOUR CARD IS ";:
     I = 11: GOSUB 400
350  PRINT " ": PRINT " ": PRINT " ": PRINT "COMPUTERS
     ARE WONDERFUL !!!": PRINT  CHR$ (7); CHR$ (7)
360  PRINT " ": PRINT " ": PRINT "DO YOU WANT TO DO IT
     AGAIN (Y/N)?": GET Q$: IF Q$ = "Y" THEN 190
370  END
380  REM   * SUB TO REARRANGE DECK *
390  FOR F = S TO S + 18 STEP 3:B(T) = A(F):T = T + 1:
     NEXT F: RETURN
400  REM  * SUB TO PRINT CARD & SUIT *
410  L = A(I): PRINT CD$(L);" OF ";SU$(L); SPC( 4);:
     RETURN
```

Telecomm SIGNews contd. from pg 40

particular program requires a slot in some cases to cover all of the special keys of some terminals.

Another question concerned the difference between the Hayes Smartmodem and the Micromodem IIe. The answer was that the Hayes Smartmodem is an external modem and requires a serial card for the //e in order to work. The Micromodem IIe is a plug in modem and plugs directly into the //e. This modem has the serial circuits built right into the card. There followed a general discussion on the pros and cons of external versus internal modems.

Next was a discussion about the Anchor modem that WAP sells and some of the unusual characteristics of this modem. A break signal is now possible at 300 baud but is still not available at 1200 baud. The Anchor still cannot send the tones associated with the "*" and "#" on a Touch-Tone phone. Other than these items the modem works fine.

STOCKSIG NEWS
by Marvin Hass

****

STOCKSIG CALENDAR

Thursday, December 13, at 8:00 PM
WAP Office
8227 Woodmont Avenue (Room 202), Bethesda

"Computerized Fundamental Investing"
Ted Rosenberg, The Burney Company

****

The December 13 meeting of the STOCKSIG will feature Mr. Ted Rosenberg of The Burney Company who will discuss their computerized fundamental investment approach. The Burney Company was one of six outstanding money managers mentioned in the October issue of Changing Times. Their approach involves the selection of undervalued companies using a computerized method involving 60 parameters. The Burney Company is also unique in having no minimum size account (there is a $180 minimum fee). This presentation will focus on the basis of their approach and should be of interest to those who do their own stock selection as well as to those who might wish to delegate the responsibility to another party.

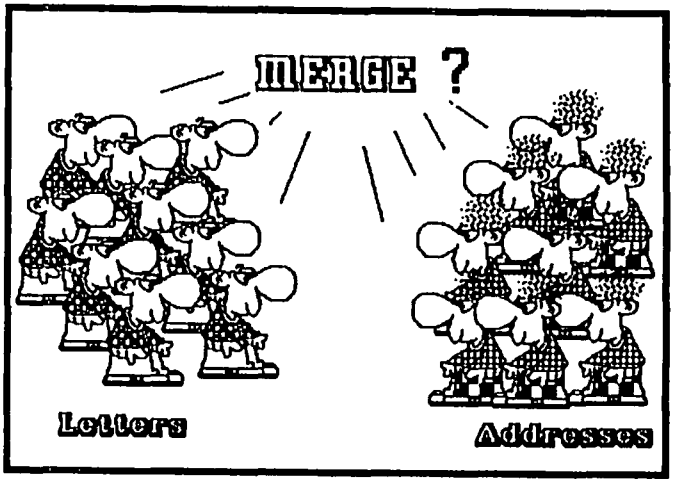
The October meeting heard Thomas A. Rorro, author of the new book, Assessing Risk on Wall Street. Mr. Rorro is an engineer, a part-time money manager, and chairman of the Computerized Investing SIG of the Washington Chapter of the American Association of Individual Investors. According to Mr. Rorro, risk analysis represents a new approach distinct from the more usual fundamental and technical methods. Stock prices fluctuations can be assumed to fit a log normal distribution as a first approximation. From the price, it is possible to calculate the profitability of various types of investments such as stocks, options, and convertible bonds. From the stock price distribution, it is possible to calculate the probability of achieving a certain profitability. It is Mr. Rorro's contention that convertible bonds under certain circumstances have a better risk-to-reward relationship than the underlying stock. Also properly chosen hedge positions can improve the risk-to-reward relationship. The approach is described in detail in Mr. Rorro's book (SOBARO Publishing Company, 7855 Colonial Village Row, Annandale, VA 22003, $15.00 to WAP members), and commercial software may become available in the future.

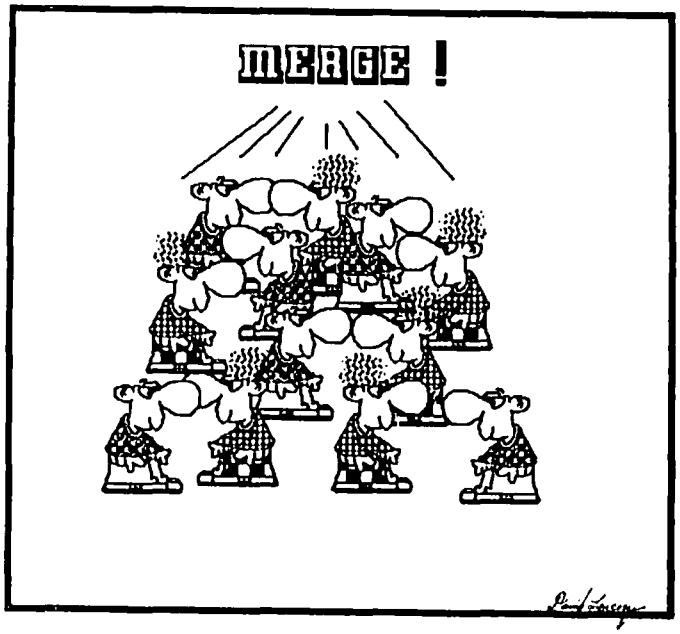
AW Hotline Calls Back contd. from pg 47

3. Type CTRL P, and respond with DO MERGE. (Note: All WPL programs are run by typing 'DO' and the name of the program to distinguish them from regular word processing files.)

4. Answer the questions asked by the program, date, name of base letter file, and name of address file.

5. Sit back and enjoy watching the computer work for you.





INAPPROPRIATE TECHNOLOGY: Part 2

Tales of the Bioplasmic Copier

by Jay M. Thal

Someone had a solution to my plight (see the November '84 WAP newsletter) that would no longer tie up my time or my computer for great lengths of time. As you probably remember, it was taking me 2 1/2 hours to prepare the DISABLEDSIG's monthly mailings.

The answer seemed simple enough, photocopy the text of my message on to postcards then turn the card over and photocopy the return address. All that would be left would be to add the mailing labels. Gone would be the yukky tongue that comes from licking 150 stamps. And, I wouldn't have to worry about the mess of a mimeograph machine.

First I typed out the text, then picked up the postal cards at the Post Office, and headed off to the trusty (?) photocopier.

Placing the copy on the glass I started feeding the cards through the machine. It still seemed simple. And then it was time to copy the second side. The cards started to feed through - and then they jammed! Despite the fact that I had tested out the double-sided copying, and the machine had just accepted the cards the first time through, it wouldn't work now.

For those of you who know little about photocopiers (like me), it is a three stage process. First an electrostatic charge is placed upon a drum in the same pattern as the message. (That's the part I consider magical - Arthur C. Clarke's Third Law: "Any sufficiently advanced technology is indistinguishable from magic.") Next, a fine dusting of powder adheres to the charge and is then laid down upon the paper stock. Lastly, the powder is fused to the paper by a heating process. And that's where things started to go wrong. The cards were getting dusted but they were not being transported to the fuser.

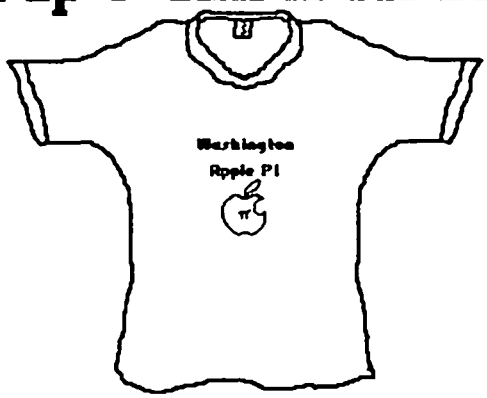
It was right after that that the copier became bioplasmic. Dropping the front panel of the copier, I bypassed the electrical interlock so that I could run the machine while it was open. Then, each time a card was fed in, dusted, and jammed I would slide the conveyor with the fuser aside, reach in and push the unfused card into the entrance to the fuser, close the conveyor, place a new card in and turn the machine on. That would result in the first card being fused and the new card jamming after dusting.

Imagine doing that 150 times. It was a less-than-perfect amalgam of man and machine. But, it's like taking a high jump in skiing - once you're committed you can't turn back. Whether or not you survive the first time, you don't have to do it again.

I've told this tale to others, and received several new suggestions. The first was to continue to use a photocopier, but to make multiple copies of the cards on 8 x 11 1/2 stock that will feed through the copier with greater ease. Then I can use a paper cutter to cut four cards from one sheet. Fingertips permitting, it could save some time - but then I'm back to licking stamps. Unless...I can build up the mailing list to 200 and print off the WAP bulk rate stamp.

Does anyone know people who would like monthly mailings of the DISABLEDSIG meetings?

Wap T-shirts for Sale



$7.00

Support your club! Buy a WAP T-Shirt. Background color is "Apple" tan - lettering is in green and red. Available at the WAP office or by mail for $7.00 each. If ordering by mail add $1.50 postage and handling per order. State size: Adults Small, Medium or Large; Children's Small, Medium or Large (the sizes run a little on the small side). Send check or money order payable to Washington Apple Pi and mail to Washington Apple Pi, 8227 Woodmont Avenue, Suite 201, Bethesda MD 20814.

MOUSEMAINT or, What I Learned From Reading Between the Lines

by Jim Graham

It's a long standing joke that, "if all else fails, read the manual..." And so it seems also for the mouse. This is a Macintosh experience, but it relates also to the mouse for the Apple][series, and even others.

It had been several months since I started using the Macintosh, and gradually the "fine-tune" seemed to be disappearing from the mouse interface. Sometimes the mouse would seem to stick, or jump, or be hard to move in small increments. At first I thought it was the nature of the wooden desk surface. It would occur randomly, almost like an intermittent problem inside the mouse hardware. I had long since learned to take the mouse apart, as shown on pages 136-137 of the Mac-Manual, and I had cleaned the ball occasionally. My problem was that I didn't know what clean rollers for the mouse are supposed to look like. I had never looked at them until they started giving me problems. True, they did have a rough-looking substance on them, but I was originally inclined to think that was part of the "friction" ensuring the rubber ball and the rollers move together.

As a last ditch effort, I decided to try the alcohol swab method clearly described in the manual. I had been reluctant because I didn't want to damage the roller surface material, whatever it was. The rough material was fairly thick, and didn't come off easily with the alcohol swab, but it did come off - and what a difference! The one thing that the manual doesn't state is that the three rollers - two rubber and one metal - should be PERFECTLY SMOOTH. It appears that you won't hurt them by trying to take off anything that makes them unsmooth. That may be intuitive to some, but it wasn't to me.

Having done this only several weeks ago, I can see that they are beginning to need it again. The frequency of cleaning probably varies by the conditions under which you use the Mac, and especially the nature of the surface the mouse rolls on. But once you know what they should look like, cleaning the rollers is a snap.

So, the message here is not simply "read the manual". Sometimes, even read between the lines. It can pay off...

WAP FINISHES THE MARINE CORPS MARATHON

by Robert C. Platt

In some marathons, marching bands or pretty cheerleaders greet the runners as they complete 26.2 miles. But for Washington's ninth annual Marine Corps Marathon, Apple //e's were dramatically poised at the finish line.

When it came time for the Marines (who are notoriously efficient) to select a system to record the finishing times for the over 12,000 runners who participated in this year's race, they decided to use Apples. Naturally, WAP was the logical place to recruit volunteers to loan them equipment.

The Apple-based marathon recording system was designed by Jack Moran, a Professor of Aerospace Engineering at the University of Minnesota. Jack is both a dedicated Apple user and marathon runner. He developed the system for the Twin-Cities Marathon that he directs in Minneapolis.

Before the race, Jack printed out bar codes representing each runner's number on his Epson printer. As each runner crossed the finish line, the exact moment was recorded on one of two Apples by typing a slash. These two Apples were running a program that read a Mountain Computer clock card with each slash and recorded the finishing times on a disk file. Marines then collected bar codes from each runner in the order that the runner crossed the finish line. Six other Apples equipped with Advanced Business Technology BarWands and interface cards then read the runners' numbers into a second disk file. The final step was for yet another Apple equipped with a hard disk to read the floppy disks from the other Apples and merge the two types of files to produce one large file that has each runner's number and finish time. The final file was then downloaded via modem to the Marine Corps' mainframe to generate award certificates. The Apples also printed interim results on schedule for release to the press.

Needless to say, a marathon finish line is a high-pressure operation. However, all Apples functioned perfectly during the nine hours it took for the entire herd to cross the finish line. (The smooth operation is a tribute to both Jack's bomb-proof data entry routines and the conscientiousness of the freshly-trained marine operators.) The Apples printed interim results on schedule for release to the press. The only mishap occurred when one floppy disk was misplaced for a while during the file merge operation.

One interesting lesson from this is that with relatively inexpensive peripheral cards, an Apple can replace far more expensive single purpose electronic equipment. In prior years, the marines used a special clock that generated a paper-tape output of all finishing times. The Apple not only generated results just as accurate, but it also produced machine-readable results.

One of the rumors circulating in the days before the race was that WAP's Tom Warrick and Bruce Field had replaced the Applesoft ROMs in the equipment to be loaned with modified versions that would assure your author with a first place finish. Both Bruce and Tom deny these unsubstantiated accounts. We will never know the truth, because Jack's system runs under Apple Pascal rather than Applesoft. His printouts showed that Brad Ingram won the race in 2 hours and 16 minutes, a full hour ahead of the first WAP finisher. Having been defeated so decisively, I've decided to explore the secrets of patching SYSTEM.APPLE and SYSTEM.STARTUP during the next twelve months. Meanwhile, other WAP members planning on running next year should check out Disk 143 which claims to have "all the software you need to convert a sedentary Apple user into a marathon runner."

WASHINGTON APPLE PI DISKETERIA MAIL ORDER FORM
Software for Creative Living

Disks from Washington Apple Pi's Disketeria are available for purchase. This form is only for ordering disks that you want mailed to you.

5 1/4" DISKETTES: - Members $ 5.00 ea.; Non-members $ 8.00 ea. } Plus $1.00 ea. postage
3 1/2" " - Members $ 7.00 ea.; Non-members $ 10.00 ea. } up to a maximum of $5.00.

DOS 3.2
( ) Volume 1 Utilities I
( ) Volume 2 Utilities II
( ) Volume 3 Games I
( ) Volume 4 Games II
( ) Volume 5 Games III
( ) Volume 6 Games IV
( ) Volume 7 Games V
( ) Volume 8 Utilities III
( ) Volume 9 Educational I
( ) Volume 10 Math/Science
( ) Volume 11 Graphics I
( ) Volume 12 Games VI
( ) Volume 13 Games
( ) Volume 14 IAC Utilities IV
( ) Volume 15 Games VII
( ) Volume 16 Utilities V
( ) Volume 17 Graphics II
( ) Volume 18 Educational II
( ) Volume 20 Music
( ) Volume 21 Apple Orchard
( ) Volume 22 Utilities VI
( ) Volume 23 Games VIII
( ) Volume 24 Games IX
( ) Volume 25 Utilities VII
( ) Volume 26 Stocks/Investments
( ) Volume 27 Math
( ) Volume 28 Planetfinder
( ) Volume 29 Utilities VIII
( ) Volume 30 Games X
( ) Volume 31 Plot Utilities
( ) Volume 32 Games XI
( ) Volume 33 Accounting
( ) Volume 34 Solar Tutor
( ) Volume 35 Garden Management
( ) Volume 36 Games XII
( ) Volume 37 Utilities IX
( ) Volume 38 Games XIII
( ) Volume 39 IAC VII
( ) Volume 40 IAC VIII

DOS 3.3
( ) Volume 41 IAC 25 Mach. Lang. Util.
( ) Volume 42 One Key DOS ***
( ) Volume 43 IAC 29 Utilities H
( ) Volume 44 Utilities I
( ) Volume 45 Diversi-Copy ***
( ) Volume 70 Business/Math/Statistics
( ) Volume 71 Music
( ) Volume 72 Keyboard Games
( ) Volume 73 Text Adventure Games
( ) Volume 74 Paddle Games
( ) Volume 90 Spreadsheet C Genl. Bus.
( ) Volume 91 Spreadsheet D Investment
( ) Volume 92 Spreadsheet E Bus. Recd.

( ) Volume 100 Utilities A
( ) Volume 101 Utilities B
( ) Volume 102 Games A
( ) Volume 103 Merry Christmas
( ) Volume 104 Business A
( ) Volume 105 FIG-FORTH/Utilities

DOS 3.3 contd.
( ) Volume 106 Science Engineering
( ) Volume 107 Games B
( ) Volume 108 IAC 10 (Graphics)
( ) Volume 109 IAC 11 (Applesoft Tutorial)
( ) Volume 110 Personal/Education
( ) Volume 111 Games C
( ) Volume 112 Utilities C
( ) Volume 113 Business B
( ) Volume 115 IAC 12/13 Misc.
( ) Volume 116 IAC 14 MicromodemII
( ) Volume 117 Picture Packer
( ) Volume 118 Utilities D
( ) Volume 119 IAC 15 Misc.
( ) Volume 120 IAC 16 Misc.
( ) Volume 121 WAPABBS 1.1 Doc. **
( ) Volume 122 IAC 17 Misc.
( ) Volume 123 French Vocabulary
( ) Volume 124 Utilities E
( ) Volume 125 IAC 18 Misc.
( ) Volume 126 Sights and Sounds
( ) Volume 127 Math/Science
( ) Volume 128 Games D
( ) Volume 129 GLAQ
( ) Volume 130 Diversi-DOS ***
( ) Volume 131 Personal/Educ. 2
( ) Volume 132 IAC 19 - Utilities F
( ) Volume 133 IAC 20 - Pascal & DOS 3.3
( ) Volume 134 New Members Disk
( ) Volume 135 WAPABBS 1.1 Disk 1 **
( ) Volume 136 WAPABBS 1.1 Disk 2 **
( ) Volume 137 IAC 21 Spreadsheet A
( ) Volume 138 IAC 23 Utilities G
( ) Volume 139 IAC 24 Education 3
( ) Volume 140 Education 4
( ) Volume 141 Special Data Bases
( ) Volume 142 IAC 28 Pinball Games
( ) Volume 143 Sports
( ) Volume 144 IAC 27 Applesoft Prog.
( ) Volume 145 Apple Logo Tool Kit
( ) Volume 146 Apple Logo Sample Prog.
( ) Volume 147 Logo Documentation
( ) Volume 150 EDSIG1 (Elem. Math)
( ) Volume 151 1983 Tax Template
( ) Volume 152 IAC 31 Miscellaneous
( ) Volume 153 Investments A
( ) Volume 154 Investments B
( ) Volume 155 IAC 33 Miscellaneous
( ) Volume 156 IAC 35 Applesoft-AW//e
( ) Volume 157 IAC 36 Arcade Games
( ) Volume 158 Apple Logo Programs

Eamon Series
( ) Volume 180 Dungeon Designer
( ) Volume 181 Beginners Cave
( )*Volume 182 Lair of Minotaur
( )*Volume 183 Cave of the Mind
( )*Volume 184 Zyphur Riverventure
( )*Volume 185 Castle of Doom
( )*Volume 186 Death Star
( )*Volume 187 Devil's Tomb
( )*Volume 188 Caves of Treas.Isl.

Eamon contd.
( )*Volume 189 Furioso
( )*Volume 190 The Magic Kingdom
( )*Volume 191 The Tomb of Molinar
( )*Volume 192 Lost Isl. of Apple
( )*Volume 193 Abductor's Quarters
( )*Volume 194 Quest for Trezore
( )*Volume 195 Underground City
( )*Volume 196 Merlin's Castle
( )*Volume 197 Horgrath Castle
( )*Volume 198 Deathtrap
( )*Volume 199 The Black Death
( )*Volume 200 The Temple of Ngurct
( )*Volume 201 Black Mountain
( )*Volume 202 Nuclear Nightmare
( )*Volume 203 Feast of Carroll
( )*Volume 204 The Master's Dungeon
( )*Volume 205 The Crystal Mountain
( )*Volume 206 The Lost Adventure
( )*Volume 207 The Manxome Foe

Pascal (See also Volume 133)
( ) Volume 300 PIG0: ATTACH 1.1/BIOS
( ) Volume 301 PIG1:
( ) Volume 302 PIG2:
( ) Volume 303 PIG3:
( ) Volume 304 PIG4:
( ) Volume 305 PIG5:
( ) Volume 306 PIG6:
( ) Volume 307 PIG7:
( ) Volume 308 PIG8:
( ) Volume 309 PIG9:
( ) Volume 310 PIG10:
( ) Volume 311 PIG11:
( ) Volume 312 PIG12:
( ) Volume 313 PIG13: Guerilla Guide
( ) Volume 314 PIG14:

CP/M
( ) Volume 401
( ) Volume 402
( ) Volume 403
( ) Volume 404
( ) Volume 405
( ) Volume 406 ZCPR2 Install
( ) Volume 407 ZCPR2 Documentation
( ) Volume 408 ZCPR2 Utilities
( ) Volume 409 Modem 730

Forth (See also Volume 105)
( ) Volume 700 Assembler/Disassembler
( ) Volume 701 Full Screen Editor
( ) Volume 702 GoForth Tutorial

Macintosh - @$7.00 (see above)
( ) SigMac Disk 1 MS-BASIC Pgms
( ) SigMac Disk 2 Atkinson's Goodies
( ) SigMac Disk 3 Fonts
( ) SigMac Disk 4 MS-BASIC Pgms
( ) SigMac Disk 5 Desk Accessories
( ) SigMac Disk 6 Mac Paintings
( ) SigMac Disk 7

* Volume 181 required with these disks. ** Vols. 121, 135, 136 must be purchased together.
*** Use of this disk requires sending money to the author ($30 for Diversi-disks and $9 for One Key Dos.)

(NOTE: ALLOW 3 TO 4 WEEKS FOR MAILING.) Total Order = ___ disks.; postage $___; Total amount enclosed $

NAME _____

ADDRESS _____

CITY, STATE ZIP _____

TELEPHONE _____ WAP MEMBERSHIP NO._____

Make check payable and send to:

Washington Apple Pi, Ltd.
Attn. Disketeria
8227 Woodmont Avenue, Suite 201
Bethesda, MD 20814

DATE _____

WAP TUTORIAL REGISTRATION

The following four WAP tutorials are being offered on Tuesday evenings from 7:30 to 9:00 PM, at the office, 8227 Woodmont Avenue, Room 202, Bethesda, MD. (The tutorials start promptly at 7:30; if you bring your computer please arrive 15 minutes early to set up.) You may sign up for any or all of the series. They are designed for the "beginner" and will be repeated monthly. A detailed outline of the tutorials was given in the September 1984 WAP Journal. -

There will be no tutorials in December.

( )	January 8	- INTRODUCTION TO APPLE COMPUTER HARDWARE	( )	February 5
( )	January 15	- HOW TO USE YOUR APPLE SOFTWARE	( )	February 12
( )	January 22	- BEGINNING APPLESOFT BASIC	( )	February 19
( )	January 29	- INTERMEDIATE APPLESOFT BASIC	( )	February 26

The fee for each tutorial is $10.00 with an Apple, monitor and disk drive, $15.00 without (monitors available for 1st 5 registrants - call office). Please note that WAP does not have equipment for you to use; if you do not bring your own, you will have to look over someone's shoulder.

___ Tutorials at $10.00 (with equipment)
___ Tutorials at $15.00 (without equipment)

Please check the desired tutorials and return this form with fee(s) made payable to Washington Apple Pi, Ltd. to:

Washington Apple Pi, Ltd.
Attn. Tutorials
8227 Woodmont Avenue, Suite 201
Bethesda, MD 20814

Name _____

Daytime Phone _____ Evening Phone _____ Total Enclosed $ _____

INDEX TO ADVERTISERS

INDEX TO AUTHORS

Compare COMPUTER WARE UNLIMITED to Anyone else!

SHOP NOW FOR THE HOLIDAYS!

"Specializing in Apple Computer Software"

ACCOUNTING

The Accountant	$67
The Accountant 5.0	88
AR/AP/GL/PR (Cont.)	ea. 165
AR/AP/PR (BPI)	ea. 260
Church Management (BPI)	325
General Accounting (BPI)	260
Home Accountant	51
Inventory Control (BPI)	260
Tax Advantage	48

COMMUNICATIONS

ASCII Express Pro	88
Data Capture 5.0	50
P-Term Pro	88
Visiterm	72
Z-Term Pro	102

DATA BASE

DB Master	160
Datastar (req. CP/M)	195
ncredible Jack	93
FS: File	86
PFS: Graph	86
PFS: Report	86
Visidex	175
Visifile	175

EDUCATION

ABC - Alphabet Beast & Co.	21
Algebra 1, 2, 3, 4	ea. 29
Algebra 5 & 6	36
Alphabet Zoo	21
Basic Skills	14
Body Awareness	35
Bouncing Kamungas	21
Cause & Effect (red/blue)	35
Cogito	24
Compumath/arithmatic	36
Compumath/decimals	36
Compumath/fractions	36
Computer SAT	55
Facemaker	21
Fact or Opinion (red/blue)	35
Foreign Languages	14
Grandma's House	21
Gertrude's Puzzles	31
Gertrude's Secrets	31
Hands on BASIC Prog.	56
KinderComp	21
Learning w/Fuzzywomp	21
Lucky's Magic Hat	$ 28
Mastering the College Bds.	121
Mastering the SAT	104
Mastertype	28
Math Blaster	34
Math & Social Studies	14
Micro Habitats	24
Micro Speed Read	87
Moptown Hotel	28
Mr. Cool	21
Number Farm	21
PSAT Word Attack Skills	36
Rocky's Boots	34
SAT Word Attack Skills	36
Speed Reader II (Davidson)	48
Spellicopter	28
Sticky Bear ABC	28
Sticky Bear Basket Bounce	28
Sticky Bear Numbers	28
Sticky Bear Opposites	28
Sticky Bear Shapes	28
Terrapin Logo	68
Typing Tutor	20
Typing Tutor III	35
U.S. Geography	28
Wizard of Id's Wiz Type	24
Word Attack	34
World Builders	24

LEISURE

A.E.	24
Arcade Machine	42
Beyond Castle Wolfenstein	24
Checkers	34
Chess 7.0	48
Computer Ambush	43
Computer Baseball	29
Computer Gin Rummy	18
Computer Quarterback	29
Cosmic Balance	29
Cranston Manor	24
Dark Crystal	26
Deadline	35
Decathlon	24
Enchanter	28
Epidemic	25
Galactic Gladiator	29
Infidel	31
Knight of Diamonds	24
Legacy of Llylgamyn	28
Lode Runner	$24
Napoleon's Campaign	43
Odin	34
Old Ironsides	28
Planetfall	28
Prisoner II	24
Pursuit of Graf Spee	43
Questron	36
Rails West	29
Run for the Money	35
Sargon III	35
Seastalker	28
Shattered Alliance	43
Sorceror	31
Southern Command	43
Starcross	35
Suspended	35
Tigers in the Snow	29
Time Zone	68
Transylvania	25
Tubeway	21
Ultima II	42
Ultima III	42
Ulysses	23
War in Russia	58
Warp Factor	29
Witness	28
Wizard and the Princess	23
Wizardry	34
Zork I	28
Zork II, III	ea. 31

SPREADSHEET

Calcstar (req. CP/M)	132
Flashcalc	71
Magicalc	107
Multiplan	153
Visicalc	175
Visitrend/Visiplot	71
Visischedule	107

UTILITY/GRAPHICS

Alpha Plot	20
Apple Mechanic	27
Applesoft Compiler	137
Complete Graphics System	56
Data Plot	41
DOS Boss	17
Double Take	24
Doubletime Printer	68
Fontrix	$ 51
Frame Up	20
Graphics Magician	42
Graftrix	44
Hi-Res Secrets	90
Merlin	44
Munch-a-Bug	28
Picture Builder	28
Printographer	28
Pronto DOS	20
Routine Machine	28
Silicon Salad	17
Super Disk Copy	21
Utility City	20
Zoom Graphics	35

WORD PROCESSING

Addressbook/Mail List	34
Bank Street Speller	49
Bank Street Writer	49
Format II	102
Homeword	48
Magic Window II	107
PIE Writer	103
Print Shop	35
Sensible Speller	85
Screenwriter II (Pro.)	88
Spellstar (req. CP/M)	67
Super Text Professional	119
Super Text Home/Office	85
Word Handler	55
Wordstar with starcard	325

MISCELLANEOUS

Dollars and Sense	69
Micro Barmate	28
Micro Cookbook	28
Appetizers	9
Soups & Salads	9
Desserts	9

DISKETTES

5¼" SS/SD WABASH	17.50/10

MACINTOSH SOFTWARE

Available - Call For a Quote

ORDERING INSTRUCTIONS

Specify II +/IIe/IIc

Maryland residents add 5% sales tax • Shipping and handling $3 per order

No charge's or COD's • Prices subject to change • Call for items not listed

——Thank You——

P.O. Box 1247 · Columbia, MD 21044 · (301) 854-2346

WASHINGTON APPLE PI, LTD.
8227 Woodmont Avenue, Suite 201
Bethesda, MD 20814

ADDRESS CORRECTION REQUESTED

BULK RATE
U. S. POSTAGE
PAID
PERMIT # 5389
Silver Spring, MD
20910

A REFURBISHED DAISY WHEEL PRINTER
FOR PERSONAL COMPUTER USERS AND SMALL BUSINESSES.

Three-In-One Offer! Just $598 (Includes On-Site Warranty)

- A 30 cps **letter-quality** printer
- A timesharing keyboard terminal (when modem equipped)
- A Selectric*-style keyboard typewriter

AJ daisy wheel printer terminals are renowned for exceptional performance, high reliability, and applications versatility. Now you can have all this for only $598** in our special limited offer.

- Optional 45 characters per second
- Changeable type faces
- Full ASCII keyboard with numeric pad
- High resolution X-Y plotting
- Complete electronic forms control
- 256-character buffer
- Asynchronous RS-232 interface
- Printwheel, ribbon cartridge, and cable included
- 30-day parts/labor warranty

We service IBM personal computers and peripherals at your home or office.

And you can choose from a list of options including forms tractor, pin-feed platen, paper trays, side shelves, extra printwheels, APL keyboard and 2K buffer.

For information telephone
Sean Belanger
(301) 840-5700

**Suggested selling price, excludes options and is subject to change without notice. Model shown includes certain options. Offer available only in the contiguous U.S.

*Selectric is a trademark of IBM.

ANDERSON JACOBSON
8653 Grovemont Cir.
Gaithersburg, MD 20877
-4191