$1

Washington Apple Pi

Volume 2 May 1980 Number 5

Highlights

In This Issue

ComputerLand® and apple II

For the best in personal computing

SOFTAPE™ Personal Software™

D. C. Hayes Associates, Inc.
MICROCOMPUTER PRODUCTS

CENTRONICS®

Mountain Hardware, Inc.

Integral Data Systems, Inc.

MUSE

THE SOURCE
Information Utility

houston instrument

Heuristics INC.

Automated Simulations

SANYO

ComputerLand®
We Know Small Computers.
ComputerLand/Tysons Corner
8411 Old Courthouse Road at Rt. 123 — 893-0424

VISA

master charge

Officers & Staff

President	-John L. Moon	(202) 332-9102
Vice President	-Bernard Urban	(301) 229-3458
Treasurer	-Robert Peck	(301) 468-2305
Secretary	-Genevie Urban	(301) 229-3458
Members-at-Large	-Mark Crosby	(202) 488-1980
	-Sue Eickmeyer	(301) 953-7355
	-Sandy Greenfarb	(301) 674-5982
Editor	-Bernard Urban	(above)
Associate Editor	-Rich Wasserstrom	(703) 893-9147
Librarian	-David Morganstein	(301) 972-4263

Washington Apple Pi
P. O. Box 34511
Washington, D.C. 20034
(301) 468-2305

Membership dues for Washington Apple Pi are $12.00 per calendar year. If you are interested in joining our club, call our club number and leave your name and address. An application form will be mailed to you. Or if you prefer, write us at the above PO Box.

EDITORIAL

SOME RAMBLING THOUGHTS FROM THE EDITOR.

We've had our elections and as of this writing, I don't know how they have turned out. However they may, I am reassured that Washington Apple Pi will come out OK. We have grown to the point that we will continue to grow. I know that we shall lose (and unfortunately may have already lost) some of our best first members. They may feel that they have outgrown us, Although I disagree, I feel that this is inevitable. However, there are many who have stayed with us and contributed from the beginning, some who have just begun to contribute, others who have just discovered our existence, and still others yet to come. As my survey a few months ago indicated, we the APPLE users of the Greater Washington, D.C. area are an interesting bunch. It proves to me that microcomputers are no longer the province of the elite--the professional computerist, the scientist or the engineer. It also proves to me that computers will become more and more the tool of the "average" American and that we are in the vanguard.

As you know, as guest of Apple Computer, Inc., I was honored to be among the first to witness the unveiling of the APPLE III. Apple had conducted several special sessions in different parts of the country for the benefit of the APPLE Dealerships. but just prior to the N.C.C. in Anaheim, California, they held a special showing for the elected board representatives and officers of the INTERNATIONAL APPLE CORE. Those of you who attend our next session on May 31 may obtain copies of many of the handouts that were supplied to me by Apple. Paul Sand of Computerland Rockville, is scheduled to brief you in detail at our June meeting on what he learned at a lengthy tutorial in Newark, N.J.

But let me say this. Although APPLE III is now unveiled, and will soon be on display at computer stores in our area, I personally have been reassured by representatives of Apple in attendance on May 18, 1980, that APPLE II is still their main computer system and will continue to be so in the near future. I had expressed concern that the III represented to me a shift away from the hobbyist and towards the small-to-medium business firms. They responded that the II will be their mainstay for some considerable time. Witness, if I might, the many new items that would be announced at their booth the next (opening) day of the N.C.C.

I went to their booth and saw much. They now include in their bag of tricks for the APPLE II, Pascal, Fortran, Pilot, the Silentype Printer, DOS 3.3, AppleWriter, and several business packages. They have been making considerable progress in the sale of II's for educational purposes. Added to Minnesota are British Columbia and North Carolina. All in all, I am convinced that the II is here for at least several more years. The Woz said this in March and it was repeated in May.

But what of the III you say? In a nutshell, they've performed some hardware sleight of hand and made a "6502a" able to count to 128K and compute at twice the speed of a 6502. They give you 80 columns, upper and lower case and 16 Hires modes. Do you want to change your character set - well now you can, even to making your characters look like birds in flight! Do you want to use your old II programs? Well you can, in the emulate mode (except that programs which make use of the game paddles will have to be modified). You can read your old II programs in on the built in floppy. You can daisy chain three other floppies as well. You have an exotic sophisticated operating system to let you write programs like the Woz himself. It takes care of the nitty gritty and leaves the driving to you. Even the reset button has been moved and won't work unless you also push control simultaneously. But - you will need a 12-inch black and white monitor to read the 80 columns upper and lower case without eye strain. Also for color you will need a monitor, and better still an RGB monitor (the best and the brightest). Great software is in the works, like the Word Painter and Visicalc III. But starting at $4300 and building to $7800 or more we'll all have to come up with better excuses to convince our wives or husbands or whomever that we absolutely must trade in our II's for the III. Again, where to see them? - at your friendly neighborhood dealer soon, if not now. But don't expect delivery for several months yet. Now let us get back to our beloved II. We ain't scratched the surface yet.

Ken Silverman took advantage of the presence of most of the IAC Board representatives and all of the Officers to conduct a special session of the IAC. items of note include:

1. Acceptance by the Board of Val Golding's budget and publication plan for the next three issues of the "APPLE

contd.

ORCHARD".

2. A subcommittee chaired by Mike Weinstock, Vice President of IAC, was created to investigate other publishers and alternative arrangements for soliciting ads.

3. Time and place of the 1981 meeting still to be determined.

4. From now on, Secretary Joe Budge will, with the help of paid individuals, send IAC Ap Notes and related materials directly to the member clubs. Board members will develop alternative methods for networking their regions and providing services and feedback to their clubs.

5. The Constitution and Bylaws have been reworked somewhat, but have yet to be reviewed by Apple's lawyers.

I had expected to return with the video tapes of the session held last March. However, Joe Alinsky of L.A. had some difficulties, but will be mailing out copies to each board member. I hope to have it for your viewing soon.

minutes

The Washington Apple Pi meeting of April 26, 1980 was called to order at 9:45AM by the President. The first order of business was the nomination of officers. Following is the resulting slate:

President	- Bernie Urban
Vice President	- Dave Efron
	Rich Wasserstrom
Secretary	- Dana Schwartz
Treasurer	- Bob Peck
Members at large (3 to be elected)	- Scooter Conrad
	Mark Crosby
	Theron Fuller
	Sandy Greenfarb
	Tom Jones
	Hersch Pilloff

Short resumes were requested of all candidates, and the ballot was scheduled to be mailed out by May 9, with a return deadline of May 27. New officers will take over on June 1.

A motion was made and carried to have the May meeting on the 5th Saturday due to the Memorial Day Weekend. Our Librarian, Dave Morganstein called for volunteers for packing and mailing diskettes. One response was received at that time. Treasurer Bob Peck said that in a group purchase the cost of a Language Board would be $415. Sue Zakar reported that her Applesoft card was fouled up in moving and suggested that members be careful to push down all contacts when moving the APPLE. Sandy Greenfarb said that there are 11 IAC disks in the offing and they should be en route soon. Hersch Pilloff discussed the DOS append error and urged members to read carefully "the fix".

The main program was presented by Tom Woteki on the Pascal Language System. He brought along his APPLE and gave an informative discussion and demonstration.

EVENT QUEUE

Washington Apple Pi meets on the 4th Saturday of each month at 9:30AM at George Washington University School of Engineering, 23rd and H Streets, NW. (However, due to Memorial Day Weekend, the May meeting will be held on the 5th Saturday.) The June meeting will be on June 28.

NOVAPPLE meets on the 2nd Thursday at 7:30PM at Computers Plus in Franconia, and on the 4th Thursday at 7:30PM at Computerland of Tysons Corner.

LETTER TO THE EDITOR - DOS APPEND "BUG"

Dear Pi,

I read with interest (and concern) your article relating to the "bug" in DOS on the append function.

In late December of last year I discovered this problem in my office and determined its cause. We devised a special-purpose fix to solve our problem and subsequently avoided the use of append (what other evils lurked there?).

I then decided to pass on the good news to the source of the problem and called the Apple Hotline. The person to whom I spoke seemed puzzled at first. She later supposedly spoke to the people who wrote DOS and reported that they did not consider it a bug. She knew of no plans to fix the problem but assured me that Apple would inform its customers of the situation. Please note that some five months have passed including a new release of DOS (3.2.1) but apparently Apple is not beating its drums very loudly. If Apple is suppressing this or failing to publicize it for marketing reasons, I feel that they are not properly supporting those who have made them what they are today.

How about it, Apple - everyone makes mistakes? Just show us your're human, too.

Ron Dreyfus

Classifieds

FOR SALE: Anderson Jacobson acoustic coupler. Perfect condition. H. Pilloff - 292-3100.

I am considering lease of an APPLE II+ with 2 disks, Sanyo monitor and Centronix 700 printer for a business/personal application using "The Cashier" software package. I would welcome any advice from W.A.P. and its readers. Keith Davis, P.O. Box 57177, Washington, D.C. 20037.

Classified ads accepted from members 50 words or less at no charge provided the material is obviously non-commercial. Submit your classified to: CLASSIFIED ADS, PO BOX 34511, WASHINGTON, DC 20034.

NOVAPPLE

OFFICERS

President - Phil Eastman
Vice President - Nick Cirillo
Secretary - Gerald Eskelund

MINUTES FOR APRIL 24, 1980

The meeting opened with a few words from the President, who indicated the program would be presented by Watts Hill, a NOVAPPLE member. Before the program began several announcements were made. Program Chairman Theron Fuller indicated that he had documentation on Forth interest groups. The President said he would like to start a disk program library for NOVAPPLE and asked that all members contribute original programs for the disk. The Treasurer gave his report indicating that we currently have $168.35 in the bank with $18 collected at this meeting for a total of $186.35. Someone asked if we should get membership cards but the concensus was that this would be of no benefit to the members unless they could get a discount at the various local stores. This might be explored at a later date. Colonel Bill Harmon presented a letter he had received from NIBBLE concerning the trac program in their original issue. It seems the system is interface dependent. If anyone is interested in the fix you can contact the Secretary or Colonel Harmon. A copy of the letter was also given to the President of Apple Pi.

Watts Hill presented the program on Total Home Control. He explained how he has his APPLE set up to control his whole house. He not only controls the lights and appliances but he also has his heat and hot water under computer control. He explained the kinds of circuits needed and presented the wiring diagrams he used for his hot water system. A dedicated clock board is needed for the operation, at least 32K and a disk. You can run programs at the same time your APPLE is in control of your house but you should not run programs greater than 31K or the system will crash. The clock board will check to see what the status of the unit is every second but you will not see the results since it is so fast. When it needs to perform a function it will interupt your program and perform the function, then let your program continue. His presentation was informative and interesting. If you need to know how to control your house, Watts Hill is the man to ask.

NOVAPPLE NEWS NOTES

This may turn out to be a new column for the Apple Pi Newsletter if we can get enough support from the members of NOVAPPLE. It will be a column for the members to tell everyone about little items they have discovered which make using the APPLE easier. It can also include errors found in programs from magazines and even from Apple Pi. Robert Steele Jr. contributes the first piece of information to this segment. He has made and tried the fix to DOS 3.2.1 suggested by Hersch Pilloff in the April newsletter and finds it works well but will not work with a Language Card. It will apparently work with all other languages. If you have a tidbit to contribute bring it to the meeting and give it to the Secretary or mail it to Gerald Eskelund, 8104 Glover Court, Springfield, Va, 22152. It will be typed up for the next newsletter.

BYTES FROM THE APPLE PI

by Sandy Greenfarb

Yes, Virginia (and DC and Maryland) there is an APPLE III and a DOS 3.3. Bernie Urban will be out West this month to witness the birth of the APPLE III, a 6502 2 MHz machine with built in disk drive. Hopefully he will be back in time to report in this issue. As I hear it from the Applevine, the machine is upward compatible. In layman's terms, that means that all APPLE II programs will run on the APPLE III. sounds similar in concept to the TRS-80 Model II.

DOS 3.3 is not compatible with DOS 3.2 and will cost about $75. Pascal users are already familiar with its 16-sector format (as compared to 13-sector Dos 3.2.) The Master diskette will contain a program called "UPDATE 16". This will update all old diskettes prepared under 3.0 - 3.2.

Trivia Department: Did you know that there are several thousand varieties of apples and that the U.S. is the world's leading apple producer. The northwestern states produce about one-third of the crop in the U.S. Washington is the leading state and New York is second. Pennsylvania, New Jersey, Virginia and West Virginia are also important centers. The encyclopedia neglects to mention California.

Call -A.P.P.L.E will be releasing a new diskette May 15 containing an "Extended", "Relocatable" Integer Basic in RAM and a vastly improved version of TED II, all for $17.50. While primarily for APPLE PLUS owners, regular APPLE II owners may enjoy some of the new features added to this RAM version of Integer Basic. TED II will be usable to all, without question. Moreso, the documentation is supposed to be excellent. Watch the next issue of Call -A.P.P.L.E. for announcement.

Speaking of documentation, I have asked Dave Morganstein, our Librarian, and he has consented to the following proposal: To form a documentation committee on a continuing basis to review the club disk library. Committee size should be arbitrarily three or four, large enough to spread effort and produce cross-views, but not too large as to be redundant. The primary purpose would be to create documentation for library diskettes where needed. This is most definitely needed with the utility diskettes and, in some cases, on the application and game diskettes. The committee would also be responsible for reviewing diskettes in

contd.

their entirety in order to determine where correction, upgrading, and enhancement should be applied. Dave has published requests for information in general, but I would like to set up this standing committee to do the job in a logical and orderly sequence. I promise no other rewards except hard work and the task of doing a worthwhile job. Any volunteers? See Dave or Sandy at the next meeting.

On page 4 of the April newsletter, Andy Rose reconfirmed that there's nothing "green" about our Greenapples. His article on disk organization highlighted the other significant weakness of most libraries (besides documentation) - a need for a categorical filing system. It is rumored that there are over 10,000 programs for the APPLE II. I have still to see an effective catalog and cross-reference system that will handle the smallest portion of these. Somewhere in our club is a person with the idea for the optimum method. Please come forward and be recognized!

A Worm in the Greenapple: In his maze article, Andy said that one could initialize the parameters with a CLR statement. The CLR command only initializes undimensioned variables and one must either read the footnotes to the footnotes to realize this, or learn from bitter experience. (In actuality, the initialization is caused by a combination of setting pointers with the first use of a variable, but that is neither here not there. The effect is the same.) There's nothing green about this mistake, Andy. It puts you in company with the best of programmers. To cut the maze initialization time by a factor of ten, remove lines 20-23 and insert the following:

```
20 MAZE=0:POKE 0,PEEK(208):POKE 1,PEEK
   (209):K=PEEK(0)+256*PEEK(1)+7:L=
   PEEK(K-2)+256*PEEK(K-1)-2
21 POKE 60,K MOD 256:POKE 61,K/256:POKE
   62,L MOD 256:POKE 63,L/256
22 POKE 66,(K+1) MOD 256:POKE 67,(K+1)
   /256:CALL -468
23 TEXT:  CALL - 936
24 CALL -1059:GOSUB 1000
```

What this sneaky routine does is move 0 into the first byte of the dimensioned variable area of MAZE, determines the lower and upper limits of the area, then zeros it at machine language speed. Warning: the "7" in line 20 represents the number of characters in the variable name (MAZE)+3.

Trickery such as I suggest is two-sided. While it may serve a beneficial function, it may also make a program more difficult to understand. More so, it may add transparent limitations (such as I mention). As most of us like things our own way and do modify the programs of others, I strongly suggest that special tricks/techniques be well documented so as not to trap the unsuspecting.

Good Guy Department: Add Mike Harvey, the Editor/Publisher of NIBBLE, to the good guys. At our last meeting William Harmon of Springfield was passing around a few copies of a reply from the magazine to a problem with one of their programs. The reply and accompanying program changes were impressive. They appear very cordial, cooperative and responsive. (I could not test them as I am now two weeks past Apple's announced delivery date for their Silentype Thermal Printer and am still waiting. Sound familiar?

(Author's note: I just deleted from this article a thank you to a particular member. My second thoughts said that there are too many that have never received proper accolades. To specifically thank one would slight too many others by omission.)

SIG Department: On May 6, SIGAMES will have conducted its first outside meeting. Hopefully, there may be a report on the SIG elsewhere in this issue. It is less that two months ago that Al Gass, SIGAMES Chairman, asked for others with a common gaming interest to meet with him after the meeting. Now the group appears to be thriving. On the bulletin board of one of the local computer stores are three similar cards from different APPLE owners, all asking to get toghether with others interested in stock/investment applications. All it takes is for one person to initially act as a focal point. Less than one and one-half years ago, Bernie Urban acted as focal point for a fledgling computer group which eventually became Washington Apple Pi. Who will ask to meet with others of a common interest and potentially form the next SIG?!

Word for Today: Synergism-the simultaneous action of separate agencies which, together, have greater total effect than the sum of their indiviual effects. The term is generally applied to medicine, but I prefer to apply it to people as well.

....

SIGNEWS

SIGAMES, Special Interest Group on Games, will hold their regular meeting immediately following the Washington Apple Pi meeting. The location will be announced at the general session. There will be a discussion of Tom Lucas's proposed implementation of an Adventure game and a report on APPLE products found at the N.C.C.

****

TELEPHONE SNAFU

"The 18-Minute Tape Erasure"

Would those persons requesting Membership Applications and not receiving them, please call again.

****

PERSONAL INFORMATION NETWORKS

A USER'S VIEW by Chuck Reinbrecht

The growth in personal computers has spawned a parallel development in low-cost time sharing. During 1979 two companies, the SOURCE and MicroNET, began offering low-cost, off-hours, dial-up computer access. They also are precursors in emphasizing information and services delivered by terminals as a new dimension in time-sharing. An overview of the facilities each provides will serve to both show the new look in time sharing and allow the reader to evaluate the potential benefits for personal use with a micro-computer or terminal.

- ORIENTATION -

The SOURCE calls itself an 'Information Utility', providing a wide variety of databases and services including:

- UPI wire service news on business, sports, columns, etc.
- NY Times Consumer Database and news wire
- UPI financial market prices for stocks and bonds
- Travel club providing information and reservations
- Discount buying service
- National Real Estate Locator Service

Started in late 1978 by Telecomputing Corporation of America, the SOURCE is in McLean, VA. Using Prime computers, it has limited experience in time sharing but is the leader in the concept of providing low-cost access to a broad array of information. Estimated users number about 3000. In April a library service was initiated at Lamar University in Beaumont, TX in conjunction with OCLC. The former name was Ohio College Library Center, located in Columbus, OH.

MicroNET is more oriented towards providing traditional 'dial-up' access to large computer capabilities for a reduced rate at non-commercial time periods. The recent offering of MicroQuote (stock and bond market data) is its first database venture. As the Personal Computing Division of CompuServe Inc., there is much experience in time sharing. Estimated users number about 1200. An interesting sidelight is the H&R Block agreement to buy control of CompuServe. Located in Columbus, OH and using DEC equipment, MicroNET began service in mid 1979.

There is a marked contrast between providing a wide variety of information versus offering computer access at a lower cost. Another contrast exists between 'people-oriented' instructions and data processing type commands. The information approach is moving in the same direction as the cable TV systems that permit user interaction such as the system in Columbus, OH and Viewdata in the UK. In the Viewdata system, information is selected by a 10-key pad and is received on a home TV.

- ACCESS -

The SOURCE provides 300 baud (or 30 char/sec) ASCII dial-in access for $2.75 an hour from 6 PM to 7AM weekdays and all day weekends and holidays. There is an $100 application fee; a monthly minimum of $5; and a charge of $1 per month for each 2K of on-line file storage. During prime time access is $15/hr. Available languages include BASIC, Fortran, COBOL, Pascal and RPG. Telenet and Tymnet are used to provide local telephone service in most large cities.

MicroNET provides ASCII access at speeds ranging from 10 cps to 120 cps and EBCDIC access at 15 cps. The application fee is $9 and connect charges are $5/hr from 6PM to 5AM weekedays. On-line file storage of 128K is provided free. Languages include BASIC, Fortran, Pascal, Snobol, Focal and APL. Local telephone access is provided in major cities with Tymnet access in other locations at an additional $2/hr. Use of MicroQuote is charged on the basis of amount of data accessed plus connect time charges. This can get rather expensive.

Both vendors bill directly to either MasterCharge or Visa and provide an 800 telephone number for questions and user assistance.

Compare these charges with the $15-25 per hour for normal time-sharing services. Monthly minimums of $100-250 further discourage the small occasional user. Major emphasis is on data processing applications, although there is an increase in the availability of databases particularly for economic and demographic analyses.

- APPLICATIONS -

For the personal/home computer user there are game programs and even the ability to chart biorhythm cycles! The problem-solver will find more standard mathematical, statistical and financial calculation programs. For elementary and high school students there are programs that provide instruction in arithmetic, algebra and foreign language drill. Also available are sorts, cross-system compilers, file creation and edit, and high speed printing at the host computer site (with results mailed to the user).

Programs to transfer files between a personal computer and the host are available for both systems. Peripherals Unlimited has a $25 SOURCE Terminal disk package and MicroNET offers 'free' disk programs for a service charge of $10. These programs allow a user to obtain data from the system and to then use a micro-computer for analysis, formatting and printing off-line at the user's leisure. This is encouraged even to the extent that the SOURCE provides a special command to put any output into a file for downloading.

- ELECTRONIC MAIL -

One of the major requirements for the office of the future is to provide electronic mail.. The SOURCE and MicroNET offer their users access to this facility

contd.

today! No geographic limitations; no delays in delivery; no need for an 800 number - the uses are limited only by one's creativity. For example the author, in Maryland, is the SOURCE coordinator for A.P.P.L.E., a user group in Seattle, WA with 3000 members worldwide. The two implementations of mail reflect the background of the suppliers and their very divergent approaches to easy-to-use commands and normal letter conventions.

MicroNET gives the user the ability to compose a letter on-line and send it to another user. At sign-on, the recipient is notified that there is a message waiting. The 'mail' is handled through the Bulletin Board system which allows users to 'post' for sale, wanted and general notices. There is also a feedback procedure (free connect time) for sending questions and comments to MicroNET. Response times are good to fair.

Mail on the SOURCE adds provisions for cc, bcc and forward (with comments) to another user. Delivery options include:

acknowledgment requested (done automatically when read)
reply requested (waits for reply after being read)
express mail (immediate display of the mail at sign-on)

File interaction is an important plus for use in: addressee lists, common paragraph insertions, pre-composing messages and filing copies of messages. Mail may be sent for delivery on a specific future date. Great for birthdays and special occasions and for reminders to the user via mail to himself. Unfortunately, system response time ranges from fair to poor which discourages use of an excellent, easy-to-use information oriented system. Promised improvements are beginning to appear and should reduce the frustrations.

'Mail' may not appear a likely reason for using a network. But as one uses it more value becomes apparent and new uses occur. Computerland stores use it for ordering and status reports plus delivery queries. INTERNATIONAL APPLE CORE uses it for communication with local user's groups. The author has found mail very helpful for getting questions answered from software vendors, equipment manufacturers and other users with similar interests. The unique nature of 'mail' tends to obtain a faster response than a letter!

A similar capability that allows one user to 'talk' to another via terminals is called CB by MicroNET and termed CHAT by the SOURCE. CB allows multiple users to have an electronic conference.

Electronic mail as a specific service is being offered to business customers by Tymnet and Telenet. Obviously, the business use would be to decrease response time for normal business mail. SOURCE and MicroNET users have that ability plus access to a user's directory to search for other users with similar interests.

- USER DIRECTORIES -

A user directory combined with the mail facility can generate a very useful information gathering combination. The effectiveness depends greatly upon the content of the directory. In this respect there are major differences between the two systems.

MicroNET creates a directory with user ID, name, telephone number and type of equipment used for access. The user can decide whether or not to be included. The directory may be listed or searched by any of the entries. Again, response time is good and the directory is always current. User instructions are data processing oriented rather than oriented toward the neophyte user.

The SOURCE directory includes ID, first name, state, and as may areas of interest that will fit on an 80-character line. The entry is created, changed or deleted by the user at his option. The directory may be searched on any item and also on a combination of state and interest. Using this search it was easy to find four users interested in APL in a rather large 'haystack'. The problem is that response time is poor but more disconcerting is the tendency of directory entries to disappear rather frequently. Again, with promised improvements these situations should change!

- SUMMARY -

The choice is not easy. First, does access to a personal information network provide an added dimension to your capabilities? The answer is easy if your needs are for large computer power, language variety, specific mail uses, games, and market prices for stocks and bonds. That solution is MicroNET with an economical application fee, large file storage, economical use charges and proven performance.

On the other hand if your interest is information, mail and services then your choice is the SOURCE. However, that choice may be hard to implement due to the $100 application fee and questions as to system dependability. Very reasonable usage costs, a wide range of information, very good mail facility and an agressive pursuit of the Information Utility concept may help your decision.

The choice is yours. My information came from promotional and available user guides of the two vendors. Evaluations and comments are based on personal use of both systems, and are obviously biased (as are all evaluations). The bias is two-fold: first, as a long time user of the SOURCE and second, an intense interest in the concept of access to a very broad range of information and services. Any comments, corrections, disagreements or suggestions will be very helpful and should be 'mailed' to the author at:

70320,152 on MicroNET (or)
TCA257 on the SOURCE.

###

APPLE owners do it with bytes

SOME INTEGER BASIC/MACHINE LANGUAGE LINKAGES

by Sandy Greenfarb

One of the problems with interfacing Integer Basic and machine language is the absolute locations of variables and line numbers. Often Integer Basic programs are written with the artificialities, "These variables must be defined first" or "HIMEM must equal 8192 and LOMEM must equal 3072", or similar. This is done in order for a supporting machine language to know where to find things. This type of program works fine until someone, having forgotten all the delicate inter-relationships, performs the inevitable minor change that makes everything go awry. Since Integer Basic obviously locates lines and variables, is it possible to use the interpreter itself to find these items? Obviously if the answer were no, this article would not have been written.

For working with variables, pointers 208 and 209 ($D0 & $D1) are invaluable. They always reference the address in the variable table of the first letter of the name of the last/most recent variable reference. Here is one example:

```
10 CHR$=CHR$:  REM REFERENCE IT IN ORDER TO SET POINTERS
20 POKE 0,PEEK(208):POKE 1,PEEK(209):  REM MOVE TO SAFE AREA BEFORE REFERENCING A
   NEW VARIABLE
30 CHR=PEEK(0)+256*PEEK(1)+7:REM DATA FOR CHR$ STARTS AT LOCATION OF START
   OF NAME+ LENGTH OF NAME (TO INCLUDE $ FOR LITERALS) + 3
40 INPUT N:  POKE CHR,N:  PRINT CHR$:  REM YOU HAVE JUST SIMULATED THE APPLESOFT
   CHR$ FUNCTION
```

This facility also works in direct mode (from the console). After a program has been run, the keyboard command Variable Name=Variable Name, followed by PRINT PEEK(208)+256*PEEK(209) will provide the location of the variable per previous rules.

The following subroutine takes this referencing one step further by actually checking the exact length of the variable name and computing the location for the start of data:

```
1000 REM DIRECTLY BEFORE THE GOSUB TO THIS ROUTINE, EXECUTE THE COMMAND VAR=VAR,
     WHERE VAR IS THE NAME OF THE VARIABLE YOU WISH TO LOCATE
1010 POKE 0,PEEK(208):  POKE 1,PEEK(209):  REM MOVE TO SAFE AREA
1020 FOR LOC=PEEK(0)+256*PEEK(1)+1 TO PEEK (0)+256*PEEK(1)+30:  REM ASSUME MAX
     VARIABLE NAME LENGTH OF 30
1030 IF PEEK(LOC)<64 THEN 1050:  REM FIRST BYTE LESS THAN $40 DENOTES VARIABLE
     NAME HAS ENDED
1040 NEXT LOC:  PRINT "ERROR HALT - NAME TOO LONG":  END
1050 LOC=LOC+3:  REM ADD FOR DSP BYTE AND 2-BYTE ADDRESS OF NEXT VARIABLE
1060 RETURN:  REM NOW POINTED AT FIRST BYTE OF DATA
```

It is actually quite simple to find the absolute location of an Integer Basic line in direct mode; simply issue the direct command "LIST X", where X is the desired line number minus one. PEEK(226)+256*PEEK(227) now points to the next higher line that exists. If no higher line numbers exist, the pointers point to HIMEM. This procedure does not work for pointing to the first line if it is numbered zero, but PEEK(202)+256*PEEK(203) always points to the first line of a program. Warning: do not try to print the value direct on machines larger than 32K.

The following Integer Basic and accompanying machine language routines demonstrate finding lines, printing integer numbers in the range 0 - 65535, and printing hex. The machine language portion is completely relocatable and can easily be located in another part of memory. As written, it can be loaded with the Mini-assembler.

```
 10 INPUT "LINE DESIRED",N
 20 POKE 0,N MOD 256:  POKE 1,N/256:  REM PASS PARAMETERS
 30 CALL 768:  REM FIND LINE, RETURN VALUE IN 0/1
 40 LOC=LOC:  REM NEED A KNOWN VARIABLE LOCATION
 50 POKE 2,PEEK(208):  POKE 3,PEEK(209)
 60 X=PEEK(2)+256*PEEK(3)+6:  REM LOCATION OF DATA IN LOC
 70 POKE X,PEEK(0):  POKE X+1,PEEK(1):  REM POKE IN THE DATA RATHER THAN COMPUTE
    IT IN CASE GREATER THAN 32767
 80 GOSUB 200:  REM THE FIRST BYTE OF A LINE IS THE LENGTH BYTE.  THE NEXT TWO
    BYTES ARE THE LINE NUMBER.
 85 REM THE SUBROUTINE IN 200 WILL PERFORM AN ABSOLUTE INCREMENT OF THE VALUE IN
    LOC (USING THE POINTER X) AND CAN BE GREATER THAN 32767.
 90 REM IT JUST KEEPS THINGS SAFE FROM MACHINE SIZE ERROR.
100 LN=PEEK(LOC):  REM LOWER HALF OF LINE NUMBER
110 GOSUB 200:  REM INCREMENT AGAIN
120 LN=LN+256*PEEK(LOC):REM LN NOW EQUALS LINE NUMBER FOUND
130 IF PEEK(0)=PEEK(76) AND PEEK(1)=PEEK(77) THEN 190:  REM HIMEM, THEN NO LINE FOUND
140 IF LN#N THEN 190:  REM ANOTHER NUMBER FOUND, THEN LINE REQUESTED DOES NOT EXIST
```


contd.

```
150 PRINT "LINE # ";N;" LOCATED AT DEC-";:  CALL 788
160 PRINT " HEX-";:CALL 799
170 PRINT: END
190 PRINT "LINE # ";N;" DOES NOT EXIST":END
200 REM INCREMENT X ROUTINE - WILL NOT CAUSE GREATER THAN 32767 ERROR
210 POKE X,(PEEK(X)+1)MOD 256
220 IF NOT PEEK(X) THEN POKE X+1,(PEEK (X+1)+1)MOD 256
230 RETURN
```

Though the following is written in an assembler format, no labels are used. It can easily be loaded with the Mini-assembler starting at $300.

```
ORG $300
*       This first routine takes the line number found in Locations 0 and 1,
*       accesses a routine in the Basic Interpreter, and returns the
*       absolute location of the line in 0 and 1. If line number requested
*       does not exist, it returns next higher line location or HIMEM,
*       as appropriate.
LDA     $00 Low byte of line number
STA     $CE Where Int Basic routine expects to find it.
LDA     $01 Same for high byte of line number
STA     $CF
JSR     $E56D Location of Int Basic routine
LDA     $E4 Where Int Basic put low byte of result
STA     $00 Where we want it
LDA     $E5 Same for high byte
STA     $01
RTS     Exit to Int Basic Program.
*       Next sequential location is $314 (Dec 788). This routine takes the
*       16-bit number in 0/1 and passes it to an Int Basic Interpreter
*       routine which prints a number in range 0 - 65535
LDA     #$01 Must have a positive number
STA     $F8 Must set this byte positive
LDX     $00 Interpreter routine expects low byte in X-REG
LDA     $01 And high byte in ACC
JMP     $E51B Prints number and then RTS in interpreter routine jumps
*       right back to main program. Next sequential location is $31F (Dec 799)
*       This routine is a monitor routine which prints 2-byte hex value.
LDA     $00 Routine expects low byte in ACC
LDX     $01 And high byte in X-REG
JMP     $F941 Prints value and exits to main program.
```

Here are some other potential linkage tools:

220 & 221 ($DC & DD) point to first byte of current Integer Basic line in use.

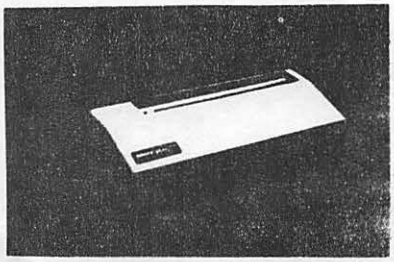
224 & 225 ($E0 & E1) point to byte after last byte executed. For example, in Line 30 of the preceding program, while in machine language (from the CALL 768), 224 & 225 would be pointing at the colon. One might reasonably be able to use this set of pointers in a calling sequence of parameters to pass to machine language.

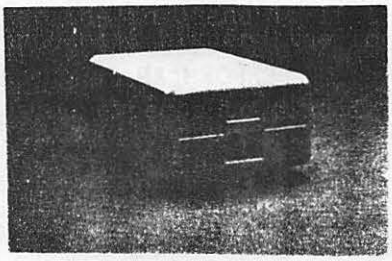
To execute a GOTO direct from machine language:

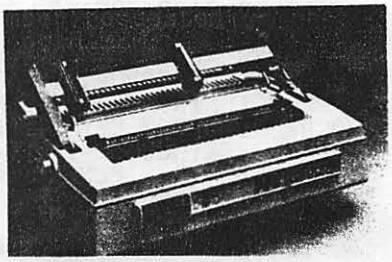
Find location of absolute location of Integer Basic line (potentially by using similar routine to one described in previous machine language example at $300). Then use this routine.

```
LDA XLO   Low byte of absolute address.
STA $C6   Where interpreter expects it.
LDA XHI   Same for high byte of absolute address.
STA $C7
JMP $E867 This interpreter routine performs the GOTO.
```

In the first issue of "APPLE ORCHARD" is an outstanding article by Randy Hyde on how to convert Integer Basic to machine language. Depending on the size of the program, this could be a monumental task. Hopefully this article supplements his and makes it possible for a programmer to take bytes out of his Integer Basic program and perform the conversion bit by bit. (Puns fully intended).







INTERRUPT YOUR APPLE

(for the time)

by Bruce F. Field

Inside your APPLE is a 6502 microprocessor chip that is constantly executing a program in it's memory. If there is no user program executing, then at least the Monitor program is interrogating the keyboard waiting for you to press a key. However while the processor is running a user program you can press keys on the keyboard all you want (except RESET) and unless the program specifically reads the keyboard, nothing happens. This is an example of polled I/O, the processor (as directed by the program) must periodically poll the keyboard to see if a key has been pressed.

Another way to get the attention of the processor is to interrupt it. An interrupt is a way for an external device to signal the processor that something special must be done. The external device pulls the interrupt input of the processor to a logic low and the processor stops what it is doing and starts executing another special programn When the special program, generally called an "interrupt service routine", is finished the processor goes back and continues executing the original program exactly where it left off. The way the APPLE is configured requires that the interrupt service routine be written in machine code, but the original program that is interrupted may be machine code or either flavor of BASIC.

The microprocessor may be interrupted in three different ways. The first two ways are by electrically pulling one of the two inputs on the microprocessor chip to a logic low level. These two inputs are connected to the APPLE expansion bus and are available as IRQ (Interrupt ReQuest) pin 30, and NMI (Non-Maskable Interrupt) pin 29. The inputs are normally held high by 3K resistors so that no interrupts occur if nothing is connected to them. The two inputs are similar except the IRQ input may be disabled by executing a SEI (SEt Interrupt disable) instruction in machine code, whereas the NMI input is permanently enabled. More on this later. The third method of interrupting the processor is by executing a BRK instruction in machine code. This is used most frequently for debugging machine language programs with the Monitor trace command.

When the APPLE is interrupted, a series of instructions in the Monitor are executed before control is turned over to the user. When the interrupt occurs (the IRQ line going low) the microprocessor finishes the machine instruction it is currently executing, internally disables the IRQ line to prevent further interrupts, saves the status register on the stack, and branches to $FF86 in the Monitor. The accumulator is then saved at memory location $45 and a test is performed to see if the interrupt resulted from executing a BRK instruction. If it was not due to a BRK then control branches to the address that is stored in memory at $3FE and $3FF. If you are going to use interrupts you must store the starting address of your interrupt service routine in $3FE and $3FF (low byte in $3FE, high byte in $3FF). The APPLE will then execute your machine language routine. In order to return to the interrupted program and have it execute properly, it is your responsibility to make sure that all the 6502 registers are restored to the values they contained when the interrupt service routine was started. The two exceptions are the accumulator, which was stored by the Monitor in $45 but which must be restored by you, and the status register which is taken care of automatically by the 6502. To return to the original program, restore the registers and execute a RTI (ReTurn from Interrupt) instruction. The RTI instruction will automatically enable the IRQ interrupt so the processor will respond to future interrupts.

When would we want to use an interrupt? If an external device, such as an alarm input, requires servicing at infrequent intervals we could write as part of every program a subroutine that periodically tests the alarm input to see if it is triggered, and then ring a bell. This procedure wastes a lot of time if the alarm is rarely triggered. A better way is to have the alarm pull the interrupt line low to let the processor know it needs attention, and then have an interrupt routine ring the bell.

Since there are only two interrupt lines to the processor, what happens when we have more than two devices that need to interrupt the processor? The IRQ (or NMI) line is connected to all the devices and when the line is pulled low we will have to poll each device to see which one needs attention. In this case it is up to the hardware designer to design the device with a status register that can be read as memory to indicate that the device needs service. The interrupt service routine must then look at the status registers of each possible interrupting device until it finds the one requiring service.

APPLE has defined a certain convention that they recommend when using the bus expansion connectors with interrupts. Each connector has an INT IN and INT OUT line to signal other devices as to whether they may interrupt the processor or not. The INT IN line at each slot is connected to the INT OUT line at the next lowest numbered slot. The definition says that peripherals in lower numbered slots have priority over higher numbered slots. What this means is that if a device in slot 4 is interrupting the processor it sets the INT OUT line high as a signal to devices in higher numbered slots (5-7) that they should not interrupt at this time. Once the interrupt service routine for slot 4 is completed the INT OUT line is returned low and lower priority devices may now interrupt. If you are designing a peripheral to operate with multiple interrupts it is your responsibility to make sure your device conforms to the convention so that it will work with other peripherals designed for the APPLE.

There are some times when it is undesirable to permit interrupts. If you expect a program to run in a certain length of time, servicing an interrupt will disrupt the timing. On the APPLE, software timing is used for the Serial Interface Card, tape I/O, and disk I/O. An interrupt occurring during a disk write could cause data to to be written randomly on the diskette destroying good data in the process. To prevent this, these routines set the interrupt disable bit in the processor status register (with a SEI) before any critical timing code, and clear the interrupt disable bit (with a CLI) when it is again safe to permit interrupts. (DOS 3.2 disables the interrupts, but I have heard that DOS 3.1 does not; proceed carefully if you use 3.1.) This interrupt disable/enable provision must also be included in any printer routines using the game I/O port.

As mentioned earlier, the interrupt disable bit only controls the IRQ line, the NMI line is permanently enabled. Because of the possibility of messing up I/O transfers I recommend you not use this line. If however you want to use it to signal something important, like the house if on fire, and you aren't going to worry about a clobbered diskette, go ahead and use it.

contd.

EXPERIMENTER'S REAL TIME CLOCK

One of the most common uses for an interrupt (at least on larger computers) is to interrupt the processor at regular intervals in time and using software (the interrupt service routine) increment a memory location to keep track of the time. We can build a very simple circuit that generates an interrupt once a second to have the APPLE keep time with the operation reasonably transparent to other user programs. You can be running your favorite game and when you're finished the APPLE will still have the correct time.

Implementing this concept on the APPLE creates a problem when I/O transfers occur. We do not want to interrupt during these transfers but on the other hand if we don't we will miss timing pulses and no longer have the correct time. What needs to be done is to design a circuit that interrupts the APPLE but also counts any pulse the APPLE misses because it is busy with I/O. Then when it is not busy we can load this count in to update the clock. And, we want to do all this with a minimum amount of hardware.

There are many different ways to design such a circuit, the way I have chosen is shown in the figure. I do not necessarily feel that this is an optimum design. My intention was to design a circuit to perform the clock function with a minimum of readily available TTL integrated circuits. There is a tradeoff between how often the interrupts should occur, which determines the clock resolution, and the ability to count a large number of pulses, which requires additional hardware. This circuit interrupts at a one second rate and will properly count pulses that are missed when the APPLE is busy for up to 4 1/2 minutes. If the interrupts are continuously disabled for more than 4 1/2 minutes (an unlikely occurrence) the time will be incorrect.

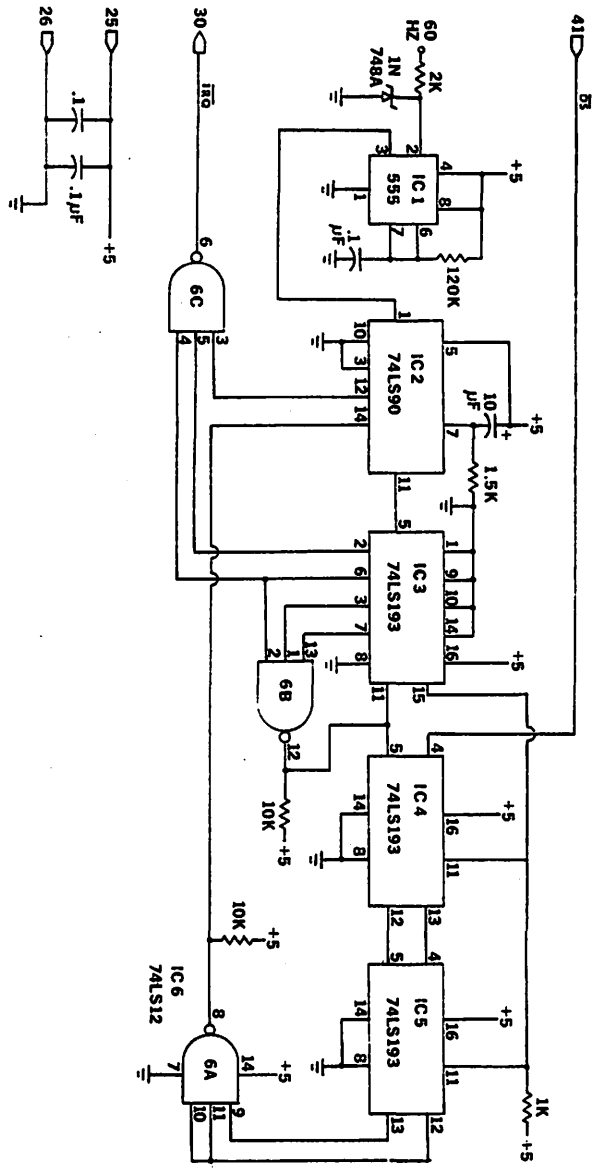
The circuit works as follows:

Low voltage 60 Hertz ac derived from the power line is shaped by IC1 to produce TTL level pulses at 60 Hz. IC2 and 3 are counters that divide the frequency by 60 to produce one pulse per second. This pulse is sent to cascaded counters IC4 and 5 which cause them to increment by one and simultaneously send an interrupt pulse from the carry output to the processor. If the processor responds to the interrupt (by executing the previously stored interrupt routine) the interrupt routine will pulse the device select line (DS, pin 41) on the expansion bus and increment the software time counter in memory. When the device select line is pulsed, IC 4 and 5 are decremented by one thus restoring them to their original count and removing the interrupt condition. If the processor is busy and does not pulse the device select line, the present count is stored and incremented by the next one second pulse. Since the counters have been incremented above the original count, interrupts to the processor are generated even if the processor doesn't respond. When the processor re-enables the interrupt input internally it will be interrupted and will pulse the device select line to decrement the counters. If the counters are not restored to their original count, another interrupt is generated immediately. This process continues until all accumulated counts have been removed.

The 60 Hertz should be at a low voltage (3-12 volts). I have been using a small wall type transformer that provides about 9 volts. Similar transformers are used for calculators, although make sure it produces ac or half-wave rectified dc. Full-wave rectified or filtered dc will not work. The actual voltage is unimportant as long as it is greater than about 3 volts. The zener diode (1N748A) limits the voltage to protect IC1; and power to the circuit can be turned off with the 60 Hz still on and it will not damage IC1. The circuit can be constructed any number of ways, but I suggest that you build it using point-to-point or wire-wrap wiring on an Apple prototyping board. The pin numbers on the circuit diagram are for the expansion bus connectors.

Of course since this circuit depends on the APPLE processing interrupts, and derives it's power from the APPLE, it will not keep time with the APPLE turned off. Also, other than disconnecting the 60 Hz input, there is no way to stop this circuit from generating interrupts. But, if you are a do-it-yourselfer and want to learn more about how the APPLE works this is a simple one or two evening project.

NEXT MONTH (maybe) - I'll describe the software to allow continuous display of the time in hours, minutes, and seconds in the upper right hand corner of the screen.



BLAISE AWAY!

THE PASCAL TO PAPER TIGER CONNECTION

BY
DR. WO

THIS IS MY FIRST EFFORT TO MAKE GOOD ON A PROMISE TO WRITE A MORE OR LESS MONTHLY COLUMN ON THE APPLE LANGUAGE SYSTEM AND ESPECIALLY THE APPLE IMPLEMENTATION OF U.C.S.D. PASCAL. MOST OF THE COLUMNS I HAVE IN MIND ARE APPLICATIONS SOFTWARE ORIENTED, WHICH IS CONSISTENT WITH MY INTEREST IN APPLICATIONS AND SYSTEMS PROGRAMMING. THIS COLUMN FEATURES A PROGRAM TO DUMP THE HIRES SCREEN TO THE IDS-440 PAPER TIGER. SUBSEQUENT ARTICLES WILL DEVELOP A SAMLL LIBRARY OF HIRES UTILITY ROUTINES FOR SAVING AND RECALLING HIRES IMAGES TO AND FROM DISK. IN ADDITION I AM DEVELOPING ROUTINES TO OPERATE MY D.C. HAYES MICROMODEM UNDER PASCAL; SO FAR I HAVE WRITTEN AN AUTO-DIAL DUMB TERMINAL AND I HOPE TO ADD SOME ROUTINES FOR MASS DATA TRANSFER. OFF IN THE DISTANT FUTURE WE COULD DISCUSS STATISTICAL APPLICATIONS.

THESE ARE EXAMPLES OF WHAT I AM INTERESTED IN. BUT WHAT ARE YOU INTERESTED IN? BOOK REVIEWS? PASCAL TUTORIALS? A TOUR OF THE B.I.O.S? LET ME KNOW AND I'LL TRY AND COOK SOMETHING UP. I'D BE HAPPY TO TRY AND ANSWER YOUR QUESTIONS ABOUT THE LANGUAGE SYSTEM OR PASCAL.

NOW FOR THE HIRES DUMP. IN GRAPHICS MODE THE PAPER TIGER USES A RASTER SCAN TECHNIQUE WHICH ENABLES EFFECTIVE PRINTING OF VERTICAL COLUMNS OF UP TO SIX DOTS DURING EACH PASS OF THE PRINT HEAD, THE DOT POSITIONS IN ANY COLUMN REPRESENTING THE BINARY BIT CONFIGURATION OF ONE RECEIVED DATA CHARACTER. THUS IT IS CONVENIENT TO THINK OF THE HIRES SCREEN AS CONSISTING OF STRIPS SIX DOTS HIGH AND 280 DOTS WIDE. CONVENIENTLY, THIS MEANS THE 192 DOTS HIGH APPLE SCREEN BREAKS UP INTO 32 STRIPS. THUS OUR TOP DOWN SOLUTION TO DUMPING THE SCREEN IS:

```
TURNONGRAPHICS;

FOR STRIPNUMBER:=0 TO 31 DO
 BEGIN
  GETSTRIP(STRIPNUMBER);
  WRITE(LP,STRIP);
  WRITE(LP,GRAFICS,VERTAB);
 END;

TURNOFFGRAPHICS;
```

TURNONGRAPHICS AND ITS COMPANION, TURNOFFGRAPHICS, SHOULD BE SELF-DESCRIPTIVE. THE WRITE STATEMENTS MERELY TRANSFER STRIP, A PACKED ARRAY OF CHARACTERS, TO THE PRINTER. THAT LEAVES ONLY GETSTRIP, THE HEART OF THE PROGRAM, TO BE DEFINED.

IF WE LOOK AT THE HIRES SCREEN A LITTLE CLOSER WE SEE THAT IT IS LAID OUT IN ROWS OF HORIZONTAL BYTES, A LITTLE INCONVENIENT SINCE THE PRINTER WANTS TO LOOK AT THINGS VERTICALLY. HOWEVER, IF WE BREAK UP OUR STRIPS INTO BLOCKS OF CONVENIENT SIZE WE CAN CHANGE THE PERSPECTIVE FROM HORIZONTAL TO VERTICAL. THIS BRINGS US TO THE (EXTERNALLY DEFINED) MACHINE LANGUAGE ROUTINE FILLBLOCK, WHICH IS THE SOUL OF THE PROGRAM. THUS OUR SKETCH OF GETSTRIP IS TO CALCULATE THE ADDRESS OF THE FIRST BLOCK OF THE CURRENT STRIP AND ALTERNATIVELY FILL THE BLOCK, PASS THE BLOCK TO STRIP AND UPDATE THE POINTERS TO POINT AT THE NEXT BLOCK. THUS:

```
FOR BLOCKNUMBER:=0 TO 39 DO
 BEGIN
  FILLBLOCK(POINTERS);
  MOVERIGHT(.....
  FOR BYTENUMBER:= 0 TO 5 DO.....
 END;
```

MOVERIGHT IS A SYSTEM INTRINSIC PROCEDURE FOR FILLING STRINGS AND PACKED ARRAYS OF CHARACTERS.

AS THE PROGRAM IS NOW WRITTEN IT REQUIRES YOU TO DRAW YOUR FAVORITE HIRES PICTURE WITHIN THE PROGRAM, WHICH CONTRADICTS THE MODULAR APPROACH TO PROGRAM DEVELOPMENT. SORRY 'BOUT THAT. IN A FUTURE PIECE I'LL INCORPORATE THE DUMP ROUTINE INTO A LIBRARY UNIT, WHICH WILL TAKE CARE OF THE PROBLEM. UNTIL THEN.....

BLAISE AWAY!!

```
 1   1    1:0    1 (*$LPRINTER:*)
 2   1    1:0    1 PROGRAM PRINTSCREEN;
 3   1    1:0    3
 4   1    1:0    3 TYPE ADDRS=ARRAY[0..5] OF INTEGER;
 5   1    1:0    3
 6   1    1:0    3 VAR BLOCK:PACKED ARRAY[0..6] OF CHAR;
 7   1    1:0    7     STRIP:PACKED ARRAY[0..279] OF CHAR;
 8   1    1:0  147     STRIPNUMBER:INTEGER;
 9   1    1:0  148
10   1    1:0  148     LP:TEXT;
11   1    1:0  449     NORMAL,ENHANCED,CPI8,CPI12,GRAFICS,
12   1    1:0  449     VERTAB,LINEFEED,FORMFEED,RETURN:CHAR;
13   1    1:0  458
14   1    2:0    1 PROCEDURE FILLBLOCK(POINTERS:ADDRS);
15   1    2:0    8 EXTERNAL;
16   1    2:0    8
17   1    3:0    1 PROCEDURE TURNONGRAPHICS;
18   1    3:0    0 BEGIN
19   1    3:1    0  NORMAL:=CHR(2);
20   1    3:1    4  ENHANCED:=CHR(1);
21   1    3:1    8  GRAFICS:=CHR(3);
22   1    3:1   12  CPI8:=CHR(28);
23   1    3:1   16  CPI12:=CHR(30);
24   1    3:1   20  VERTAB:=CHR(11);
25   1    3:1   24  RETURN:=CHR(13);
26   1    3:1   28  LINEFEED:=CHR(10);
27   1    3:1   32  FORMFEED:=CHR(12);
28   1    3:1   36
29   1    3:1   36  REWRITE(LP,'PRINTER:');
30   1    3:1   57  UNITCLEAR(6);
31   1    3:1   60
32   1    3:1   60  WRITELN(LP);
33   1    3:1   68  WRITE(LP,GRAFICS);
34   1    3:1   80  WRITE(LP,GRAFICS,CPI8);
35   1    3:0  104 END;
36   1    3:0  116
37   1    4:0    1 PROCEDURE GETSTRIP(STRIPNUMBER:INTEGER);
38   1    4:0    2 VAR SCREENROW,BYTENUMBER,BLOCKNUMBER:INTEGER;
39   1    4:0    5     POINTERS:ADDRS;
```

contd.

```
40   1    4:0     0 BEGIN
41   1    4:0     0
42   1    4:1     0  FOR SCREENROW:=(6*STRIPNUMBER) TO (6*STRIPNUMBER+5) DO
43   1    4:2    17   POINTERS[SCREENROW MOD 6]:=
44   1    4:2    27     8192+40*(SCREENROW DIV 64)+128*((SCREENROW MOD 64) DIV 8)
45   1    4:2    44      +1024*(SCREENROW MOD 8);
46   1    4:2    62
47   1    4:1    62  FOR BLOCKNUMBER:=0 TO 39 DO
48   1    4:2    73   BEGIN
49   1    4:3    73    FILLBLOCK(POINTERS);
50   1    4:3    77    MOVERIGHT(BLOCK[0],STRIP[7*BLOCKNUMBER],7);
51   1    4:3    96    FOR BYTENUMBER:=0 TO 5 DO
52   1    4:4   107     POINTERS[BYTENUMBER]:=POINTERS[BYTENUMBER]+1;
53   1    4:2   134   END;
54   1    4:0   141 END;
55   1    4:0   160
56   1    5:D     1 PROCEDURE TURNOFFGRAPHICS;
57   1    5:0     0 BEGIN
58   1    5:1     0  WRITE(LP,GRAFICS,NORMAL);
59   1    5:1    24  WRITE(LP,NORMAL,CPI12);
60   1    5:0    48 END;
61   1    5:0    60
62   1    1:0     0 BEGIN
63   1    1:0     0
64   1    1:0     0  (* DRAW YOUR FAVORITE HIRES PICTURE HERE *)
65   1    1:0     0
66   1    1:1     0  TURNONGRAPHICS;
67   1    1:1    17
68   1    1:1    17  FOR STRIPNUMBER:=0 TO 31 DO
69   1    1:2    34   BEGIN
70   1    1:3    34    GETSTRIP(STRIPNUMBER);
71   1    1:3    39    WRITE(LP,STRIP);
72   1    1:3    55    WRITE(LP,GRAFICS,VERTAB);
73   1    1:2    79   END;
74   1    1:2    89
75   1    1:1    89  TURNOFFGRAPHICS;
76   1    1:1    91
77   1    1:0    91 END.
```

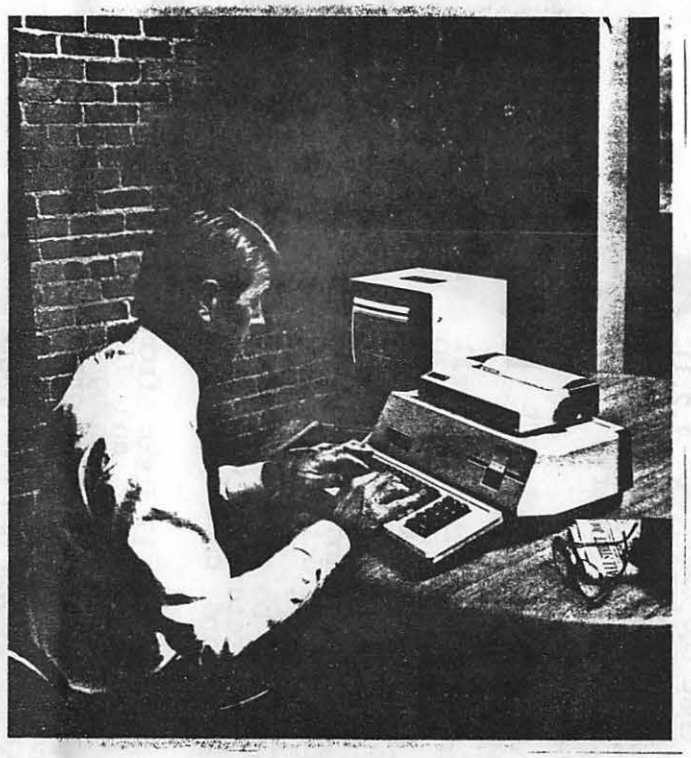
```
PAGE -   0
Current memory available:   10142
0000:                               ;MACRO TO PULL A WORD FROM
0000:                               ;THE STACK AND SAVE IT IN
0000:                               ;A SPECIFIED LOCATION
0000:                               .MACRO POP
0000:                               PLA
0000:                               STA Z1
0000:                               PLA
0000:                               STA Z1+1
0000:                               .ENDM
0000:
2 blocks for procedure code  9363 words left
```

```
PAGE -   1  FILLBLOC  FILE:FILLBLOCK

0000:                               .PROC FILLBLOCK,1; 1 WORD OF
Current memory available:    9585
0000:                                                  ; PARAMETERS
0000:
0000:                               ;BLOCK IS DECLARED IN THE
0000:                               ;GLOBAL SEGMENT OF PRINT-
0000:                               ;SCREEN
0000:                               .PUBLIC BLOCK
0000:
0000: 0000              PASCAL      .EQU 00
0000: 0001              PASCALHI    .EQU 01
0000: 0002              POINTER     .EQU 02
0000: 0003              POINTERHI   .EQU 03
0000: 0004              PNTRSQ      .EQU 04
0000: 0005              PNTRSQHI    .EQU 05
0000:
0000:                               ;INITIALIZE BLOCK & COUNTER
0000: A9 00                         LDA #00
0002: A2 06                         LDX #06
0004: 9D 0000           ZERO        STA BLOCK,X
0007: CA                            DEX
0008: 10FA                          BPL ZERO
000A:
000A:                               ;SAVE PASCAL RETURN ADDRESS
000A:                               POP PASCAL
000A: 68                #           PLA
000B: 85 00             #           STA PASCAL
000D: 68                #           PLA
000E: 85 01             #           STA PASCAL+1
0010:
0010:
0010:                               ;BEGIN MAIN ROUTINE
0010:
0010:                               ;POP THE POINTER TO THE SCREEN
0010:                               ;POINTERS, THEN START THE MAIN
0010:                               ;LOOP BY LOADING
0010:                               ;THE ACCUMULATOR WITH A
0010:                               ;SCREEN POINTER AND SAVING IT
0010:                               ;IN POINTER.
0010:
0010:                               POP PNTRSQ
0010: 68                #           PLA
0011: 85 04             #           STA PNTRSQ
0013: 68                #           PLA
0014: 85 05             #           STA PNTRSQ+1
0016: A0 0B                         LDY #0B
0018: B1 04             NEWBYTE     LDA @PNTRSQ,Y
001A: 85 03                         STA POINTERHI
001C: 88                            DEY
001D: B1 04                         LDA @PNTRSQ,Y
001F: 85 02                         STA POINTER
0021:
0021:
0021:                               ;LOAD THE ACCUMU-
0021:                               ;LATOR WITH THE VALUE CUR-
```

contd.

```
PAGE -   2  FILLBLOC  FILE:FILLBLOCK


0021!                              ;RENTLY POINTED AT, THEN
0021!                              ;STRIP BITS AND
0021!                              ;DISTRIBUTE TO BLOCK BYTES
0021! A2 00                        LDX #00
0023! A1 02                        LDA @POINTER,X
0025! 4A                  STRIPBIT  LSR A
0026! 3E 0000                      ROL BLOCK,X
0029! E8                           INX
002A! E0 07                        CPX #07
002C! D0F7                         BNE STRIPBIT
002E!
002E!                              ;DONE WITH THIS BLOCK?
002E!                              ;IF NOT, ADJUST THE POINTERS
002E!                              ;AND GET THE NEXT SCREEN BYTE
002E! 88                           DEY
002F! 10E7                         BPL NEWBYTE
0031!
0031!                              ;WHEN DONE, CHECK FOR NON-
0031!                              ;PRINTING GRAPHICS CHARACTERS
0031!                              ;#03 CNTRL-C
0031!                              ;#0D CARRIAGE RETURN
0031!                              ;#10 DLE CHARACTER
0031!                              ;#3F QUESTION MARK
0031! A2 06                        LDX #06
0033! BD 0000              TESTCHAR  LDA BLOCK,X
0036! C9 03                        CMP #03
0038! F0**                         BEQ ODDCHANGE
003A! C9 0D                        CMP #0D
003C! F0**                         BEQ ODDCHANGE
003E! C9 3F                        CMP #3F
0040! F0**                         BEQ ODDCHANGE
0042! C9 10                        CMP #10
0044! F0**                         BEQ EVNCHANGE
0046! CA                  MORETESTS DEX
0047! 10EA                        BPL TESTCHAR
0049!
0049!                              ;WHEN THROUGH TESTING,
0049!                              ;RETURN TO PASCAL
0049! A5 01                        LDA PASCALHI
004B! 48                           PHA
004C! A5 00                        LDA PASCAL
004E! 48                           PHA
004F! 60                           RTS
0050!
0040* 00
003C* 00
0038* 00
0050! 29 3E               ODDCHANGE AND #3E
0052! 9D 0000                      STA BLOCK,X
0055! 4C 4600                      JMP MORETESTS
0044* 00
0058! 4A                  EVNCHANGE LSR A
0059! 9D 0000                      STA BLOCK,X
005C! 4C 4600                      JMP MORETESTS
005F!

PAGE -   3  FILLBLOC  FILE:FILLBLOCK


005F!                              .END
```

```
PAGE -   4  FILLBLOC  FILE:FILLBLOCK  SYMBOLTABLE DUMP

AB - Absolute     LB - Label     UD - Undefined     MC - Macro
RF - Ref          DF - Def       PR - Proc          FC - Func
PB - Public       PV - Private   CS - Consts

BLOCK    PB ----!  EVNCHANG LB 0058!  FILLBLOC PR ----!  MORETEST LB 0046!  NEWBYTE  LB 0018!  ODD
HANG LB 0050!  PASCAL   AB 0000
PASCALHI AB 0001!  PNTRS0   AB 0004!  PNTRS0HI AB 0005!  POINTER  AB 0002!  POINTERH AB 0003!  POP
     MC ----!  STRIPBIT LB 0025
TESTCHAR LB 0033!  ZERO     LB 0004!
```



A PAGE FROM THE STACK

LIBRARIAN'S CORNER - Dave Morganstein

A good first subject for this month is "THE BAD COPY" and how to cope with it. I don't know why copies turn out improperly, though I suspect that the super-quick two controller copier (a modified COPY 3.2.1) is at fault. Disk to disk incompatibility is, however, surely part of the problem. But before returning your library disk, please do the following:

1. Boot-up with another disk, put in the library disk, and type CATALOG. It may work fine.
2. If it won't catalog after several tries, (i.e. gives the ominous "I/O ERROR", beep...) try one last thing. Check your disk speed. this can be done using the program on Apple Pi Disk 1 or just use a florescent bulb and the timing ring on the bottom of the drive. If the drive is timed correctly, and you have tried the alternative booting procedure without success, then return the disk. It will be promptly (?) and cheerfully (!) replaced.

NEWS: I recently received 17 disks from Terry Taylor (Denver Apple Pi and IAC Librarian) in trade for the Washington Apple Pi library. A few of these programs are already available on our disks, including a super Dungeon program called AEMON. This program allows you to create your own character, and you're off on a journey!

COMMERCIAL SOFTWARE-Software you might consider:

Wilderness Campaign (Bob Clardy) - A delightful Hi-Res adventure which has a map of the country (playing area) including a desert, swamp, forest and mountains. You must feed and arm your troops, barter for useful articles, fight mighty monsters and cope with Mother Nature. Don't miss this one!

CRAE (Highlands Computer Services) - An interesting Applesoft utility which includes the following:

Append programs
Change all occurrences of any string
Dump in hex and ASCII
Find any string globally
List in compact form
Number lines on input
Quote lines (i.e. move them around)
Renumber lines
Verify off (Change without verifying).

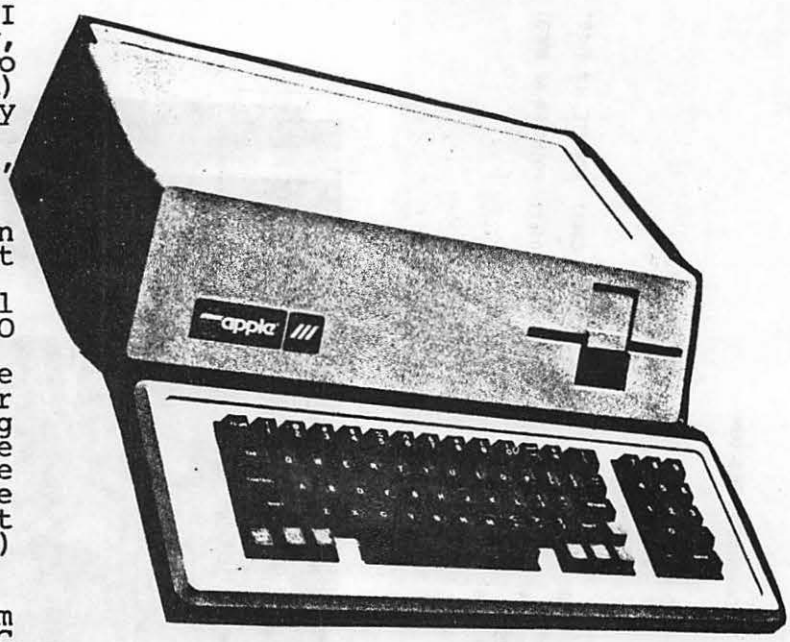
MORE NOTES

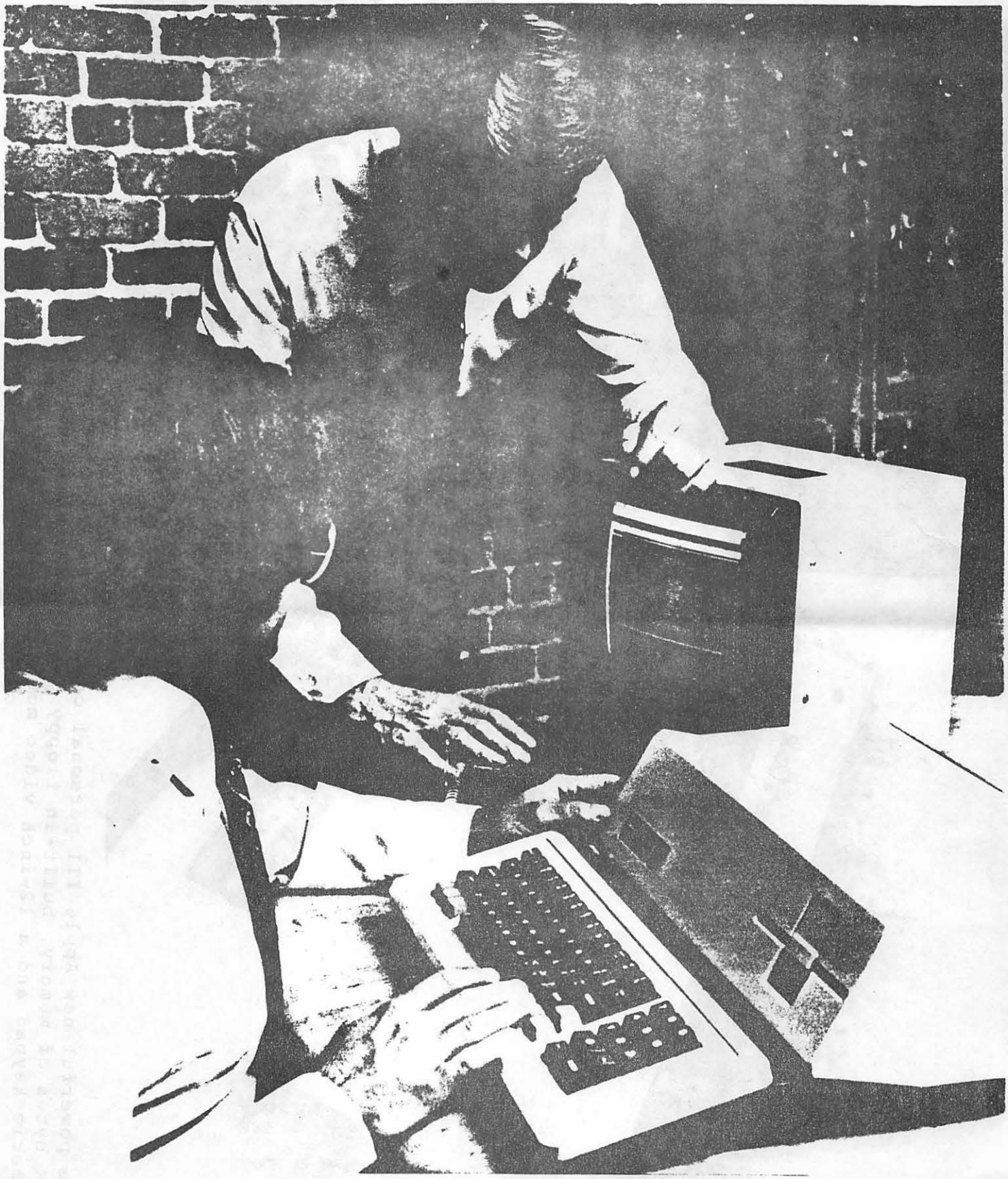
Utility Disk 8 contains the binary program LAZARUS, written by Andy Hertzfeld. This little gem can be used to resurrect a "dead" Integer Basic program fragment. LAZARUS is fully documented in the WOZPAK ii (buy it, you'll like it!) . Just BLOAD to memory and either:

1. Call 768 - for longest available program segment found; or
2. Call 772 - for last segment found.

Then just LIST. Lo and behold, there is a program in memory. To test it, start by >LOADING an Integer Basic program, then NEW it away and try LAZARUS. When you LIST, it'll be back!. Thanks, Andy, for this and many other utilities.

HELP! Please contact me if you want to review one of our 15 disks.





The Apple III

The powerful new Apple III personal computer includes a central processing unit with 96K bytes of memory, built-in floppy disk drive, new integrated keyboard with 13-key numeric keypad and a 12-inch video monitor.

APL-328

WASHINGTON APPLE PI
MAIL ORDER FORM

Washington Apple Pi now has a program library, and disks are available for purchase by anyone. The price to members is $5.00 per disk, and $8.00 to non-members. These disks are chock full of exceptional programs - the utilities are especially useful. The games are some of the best - not just simple and uninteresting ones. You may pick them up at any meeting or have them mailed for $2.00 per disk additional. They will come in a protective foam diskette mailer.

Also available for purchase by members at a discount price is the new APPLE II REFERENCE MANUAL (replaces the Red Reference Manual). The price of this manual is $17.00. You may pick it up at a meeting or have it mailed to you at no extra charge.

	Amount
1. New APPLE II REFERENCE MANUAL - $17.00 each	------
2. PROGRAM DISKETTES	
Members: $5.00 per disk picked up at meeting $7.00 mailed to you...	
Non-members: $8.00 per disk picked up a meeting $10.00 mailed to you...	
Volume 1 Utilities	------
Volume 2 Utilities	------
Volume 3 Games	------
Volume 4 Games	------
Volume 5 Games	------
Volume 6 Games	------
Volume 7 Games	------
Volume 8 Utilities	------
Volume 9 Educational	------
Volume 10 Math/Science	------
Volume 11 Graphics	------
Volume 12 Games	------
Volume 13 Games	------
Volume 14 IAC Utilities	------
Volume 15 Games	------
TOTAL ORDER = $	------

Check here if you want these shipped---

NAME ----------

ADDRESS ----------

CITY, STATE, ZIP ----------

TELEPHONE ----------

Membership No.(1st three digits after WAP on mailing label) ---------

Make checks payable to "Washington Apple Pi"

Send order to: Washington Apple Pi- ATTN: Librarian
PO Box 34511
Washington, DC 20034