WASHINGTON APPLE PI

Vol. 1 No. 4
May, 1979

Dear APPLE Enthusiast,

It's time to exercise your franchise. We have nominated candidates, and you will be voting for officers of your choice at our next meeting. I urge those of you who haven't shown up at our recent sessions to do so at our next one. Please note below the change in schedule made necessary due to the conflict with Memorial Day weekend. I must mention, however, that Mark Crosby, who had been nominated for President and alternatively for Treasurer, has withdrawn his name due to his present state of overcommitment to other activities. Quelle domage! We still expect you to contribute to our newsletter, Mark. To rectify this situation, prior to our first vote I shall re-open the nominations for President and Treasurer. The nominations as they now stand are:

President	John Moon		
Vice-president	Susan Eickmeyer Sandy Greenfarb Bernie Urban		
Secretary	John Ditman Genevie Urban Hal Weinstock		
Treasurer	Robert Peck		
Members-at-Large (3)	Tom Bottegal John Ditman Susan Eickmeyer Sandy Greenfarb	David Itkin John Moon Robert Peck Mort Simon	Bernie Urban Genevie Urban Hal Weinstock Thomas Robinson

Mark Crosby

Short biographies, where available, are attached at the end of the newsletter. I propose that we add one additional member-at-large to be chosen from the Green-apples, and nominate Rick Hodder. Do I hear a second? Unless someone objects, I plan to go through the election to the first four offices, announce the results and then hold the election of the members-at-large.

Church Bells in the Offing?

NOVAPPLE - Northern Virginia's APPLE users group members have agreed to a "trial marriage", at least to the extent of contributing articles, effort and monetary support to get out the newsletter. Phil Eastman, their VP, is our contact. We shall be identifying their inputs by including NOVAPPLE designations with each

article. We shall also be mailing the newsletter to all individuals on their mailing list. Welcome, folks!

Minutes of the 4/28/79 Meeting

Copies of the Constitution and By-Laws and the membership list were distributed. Corrections and additions to the membership list were passed on to Genevie Urban. The members encouraged the process of exchanging newsletters with other groups. Incorporation news dealt primarily with the agreement reached with NOVAPPLE (see above). Nominations for officers were made (see slate on page 1). Considerable discussion took place concerning the moral and legal questions and issues raised by the free exchange of programs marketed by others. A motion was made to appoint a committee to research the entire issue and report back to the membership within 60 days. An amendment was added that we adopt, in the interim, comments on this subject that appeared in the last newsletter. Both the motion and amendment were carried, and Mort Simon and Fred Artiss have been appointed to this committee. Tom Bottegal, as spokesman for GWU, stated that no copying of copyrighted materials would be permitted on the premises. Hersch Pilloff discussed DOS 3.2. There were several calls for help on specific programming problems encountered by individual members. The meeting adjourned to the 6502 class by John Moon and to the exchange of programs.

Bernard Urban

Floppy Disk Storage Woes?

Several members have expressed interest in the neat mini-floppy envelopes which Mark Crosby, Dick Hodder and others have carried to our meetings. Well, Mark says he has a source willing to sell these in quantities of 10 or more at $6.40 per set. A set is ten 3-hole punched sheets of clear vinyl with two indexed pockets per sheet, for a total storage of 20 floppies ready for your three-ringed binder. Call Mark as soon as possible or grab him at our next meeting to place your order. The best time to order is now.

Something to Think About by Bernie Urban

Copies of a paper, "The Microelectronic Revolution", by Jon Roland, that appeared in The Futurist, April 1979, will be available at our next meeting. Roland, an independent consultant and microsystems analyst, makes several predictions concerning the effect of large scale integrated circuits on our lives. Here are some excerpts from his lengthy, thoughtful paper.

"Technology has dealt us another surprise, one that will soon radically transform the ways we live and do business and the skills we need...What has happened is the development and wide spread use of large-scale integrated circuits,... During the next few years this microelectronic intelligence is likely to be incorporated into almost every product large enough to contain it... Many of these products will become linked together by a world wide communications system into a vast network that will dominate our lives and fundamentally change the world in which we live."

Roland defines throughput density as a combination of gate density (the number of switching elements that can be packed into a given volume), switching speed (the number of switching operations per Δt), and transmission speed (speed of signals between switching elements). I have tabulated some of his predictions concerning future progress in throughput density.

Throughput Density
Progress by Orders of Magnitude

	Past 20 yrs.	Next 5 yrs.	To Reach Density of Human Brain	Theoretical Limit
Gate Density	4	2	1	4
Switching Speed				1
Transmission Speed				1
Subtotals		2	1	6
Total Theoretically Attainable				9
Probably Attainable				7

He states, "Remember that it only took two orders of magnitude improvement in the efficiency of energy conversion over a century to produce the Industrial Revolution. Devices that could bring three more orders of magnitude improvement in throughput density are already under development in laboratories and will be in production in seven to ten years. Even if all progress in this direction were to stop today, the impact would be revolutionary."

"If a pico processor (throughput density 1 million times greater than our micros-Ed.) could be combined with memory of comparable speed and compactness and the resulting pico computer implanted in a person's skull and interfaced with the brain, that person could have more computer power than exists in the world today and all the stored knowledge of humanity as accessible as any brain cell. Such a thing could fundamentally change human nature and it is closer to realization than bionic limbs, organs or senses."

Roland describes the coming personal computer network with everyone equipped with a dator (a Swedish term for computers and terminals). "A single such dator will be able to provide interpersonal communications, conduct financial transactions, report the heart condition of the bearer to the nearest medical facility and offer a choice of every variety of audiovisual entertainment."

What Roland alludes to but does not explicitly describe are the negative possibilities that also arise. I believe that the probabilities are high that many of his predictions will come about. I also believe that we as members of an "in group" with knowledge of microcomputers have a responsibility to ensure that the future he portrays does not become warped into one akin to George Orwell's 1984. The network linked together with a world-wide communications system can be used for negative purposes. The person with the implant can be converted into an automaton. The dators can be used to check on every action of the individual. He paints a rather rosy and utopian future but fails to point out the dire consequences if this technology is abused.

In his concluding paragraphs Roland writes, "The microprocessor represents a true revolution, and its potential impact on future job skills and prospects for employment needs to be understood by everyone. It is a useful exercise in forecasting to assume the general use within the next few years of personal computers as terminals, and then systematically to survey the businesses and occupations listed in the yellow pages of the telephone directory of any large metropolitan area, considering the likely impact of the microprocessor revolution on each. The result of such a survey is staggering: More than half of the occupations represented in such listings will cease to exist! Most of the others will be radically affected. Most businesses and industries will have to be extensively restructured, and many will not survive the change. The effect is likely to be traumatic. Many young people now training for their future occupations are likely to find that these occupations will no longer exist when they are ready to begin their careers. Contemporary society is facing the greatest occupational upheaval in history, and we need to plan for it."

I agree. Do you? What do you think our role should be in all of this? There is much in the paper that I have not covered -- considerably more food for thought. I urge you to read it.

Software Review: SINGLE DISK COPY by Howard Richoux

Program by: PERIPHERALS UNLIMITED, 6012 Warwood Rd., Lakewood, CA 90713 Ph. (213) 425-8752

After wading through many forgettable game programs and a number of business programs for the APPLE which didn't work right even as a demo, I am refreshed to find one program which fulfills a real need and works the way it should -- easily and reliably. As a computer system, the APPLE is a little unusual in that it's disk at 116K bytes is only 3 times as large as the RAM memory. Typically, this ratio is more like 100:1. What it means is that a whole diskette can be loaded into the APPLE in three pieces and by swapping diskettes in and out, a copy can be done in relatively few steps. This produces a disk which is an exact duplicate of the original including the volume number. The process is easy enough for me to have canceled my order for a second disk drive.

The steps are easy enough:

1. Boot in the DISK COPY Program.
2. Insert the disk you wish to copy-for the volume number.
3. Insert the target disk for initialization.
4. Insert the original to read a chunk.
5. Insert the target to write it out.
6. Repeat steps 4 & 5 until copied -typically twice.
7. The target disk is now identical to the original.

The program costs under $20 (about 96% savings over a second drive). I bought mine at COMPUTERS PLUS, but I'm sure it is at other stores also.

Nothing is perfect, so of course I would like a few improvements. What would really be nice would be a more sophisticated program to compress the files while copying. This would join together files which were fragmented by repeated use such as repeated saves of a BASIC program.

Locating Integer BASIC Variables by John L. Moon

Without going into detail on the way that Integer BASIC keeps its symbol table (that's subject to a much longer article than this), here is a routine that can be used to find the address of the data area of a BASIC variable.

INPUTS:

DIM ADDR$(30): ADDR=100
ADDR$="name of a variable": GOSUB ADDR

OUTPUTS:

Z6 = machine address of the data area of the variable. If the variable is undefined, then Z6 = -1.

LIMITATIONS:

Will not work as coded for string variable names. (It doesn't recognize the internal form of the $ in a string name; this is probably a pretty simple modification to make, but I haven't needed it yet.)

DESCRIPTION:

Uses variables Z2, Z3, Z4, Z6 and ADDR$. ADDR$ contains the name of the variable that is to be located. If the variable is in the BASIC symbol table, then the machine address of the beginning of the variable data area is returned in Z6. If the variable has not yet been defined, then Z6 is set to -1. Z2 will be returned with the length of ADDR$; Z3 will be the address of the end of the symbol table. Z4 is used internally as a scratch variable.

ROUTINE LISTING:

```
100 Z6=PEEK(74)+PEEK(75)*256:          Z6=Beginning of variables
    Z2=LEN(ADDR$): Z3=PEEK(204)        Z2=Length of variable name
    +PEEK(205)*256                     Z3=End of variables
110 Z4=0                               Z4 is an index into var. names
120 IF ASC(ADDR$(Z4+1))#PEEK           The search for a name equal to
    (Z6+Z4) THEN 140: Z4=Z4+1: IF       the value in ADDR$
    Z4=Z2 THEN 130: GOTO 120
130 IF PEEK(Z6+Z4)>1 THEN 140:         If found sets Z6 and returns,
    Z6=Z6+Z4+3: RETURN                 otherwise,
140 Z6=Z6+Z4                           skip this variable
150 IF PEEK(Z6)<=1 THEN 160:           by going down the name until the
    Z6=Z6+1: GOTO 150                  address of the next symbol is
160 Z6=PEEK(Z6+1)+PEEK(Z6+2)*256:      found. If names remain in the table
    IF Z6<Z3 AND Z6#0 THEN 110:        keep searching, else return -1 as
    Z6= -1:RETURN                      an error indication.
```

TYPICAL CALLING SEQUENCE:

```
1000 ADDR$="Z6":GOSUB ADDR: REM Z6= Address of Z6
```

or assuming DIM REAL(21) then

```
1010 ADDR$="REAL": GOSUB ADDR: REM Z6 = Address of REAL(0)
```

The resulting address can be used to peek and poke directly into the data area of an integer variable.

Floating Point in Integer BASIC by John L. Moon

My APPLE lives and breathes Integer BASIC. Oh, sometimes I delve down into machine language, but as for APPLESOFT, I've only loaded it into the machine twice. This is an effect of not having an APPLESOFT ROM card or the Disk. Besides, it wasn't until I started trying to do motion simulations and 3-D graphics and the like that Integer BASIC didn't have all that I needed. But finally, numbers from -32768 to 32767 just were no longer enough. So began my looking at the floating point routines in the Monitor. It didn't take all that long to figure out the entry points and the data areas that were used, but when I coded up a short BASIC program to poke numbers into the routines and call them, funny things started happening. Like nothing. Even BASIC wouldn't come back until I pushed reset. A quick check with the Red Manual memory maps confirmed my fears - the floating point work areas and the BASIC utility areas are sitting on top of each other!

I put my floating point desires on the back burner for awhile. Then, reading through "Peeking at Call APPLE" my interest was restimulated when I read how Don Williams had provided an interface to the floating point routines. Naturally, I didn't quite like the way he went about it (the not invented here syndrome, commonly abbreviated NIH) so I came up with the set of routines described below.

Before getting too far into ones and zeros, a description of just what floating point means to the APPLE is worthwhile. Floating point is just the computer's form of scientific notation. Using floating point, the computer does arithmetic much like people used to do with slide rules before the days of hand calculators. A number is divided into a "mantissa" which is typically represented in a signed magnitude format, and an exponent which, in the APPLE's case, is biased by $80. The number is represented as:

$$\pm \text{mantissa} * 2^{\pm \text{exponent}}$$

The floating point number takes up four bytes. The first byte is the exponent. The high order bit in the byte is the sign bit. Because of the $80 bias, 0=negative and 1=positive for the exponent. The next three bytes make up the mantissa. Since the "2" in the above expression is a constant, it is implied and therefore not actually coded within the number. The high order bit in the mantissa is the sign bit for the mantissa. It has the normal meaning of 0=positive and 1=negative. An implied binary point is in the high order byte of the mantissa. To preserve the precision of floating point numbers, they are kept in a "normalized" format where the number is left shifted until the sign bit and the highest bit of the number differ. The result for the high order byte of the mantissa is:

01.xxxxxx	positive mantissa high order byte
10.xxxxxx	negative mantissa high order byte

As I said above, the binary point is implied which means it doesn't actually take up any space. For each of the left shifts to normalize a number, the exponent is decremented so that the number value remains the same.

As examples, the following table gives several numbers and their floating point representation.

The following BASIC subroutine can be used to call the machine language routine:

```
20 FOR Z4=0 TO 3: POKE 4+Z4, PEEK
   (Z1-4+Z4): IF Z5 THEN POKE Z4,
   PEEK (Z1-8+Z4): NEXT Z4: POKE
   791,Z2: POKE 792,Z3: CALL 768
30 IF Z5 THEN GOSUB 40: IF Z1-4<R1
   THEN RETURN: FOR Z4=0 TO 3: POKE
   Z1-4+Z4, PEEK (4+Z4): NEXT Z4: RETURN
40 Z1=Z1-4: IF Z1<R1 THEN 50: RETURN
50 Z1=Z1+4: PRINT "STACK UNDERFLOW ERROR":
   RETURN
```

This routine pokes data into the locations 0 through 7 and sets up the JSR address for the machine language routine. The BASIC routine assumes that Z1 is the address of the memory being used as a floating point stack. R1 is the address of the bottom of the floating point stack. Z5 tells the routine whether the operation requires one or two operands (see table below for the correct values for each operation); Z2 and Z3 contain the low and high bytes of the address of the floating point routine. When the floating point routine returns to this BASIC subroutine, the subroutine copies the result back to the floating point stack. The correct parameters for this routine are:

FP Op	Z2	Z3	Z5	Hex Entry	Comments
Float	81	244	0	$F451	Converts 16 bit to FP
Fix	64	246	0	$F63D	Converts FP number to integer
Add	110	244	1	$F46E	Adds two FP numbers
Subtract	104	244	1	$F468	
Multiply	140	244	1	$F48C	
Divide	178	244	1	$F4B2	
Abs value	50	244	0	$F432	Absolute value of a FP number
Negate	164	244	0	$F4A4	

To use the floating point routines from machine language is not as difficult. The BASIC data area would not have to be saved and restored. I have given the routine addresses in hex so that they can be called directly and you can figure out from where the BASIC program gets and puts its operands where the data should go.

I have surrounded this rather austere interface with a number of goodies such as conversions from ASCII to floating point, printing of floating point results, as well as predefined BASIC routines to call each of the operations and manipulate the floating point stack. The entire routine and its documentation is much too large to put in a single article but I will bring it to the next meeting for anyone who wants to copy it.

Next month I'll delve into Sweet 16, Wozniack's "Dream Machine" that also lives inside the Monitor without appropriate documentation.

```
10.0        83 50 00 00
1.0         80 40 00 00
0.1         7C 66 66 66
0.0         00 00 00 00   (zero is a special case)
-0.1        7C 49 99 9A
-1.0        7F 80 00 00
```

The difficulty with using the floating point routines from Integer BASIC stems from the floating point usage of the same data area as BASIC. The floating point registers are located from $F3 through $FC, the same area that BASIC uses to keep track of things like which line number is being executed, the BASIC loop counters and subroutine stack pointers and other valuable stuff. To solve this, a machine language program is necessary to move the BASIC utility area off to a save area, then to copy the floating point operands from a parameter area into the floating point registers, call the routine, return the answer to the parameter area and then restore the BASIC utility area to its original value before returning to BASIC.

The following machine language program assumes that locations 0 through 7 have the two floating point operands and that some BASIC program has put the routine address into the JSR prior to calling the routine. The results of the floating point operations are returned to locations 0 - 7.

```
300- A0 F3     LDY #$F3          Prepare to move utility area
302- B9 00 00  LDA $0000,Y       To a save area in low memory
305- 99 1C FF  STA $FF1C,Y       So that the floating point registers
308- C8        INY               Can be used without bombing BASIC
309- D0 F7     BNE $0302
30B- A0 07     LDY #$07          Move the floating point operands
30D- B9 00 00  LDA $0000,Y       From 0-7 to the registers
310- 99 F4 00  STA $00F4,Y       So that they can be used
313- 88        DEY
314- 10 F7     BPL $030D         Call the floating point subroutine, the
316- 20 00 00  JSR $0000         JSR address must be filled in by BASIC
319- A0 07     LDY #$07          Return operands to 0-7
31B- B9 F4 00  LDA $00F4,Y       For the BASIC program to pick up
31E- 99 00 00  STA $0000,Y       the answers
321- 88        DEY
322- 10 F7     BPL $031B
324- A0 F3     LDY #$F3          Restore the BASIC utility area
326- B9 1C FF  LDA $FF1C,Y       To the original contents from the
329- 99 00 00  STA $0000,Y       Save area
32C- C8        INY
32D- D0 F7     BNE $0326
32F- 60        RTS               Return to BASIC

3F5- 20 2D FF  JSR $FF2D         On overflow, sound bell, print "ERR"
3F8- 60        RTS               and return
```

Apple Users—New Shape Designer!!

Now—a new program that takes the tedious, time-consuming coding of shape vector tables and makes it fun! SHAPE DESIGNER does all the coding while you hit U, D, L, R keys (up, down, left, right) as you draw. Then you name the shape and it is automatically saved on your disk. Later, you can assemble up to 255 shapes into a single SHAPE TABLE, again automatically, and save that on disk too! All shapes are drawn on the Hi-Res screen during program execution as verification. Two additional utility programs transfer shapes from disk to disk and display them anywhere on screen for design use. Documentation explains how to use the SHAPE DESIGNER and how to use SHAPE TABLES in your own programs (it's marvelously simple). Now you can create your own Hi-Res games, architectural drawings, and anything requiring shapes. A long-awaited program, the SHAPE DESIGNER is a must for all APPLE owners. You need Disk II and a recommended 32K.

Shape Designer (on disk)

$15.95 (include payment with order, no charge)

Research Associates
1373 "E" Street S.E.
Washington, D.C. 20003
(202) 488-1979

Music for Greenapples by Rick Hodder

This month and next I will be showing you how to put music in your programs. First of all, look on page 45 of the Red Reference Manual and you will find a subroutine for music in both BASIC and machine language. If you use the machine language subroutine, you will have to BSAVE it and reload it before or during the program. Otherwise, in BASIC you must branch (using a GOSUB) to the subroutine. Then to define a note, you must set the pitch by POKEing it into memory location 0 and duration by POKEing it into location 1, e.g. POKE 0,(pitch value) and POKE 1,(duration value). For the pitch value, I will show you the "G" scale this month. For the "G" starting below middle "C", the pitch is 255(A=230, B=202, C=190, D=170, E=151, F=134 and G=126). The duration value for a whole note is 150, a half note is 120, a quarter note is 100, and an eighth note is 50. See you next month. ************************

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% NEXT MEETING OF WASHINGTON APPLE PI
% Saturday, May 19, 9:30 am
% GEORGE WASHINGTON UNIVERSITY
% Tompkins Hall, School of Engineering, Rm 206
% 23rd & H Streets, NW
% Parking roulette, or in students' parking lot,
% if chains are down. Convenient to Metro.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

CALENDAR OF EVENTS

Date	Events/Meetings	For Further Info. Call
May 17, 18	Conf. on Microcomputers in Education & Training, Pentagon Quality Inn, Arlington	Ray Fox (703)347-0055 Off.
May 24	NOVAPPLE 7:30 PM, Computerland, Tysons Corner	Jim Nielsen Off. 693-7530
May 25	Computerland APPLE Users Group 7:30 PM, Computerland, Rockville	Kim Brennan Off. 948-7676
May 31	Chesapeake Microcomputer Club 7:30 PM, White Oak Library	Mani Alexander Off. 452-5232
June 13	Assn. of Personal Computer Users 7:30 PM, Chevy Chase Library	Daphne Schor Off. 544-8530

Biographical Sketches

Thomas Bottegal - Full-time work in computers for over 5 years. Familiar with 6502, 6800, 8080, Z80 computers. GWU courses currently teaching:

CS 51 Intro. to Programming (HP BASIC, APPLE BASIC, FORTRAN
CS 52 Computers and Societal Problems
CS 157 Intro. to Assembly Language (8080, IBM 370)
CS 158 Intro. to Algorithmic Methods and Higher Level Languages
CS 201 Graduate course: CS 157 and CS 158

Interests: Computer education, General usage of home computers.
Computers owned: none (collect pocket calculators).

Mark Crosby - Age 31. 7 years in trade association dealing with adult career education. Previous work in time sharing environments. Former music major - current electronics nut - like plants and animals - designing solar heating system for my home.

David Itkin - Age 17. 12th Grade. Have used terminals in school since 8th grade. Our school now has an OSI which I play with. I have an interest in electronics and ham radios.

John L. Moon - B.S. in Computer Science, Univ. of Florida; graduate work at GWU in Computer Science. Member of ACM. Manager, Software Architecture at IBM FSD, Manassas, Va.

Robert Peck - Lieutenant Commander, Medical Service Corps, U.S. Navy (Retired 1 May 1979). 15 years as enlisted hospital corpsman and 15 years as navy officer in health care administration. Officer in charge of 6 regional medical clinics in Washington, D.C. for the past 1 1/2 yrs. Administrator, Armed Forces Institute of Pathology, Wash., D.C. for 5 yrs. Fiscal and Supply Officer, National Naval Medical Center, Bethesda, Md. for 3 years. Member of Assn. of Records Managers and Administrators. Past Vice-president of local chapter and currently member of Board of Directors. Past National Chairman, Registration Committee 1978 ARMA Conference.

Bernard Urban - Programmed and designed systems for computers since 1951. Computers I have known include the SEAC, UNIVAC, 1103, CIRCLE, IBM 704, 709, RCA 301, 501 and 601. I still design man/machine systems on occasion but I no longer program at work. I consider myself an information specialist and my main interest is in fostering the creation of information exchanges and clearing houses, e.g. community development and urban affairs, microcomputer technology, CAI for the handicapped and learning disabled. I currently serve as co-chairman of Task Group #12 "Handicapped and Special Education" of the ACM Elementary and Secondary Schools Subcommittee.

Genevie Urban - Programmed and designed systems for computers for 15 years before stopping to raise a family. I worked with SEAC, UNIVAC, IBM 701, 705, 709, 7090, 7094. My present interest in computers is, of course, with micros and particularly in education.

Enter Addresses

record change of address

be able to print a printout
of addresses

Query by name
by first letters

Name, qualifiers
address
city
state
phone #'s
notes