THE PAPER

VOLUME IV, Issue 2 A CSD PUBLICATION $3.00

Table of Contents

++

Subscription Codes

Please check the one letter subscription code found after your subscriber number on the mailing label. It indicates your status as of November 20. The interpretation is as follows:

P - Paid - Thank you
B - Bill - This IS your bill
S - Survey - You did say 'YES!'
L - LIPS - A loyal member?

If your status is not P for PAID, this is your bill for Volume 4. Sorry, we cannot afford to continue sending out reminders. We hope everyone enjoys this issue and looks forward to those remaining.

++

General Information

The PAPER is published 6 times per year by Centerbrook Software Designs at Pearl St., Livingston Manor, NY 12758. Telephone: (914) 439-3591.

New subscribers will receive all issues of the current volume. Single copy price is $4 and the subscription price is $20 for all the issues of the current volume. Subscription orders should be mailed to The PAPER, Pearl St., Livingston Manor, NY 12758

Third class postage is paid at East Setauket, NY 11733 (Permit #96). POSTMASTER: Mail all address changes to the address above.

The PAPER and Centerbrook Software Designs are in no way associated with Commodore Business machines. CBM is not responsible for any of the contents of The PAPER unless otherwise noted. PET and CBM are trademarks of Commodore Business Machines.

All readers are encouraged to submit articles of general interest to PET users. Materials submitted must be free of copyright restrictions. The contents of The PAPER are not copyrighted. All articles remain the property of the author and may be reprinted with their permission. When reprinting please include a note stating that the article was originally published in The PAPER.

Subscription Rates:

USA third class: $20/volume
Canada first class: $25/volume
Foreign surface: $30/volume
airmail: $40/volume

Payment in check or money order in US funds must accompany all orders. Only prepaid purchase orders will be accepted. All checks should be made out to The PAPER. Sorry, we cannot accept bank or credit cards.

Advertising

Advertising rates are $25 per quarter page per issue. Copy must be camera ready or there will be an additional charge. Special rates may be negotiated.

Circulation

Volume 4 Issue 1
Printed: 2200 USA: 612 Canada: 44 Foreign: 25
Conferences: 350 Commodore: 400 Dealer: 500

Software

Software published in The PAPER is believed to be free of copyright restrictions. It is meant to work on the machine indicated. Many programs were originally designed to work with only one ROM set but efforts have been made to convert them to work on all present ROM releases.

Staff:

Publisher: Ralph Bressler
Editor: Doug Haluza
Assoc. Ed: Roy Busdiecker
Vic SantaLucia

Staff writers: Bill Batcher
Ken Barroll
Jim Fowler
Jerry Key

The Late Great Commodore (?)

At one time when people would ask me "What microcomputer should I buy?" I was able to answer "The Commodore PET" with no hesitation. I felt this was the right machine for a variety of people and I could give many reasons why. Lately, I have begun to question this recommendation and cannot answer so quickly. Do you have the same feeling? My questions are a result of three major factors. First, Commodore still does not offer the kind of support I would like to see particularly in the educational market. Also, the software available for some other machines is very attractive. Finally, new machines by IBM, NEC, Monroe and many others have very nice features at competitive prices.

Near the beginning of September I talked to several people at Commodore's King of Prussia headquarters. They told me how they had recently formed a group to help educators using the PET. They would offer some technical support for the hardware and additional assistance with instructional software. Two months later I've heard nothing else and am convinced this was another good idea that just died. Commodore's sole contribution to education has been the 3 for 2 deal which does make good machines available at the best price around. But how long can Commodore expect to sell hardware without any support for software?

Some of the instructional packages for other machines are really quite impressive. I do not agree with those that say there is NO good software for the PET. However, the development of good educational materials for the PET has simply not matched that for other machines. Apple has complete instructional systems which pretest the student, provide drills and keep track of his daily progress. Atari has introduced programs which use their recorder's audio track to "speak" to the user in a human voice. Even Texas Instruments has their foot in the door with a very good version of LOGO. Many larger publishers may lean toward the TI machine since software packages can be placed in a ROM cartridge which is virtually impossible to copy. Much of the good software that exists for the PET has been developed by individual teachers and is not widely available on the market. Many big publishers and audio-visual companies are entering the micro software market. When they choose which micros to support, what will their choices be? How long can Commodore survive in education without strong support for software developers? Every change in the PET has been accompanied by silence on the part of Commodore which frustrates the software developer who wants to offer support and have versions for all machines.

I do feel Commodore can compete with the others in the small business market. WordPro and WordCraft are two good word processors. JINSAM is an excellent data base manger and others are available. Many packages exist for general ledger, accounts, and inventory control. Still, the greater memory capacity of Apple and TRS-80 make bigger and, in some cases, better programs available. Apple and Tandy's advertising also makes them seem more prominent than Commodore.

The new generation of micros like the IBM Personal Computer and the new Monroe Educational System offer many features not found in the PET or some of the other "old" micros. These include nice color graphics with high resolution screens, more RAM memory and integral disk drives. They cost more and the software available is still limited but I feel they are competitive for some parts of the educational market. The exact effect on the market of the low cost IBM machine is still unknown. However, their uncharacteristic method of distribution through other stores and their reliance on outside software sources will give them greater appeal than if they had tried to go it alone. The move by IBM lends added respectability to micros and encourages the remaining holdouts to buy microcomputers. The market will certainly expand but the new products will also cut into someone's market share. Will it crowd out the Apple III even though it has been recently upgraded? Will the Radio Shack Model II show some decline in sales? What will the effect be on the sale of the new Commodore 8096 and 2 megabyte drive system?

I still feel that the PET is the best machine for teaching programming to students. Its low cost puts the maximum number of machines in the hands of students. Its screen editing and reliable cassette system make learning to program easy. To teach word processing an 8032 with a disk drive is a good choice and even a 40 column screen is adequate. Many excellent programs also exist to teach accounting and data processing and for this reason CBM is also a good choice for the small business. Commodore seems to be trying hard to introduce hardware to meet the needs of a variety of present or future users. The recent introduction of the VIC with a line of accessories to follow has raised the spirits of Commodore supporters. This machine is, in some ways, superior to the PET. At the other end of the spectrum the 8096 upgrade and larger capacity disk drives have made the CBM more attractive to business customers. An unknown factor in Commodore's efforts to capture more users and a larger market share is the new SuperPET. If this machine can be delivered on time without delays, it may prove to be a real coup for the company. Few other micros can match the performance to price ratio of this machine and its language systems.

I think all the "old" micros will be pushed hard by new entries like IBM and some of the Japanese models. Exactly how Commodore will be effected depends on their willingness to support the hardware they sell with software. There must be a cooperative effort between Commodore and software developers if either are to survive. Many have predicted and many have feared that Commodore would go under at any moment. We who support the PET can only hope that the introduction of new products by Commodore, and better support of their products, will be in time to counteract the bad image Commodore has earned.

Copy This If You Can

This almost seems to be a challenge issued by some software houses. I know that the issue of software protection has been discussed in these pages and in many other publications but a new issue has come to my attention.

A school wishes to use a particular drill package in several schools and in many different classrooms. Let's say they have 6 sites with three machines at each site or a total of 18 machines. As things stand now with most companies they would have to buy 18 programs. At, perhaps, $50 a program this is $900 and many packages are much more. If the program is not protected, will the school buy one copy and make 17 others thereby saving $850? If it is protected, will the school still buy it, and if they do, will they feel cheated? In fact, some companies go as far as to include a "one machine-one program" statement. This means that the program you buy may ONLY be used on ONE machine EVER. This is not a hypothetical situation. Many schools have acquired a number a machines on one-time grants and are now trying to purchase software on limited school district funds. I know of one school that wished to set up word processing for 12 stations. When their dealer called the program vendor, the answer was "buy 12 copies". This would cost about $3600 which they could not afford! What is the solution?

Mike Markkula, president of Apple Computer, recently sparked controversy at the NE Computer Show by stating that he felt it was time to end software protection. I agree wholeheartedly. Too much time, effort and money is spent thinking of clever ways to protect programs. Many of the protection methods use of valuable "chip space" uselessly. Also, most of these methods can and have been broken. Companies must begin to offer quantity discounts to schools and relax stringent license requirements. Schools must realize that a small disk or tape may represent 100's of hours of programming. They must learn to include money in their budget for software, as well as, hardware. This kind of atmosphere will generate more and better programs for all users but will allow the software vendor to make a reasonable profit. Isn't that what we all want?

Reader I/O

Ralph - I own a 16K PET 2001 and a CBM 2040 Disk Drive which I recently had expanded to 32K and upgraded to BASIC 4.0 and DOS 2.1. I had been using a Tool-Kit, which I replaced with Command-O. My problem is that when I turn on Command-O, I can no longer use the Wedge or my Keyword program (COMPUTE! Aug 1981, pp. 120-122). I don't know machine language (yet), so I don't know how to relocate either of these routines. - Dave Sessions

Dave - I'm afraid all I can offer is bad news. First, let's find out what everything does and where it sits. Universal Wedge sits at the top of memory and uses the CHRGET routine to check what you type in for its own commands. Keyword lives in the second cassette buffer and uses interrupts to determine when certain single characters are typed. Command-O's address is $9000, one of the ROM expansion slots and it uses BOTH interrupts and CHRGET to do all that it does. Notice that there apparently is no conflict in where the three things reside! However, Command-O uses a good part of the second cassette buffer to record its own information and what it is doing and will overwrite Keyword. Keyword and Command-O both use interrupts so there's another conflict. Finally, Command-O and the Wedge both use CHRGET which really botches things up. There is NO simple way to change these routines or just relocate them. My suggestion is use the new BASIC 4.0 disk commands or the SEND command from Command-O. For now you'll have to forget Keyword, I guess. It may be worth it to look at the new POWER package. Jim Butterfield showed it to me and it looks much more powerful than Command-O.

Ralph - The version of KEYPRINT which appears in Volume 4, Issue 1 will not work on the 8032. To make it work with the ESC key only three changes are necessary. Readers may use the monitor to change $036A to $9B, $03D4 to $50 and $03DA to $4F. The corrected version must then be resaved using the monitor. In order to use the program it must be loaded before any other program is loaded. If a program is saved KEYPRINT is destroyed. - Ed Steinfeld

Ralph - The KEYPRINT 4.0 program from the last issue will not work on the Fat-40. When I press the backslash I get one screen dump and then the PET hangs up. - Mo Judson

Ed - Thanks for this correction. The 8032 is one machine I do not have direct access to and cannot test programs on. A similar solution will NOT work for the Fat 40. The locations from $033A to $0359 are used for the disk commands while $03E9 to $03F9 are used for tab stops. The program is two bytes too long to fit in this space. It could be relocated back to $033A as long as you do not use the BASIC 4.0 disk commands. If anyone is interested let me know. Remember that you can use any key to activate the program using the method explained in the article. Once the changes are made type SYS 4 to enter the monitor then type the following to save the corrected version. - Ralph

```
   .S "0:KEYPRINT 4.0F",08,035C,03ED  for disk
   .S "KEYPRINT 4.0F",01,035C,03ED  for tape
```

Ralph - I wish to operate a Commodore 3016 and 3040 disk drive from a marine 12v battery via wiring modifications and outboard circuitry if necessary. The result should allow the conversion back to the usual AC power, but it need not be switchable between the two (although this would be an advantage). Can you supply the following?

1. Details of required DC power supply voltages, current capacity and

regulation specifications for the two units.

2. A tested circuit by someone else who has done it.

3. The name and address of someone whom you feel might have the answers to some or all of my questions.

I realize that a 12v to 240v inverter of 200 watts capacity is an alternate, but the direct approach is preferred. - Francis Chambers, Rock House, Ballycroy, Westport, Co Mayo, Ireland.

Frank - First, let me say that I generally am able to insert and extract IC's but most of my soldering and tinkering is done by someone else. In the States, of course, we have 120v AC at 60Hz which is different from your setup. I turn this one over to other readers. Can anyone help. You might contact an outfit that offers a battery backup for the PET and drive. They might be willing to help with your problem. Try writing Electronic Technology Corporation, PO Box G, Old NC 42, Apex, NC 27502 - Ralph

Ralph - I have recently installed the MTU Visible Memory board in my PET along with a fan to reduce the heat. It took me almost a week to get the hang of all the instructions so I could get it running. I'm still not sure what I will use it for. I am particularly interested in your comments about using the ML monitor to save the code contained in special ROMs such as WordPro so that these sockets can be freed. Can you please tell me exactly how to do this? - Larry Riley

Larry - Sorry, its taken so long to answer. I'll explain what to do based on the assumption it will work on your setup as it does on mine. First plug the chip you want to use into the properly addressed ROM slot on the MTU board. Let's assume its WordPro and you plug it into slot 2. WordPro sits at $A000 and that is what this slot is set up for. Now POKE 48895,32+16. This enables slot 3 which I believe has your $B000 BASIC 4.0 ROM in it and slot 2 which has the WordPro ROM at $A000. Now type SYS 4 to get into the ML monitor and type .S "0:WORDPRO ROM",08,A000,B000 or .S "WORDPRO ROM",01,A000,B000 and then .X to exit. This will save the ROM to disk or tape respectively. This is probably overkill since most of these chips use only 2K and we saved all 4K. You might try A000 to A800 and see if it works. Now pull the ROM, reset the system and type POKE 48895,32+4. This enables the 4K of RAM that sits at $A000 on the MTU board so that you can load the Wordpro ROM code into it. Now load the ROM you just saved as you would any program and type NEW. Load the WordPro program and everything should work.

For other ROMs you must know the beginning address and use it instead of A000 as we did above. Some ROMs will not work this way since the program tries to write to them as an added protection feature. If it's a ROM then no writing occurs but if it's saved to RAM things get changed and will no longer work. Visicalc must do something like this since I have yet to save it so that it works in RAM. I have been able to defeat some of these chips but not all.

It must be made very clear that I BUY ALL these packages and THEN save the ROMs. I am not trying to copy a ROM so that I can use the programs illegally. I simply have too many chips to fit in my PET. Of course, it could be used illegally but it seems to me that a $400+ MTU board would be pretty expensive to use just for pirating software. - Ralph

Ralph - There is a bug in DOS 2.5 (8050 disk). COPY D0 TO D1 will only work if the IDs of BOTH disks are the same or if all the files to be copied are listed in the same directory block -- max of 8. The only way is to copy one program at a time. My program to do this is so slow I'm ashamed to send it to you. I need a faster way to read the directory.

I like the assembly language articles, but there is something I would like to learn. How do you use some of these built in routines? An article might say "get a character from the disk using SO&SO routine a $XXXX". Fine, where did SO&SO put the bloody byte it got, how do I find the string I want to put it into, and how do I do that? Was I supposed to have OPENed a file? Must I CLOSE it?

Finally, how about a moratorium on routines that fit in the cassette buffers? DISK-O-PRO, COMMAND-O, BASIC 4 and darn near everyone else have taken them over as scratch pads, and ruin all the cute little routines we stuff in there. - R. Vanderbilt Foster

RVF - Now I know why when I tried to COPY a disk at local dealer recently I got a DISK ID MISMATCH after about 8 files were copied. BACKUP worked fine in this case. I know what you mean about the use of the built in PET ROM routines. I will try to encourage authors to explain EXACTLY what they did and HOW they did it. This is important because many of us would like to go back and use that same routine for our own purposes. Also, the use of these routines is not well documented anywhere else. I also have been disturbed by the seemingly endless use of the cassette buffers as "holding" areas. It was easy to write small routines to fit in this area since we knew everyone had it on their machine. The only alternative is to load the routine into high memory. The problem here is that there are still 8K, 16K and 32K machines which means the code must be relocatable and instructions must be given to do the relocating. Any suggestions? - Ralph

Ralph - I am enclosing a letter from a PET user in Peru who is facing a lack of documentation even worse than that encountered by the average American PET hacker. Some readers might like to respond and help out a fellow programmer and gain an interesting pen-pal in the process. As a bit of inside dope, I can tell you that the Code Works will soon release a book and accompanying disk with The Best Of Cursor, it should be out soon. - Randall Lockwood

Randall - Thanks for the inside information on Cursor. Rather than print the letter as is, here is a condensed version with the address. I would also encourage everyone to write. This is the first time I have ever heard of someone "south of the border" having a PET.

Dear Sir - We hope by writing this letter to contact a circle of people who know something about the PET microcomputer. We have done a lot of programs for larger machines like IBM and NCR in languages like RPG, COBOL and FORTRAN. To learn these machines and languages we had many books from our libraries. However, the support for microcomputers and BASIC in our country is not yet very great. We need to know about instructions like SYS, POKE, PEEK and so on and about how the PET uses them. Please send any information you can to: Jorge Zavala Vinces, Casilla 1230, Arequipa, PERU

Ralph - There is a small bug in Doug's program DIRECT III in Volume 4, Issue 1. Users may have noticed that it doesn't work properly with a program code number over 800. The change is at $036C which should be $90 instead of $10. This replaces a BPL with a BCC. Readers may correct their programs by typing in the new symbol in the monitor or by using POKE 867,144. Be sure to save the corrected version. The error is one I have often made. It is due to the confusion between what the CPU says is a negative number and what you and I call "negative". - Jim Fowler

Jim - Thanks for the correction. Users should remember that the carry will remain set when a smaller number is subtracted from a larger but will clear when a larger is subtracted from a smaller. This is a borrow.

Background/Foreground

Subscriptions Revisited

Subscriptions to The PAPER cover one volume from beginning to end. We are trying to publish every other month so this would mean at least 6 issues per year. All prices that we quote are in US dollars only. The cover price will be $4 which means 6 issues would cost $24 in the US. Subscription prices are as follows:

US	$20	FOREIGN surface	$30
CANADA	$25	FOREIGN air mail (AO)	$40

These rates reflect the great difference in postal rates between the US and everywhere else. From time to time we offer special discounts but otherwise these are the rates.

Several people have written that their copies didn't arrive. Please make sure your address is correct and that nobody intercepts your issue. Other people have asked for an additional copy because they lost theirs or it was damaged. Please send $3 for any additional copies you need to cover our costs.

International MTU Users Group

Many of us who have the MTU Visible Memory board have put it to good use. However, we know it is capable of MUCH more. DENSPET is the international association for the exchange of programs for the MTU (and IJJ) 200 by 320 dot high resolution PET accessory. Send a sample of your work or $5 and receive a sample in return plus a subscription to the newsletter and lists of available programs. Write: Frank Chambers, DENSPET, Rock House, Ballycroy, Westport, Co Mayo, IRELAND.

A note from Frank explains that the IJJ board is the same as the MTU hardware. Their software package is called PETGraph and is different from the MTU Keyword Graphics package. DENSPET bulletin #1 compares these two in detail.

PRINTOUT Prices Reduced?!

A recent note from Julian Allason, publisher of PRINTOUT, tells us that the subscription rate for his magazine here in the US has been cut almost in half. A full year's subscription is now only $29 as compared to $54 last volume. The issues will be air-speeded rather than sent by standard airmail. I can atest to the fact that this arrangement has not slowed down delivery any. This is a real bargain and all dedicated PET users MUST take advantage of this offer. If your interest is solely Commodore then this magazine is for you. It has much better PET coverage than any other magazine including COMPUTE!

SATN

Software Arts, the producers of VisiCalc, have recently announced a newsletter dedicated to the users of their million selling program. The ad rubbed me the wrong way because they were charging $30 for six issues of something I felt they should have supplied free.

I did send the money and after a short wait received the first two copies which were packed with information. Of course, the manual tells you about all the commands and does a good job. The newsletter gives you suggestions and practical applications. The first issue was dedicated to the @LOOKUP command which I had never used. A newsletter for a single program may seem odd but it just points out how versatile the program is. If you want to use Visicalc to calculate job costs, figure withholding tax, establish mortgage tables or have some fun with mathematical problems and sports team standings this publication is for you! Send for information or $30 for a subscription to SATN, PO Box 494, Cambridge, MA 02139

ABACUS Releases

I rarely make new product announcements until I've seen and used the product and then I review it. But when a company has proven itself like Abacus Software, I feel confident giving them a little space. They have just released three programs for the VIC and a new one for the PET.

The PET/CBM program is a Tiny BASIC Compiler which reads your BASIC program and produces 6502 machine language code. Abacus says it supports all floating point arithmetic and functions and will run on any 40 column PET with 8K. The price is $25. The question now is, "What separates this from others selling at $350?". VIC Piper allows you to compose, save, edit and play music with the VIC. Most factors effecting the music can be controlled and pictures may be printed to illustrate the music. Cost is $25

VIC VIGIL is a version of the popular graphics language for the VIC. A set of commands has been added to control all the VIC features like sound, light pen, game paddles and joysticks. Nine preprogrammed games are provided. A 3K memory expander is needed but the price of VIGIL is only $35. See the review of the PET version in this issue for more information.

VIC HiRes/Multi-color are utilities which allow you to explore these two modes on the VIC. HiRes allows plotting in 104 x 152 positions. Multicolor gives you the same ability and an additional color in a 52 x 76 size screen. Sample programs are included and both utilities are $20.

Contact Arnie Lee, Abacus Software, PO Box 7211, Grand Rapids, MI 49510

Letter Quality Printers

I have now used several letter quality printers to produce this newsletter and all of them are expensive, some less so than others. Each printer has some advantages and disadvantages. These usually center around quality of type and speed of printing with these two factors seeming to have an inverse relationship.

I am convinced that for large volume printing of good quality the NEC Spinwriter is the best. This issue was printed with one. It is also really heavy duty and has a reliable ribbon and paper feed. The last issue was produced with a Diablo which was just back from the factory. It caused no end of trouble and was less reliable than the NEC. However, I have heard others say that they have never had trouble with their Diablo and they swear by it. For overall great print quality and the speed of moving a mountain you have the IBM line. They produce the best quality print and do it consistently but at a rate that makes you yearn for dot matrix. All of these printers are well over $2000 by the time you get them interfaced correctly.

In the "under or close to $2000" price range we have the C.Itoh Starwriter. The first comments I heard about this machine varied from raves to curses. Bob Baker who writes for Kilobaud felt the one he had rivaled the Spinwriter in almost every way. Most recently I have heard only good things. Another choice which I have seen lately is the Olympia ES-100 which has been used in Europe for some time. There are a few tricks to getting this to work with WordPro but they can be overcome.

Observations on Vol. 4, Issue 1

by Roy Busdiecker

This certainly was an exciting issue for me. Due to a little mixup, my first realization that Volume 4, Issue 1 had been published was a note from Ralph telling me that time was running out if this column was to be included in issue 2. After a quick, and probably too abrupt, note back to Ralph (my apologies, Ralph!), the issue showed up (first class mail!). After a quick glance through it at dinner, I'm now seated in front of my 8032/WordPro4+ system doing a simultaneous read (from The PAPER) and write (to the Observations). Am I spooling myself? Anyhow, if this installment seems either rushed or disorganized, you know that's exactly what it is.

General Comments

The PAPER's appearance continues to improve. I noted the effective use of enhanced print to emphasize key items and generally dress-up the page. Thanks to the folks at American Peripherals for their contribution to our favorite newsletter/magazine.

Here's a "Gotcha!" Page 2 says each issue in the current volume costs $4.00. Now look at the front cover and see what IT says!

We still need to work on circulation! There are literally thousands of CBM-family users who could benefit from reading The PAPER. Spread the word ... advertising is expensive, and word of mouth is more effective. Speaking of advertising, I'd like to see more of it in The PAPER. When you order something you saw there, be sure to tell the folks you're buying from. They often don't have any idea which ads are working for them, and are more likely to continue those that bring feedback. And if you order from someone who DIDN'T advertise, suggest that they consider it. I do not get a commission on ads or subscriptions ... I'm just interested in seeing this publication continue. As I've mentioned frequently, there are many unique qualities to a newsletter that you just can't get from a "slick" magazine.

Congratulations

Ralph's move I knew about, but not the arrival of Kurt. I'm sure all readers join me in congratulating the new (again) parents, and wishing the entire family well. Combining the household move with the arrival of a child certainly makes life interesting! (I speak from experience ... our firstborn arrived 13 hours after we moved and guess who got to do all the unpacking!)

Commodore Products

VIC-20 has been joined by the VIC Printer ($395), now on the shelf at your local Commodore dealer (at least if you live in Northern Virginia and patronize Virginia Micro Systems, Inc.!). Ditto for the CBM 2031 single disk drive ($695), which uses the 4040 format. Latest versions of the 8050 drive that we've seen at the store have double-sized analog boards with two unused head plugs in addition to the two currently in use ... wonder what it will take to make it work double sided. That confirms the report that 2 MB storage is near.

The Micro Main Frame is now being called SuperPET ... apparently someone else had prior claim on the original name. We haven't received any yet, but they're supposed to be coming soon. Commodore did have one at the Federal ADP Expo in Washington, DC, in September. The CBM 8096, on the other hand, IS available from dealers as an upgrade to the 8032 ($500). Before you go running out to buy one, make sure you know what you're getting and that it will do what you want. The bank-switching arrangement means that, while the extra memory is available, you only get access to one 32K segment at a time ... postpone your dreams of seeing "93184 BYTES FREE" when you turn the power on.

In the last issue, I mentioned the popularity and value of the Epson MX-80 printer. Subsequently, there was an article in a trade magazine saying that Epson had surpassed Centronics (by a large margin) as the #1 company in terms of printer sales. Now the MX-80 has been joined by the MX-80 F/T, which provides either friction or tractor feed at about $100 over the cost of the MX-80; and by the MX-100, which provides the F/T feature AND 132-column width at normal print size. The 100 is about $300 more than the plain 80.

Computer-communicators will be excited by the DC Hayes SmartModem, which provides direct connection (no need to put your handset in an acoustic coupler) and Auto-Dial ... that's fun! The McTerm terminal package works nicely with it, but does not support the Auto-Answer capability of the SmartModem. If any of you hackers get that capability added, please drop me a line ... or phone (703) 491-6502.

You Can't Please

The results of the comments on the reader survey were very interesting. Most of them, I agreed with (except the one that sais "no need for a new name"). The original name of the publication was The PET PAPER, which had the benefit of giving some indication of what it covered. The "PET" was dropped for fear of some action by Commodore ... but that was during a previous administration at the company, and the PET name has now been used by a number of publications. As a matter of fact, The PET Gazette (now part of COMPUTE! magazine) used the name right on through the time when The PAPER dropped it. I'd still like to see the PET reinserted in the name of The PAPER.

From reading the other comments, I'd say that Ralph erred in titling the section "You Can't Please". It's obvious from the range of positive remarks and the varied interests of us readers that The PAPER DOES please. And by the way ... some of those articles you'd like to see can best be written by you! You write the one you know about that someone else wants to see, and eventually one of the others will write the article you want.

Input/Output

The first entry in the Reader I/O section was one of those little treasures that's worth the price of the issue! Thanks Bill for the question. Thanks, Ralph for the answer. Boy, am I glad you got interested in machine language enough to fix those Cursor programs! Next time can we have the fix for Aliens!? (Roy - I think its the same fix! For the Fat 40 the instructions which deal with sensing the keyboard must be changed. More later when I figure it all out. - Ralph)

Joe Spatafora's request should be taken a step further ... share the best VisiCalc applications through articles in The (PET) PAPER!

Eugene Smith's comment about Papermate reminds me ... in the last issue, I endorsed the 8032/8050/WordPro4+ as one of the best professional word processing systems around. The second endorsement, for one of the best low-cost WP packages consists of 4016, 2031 or cassette, Papermate, and MX-80. With cassette, you should be able to get that package for around $1800 from a dealer (you may pay less by mail order but a dealer is safer ... caveat emptor!). The single exception in which I have confidence is AB Computers (for you newcomers, Gene Beals of ABC founded and ran PET User Notes, one of the first and most popular PET newsletters).

Hope Jim Yost's monitor project succeeds, and that all you machine language enthusiasts are giving him the help and support he needs. Hey, Ralph ... how about an "Ultimate Monitor" contest? And, if Micromon is public domain, how about at least a monitor dump in th next issue, so the rest of us can try it? (Roy - Micromon is in the public domain but is too long to list here. It's too frustrating to try to type in a long program only to find out a byte is wrong somewhere. The best source is CIPUG whose address was given in the last issue. Their version is greatly enhanced and comes with instructions. - Ralph)

Music and Data Bases

Just took a brief time out to check another treasure ... the fix to AB Computers' Visible Music Monitor. Sure enough the POKEs that made VMM work on the 4.0 upgrade also solved the big problem for the 8032. Since the graphics (which are really neat if you haven't seen them ... two full clefs, with notes moving across them as the computer plays real four-part music!) are generated directly to screen locations, the display is still not right on the 8032, but the program appears to work properly. I did not actually move the sound board to the 8032, but see no reason for that not to work if the program is running.

In response to Ralph's request for info in Data Base Managers, I have used the JINSAM 8.0 system ($795), and endorse it wholeheartedly. One of its strongest features is the ability to translate files to formats that will be acceptea by VisiCalc or WordPro. In time, the real quality software houses will provide more cross-support to other popular packages ... Jini Micro Systems is to be commended for being in the lead in providing this bonus value. I've also used Commodore's OZZ, which has a lot of flexibility and power. Unfortunately, both code and data files have been made inaccessible to users, which gives no way to work around the package's shortcomings (limitation in number of calculations, inability to alter entries in prime sort key, limitations in flexibility of sorts). It's still a good general purpose package at $395.

My PET Peeve

Commodore has many good machines, and overall has a good record on design decisions. The worst mistake was the failure to use real ASCII. I believe that was intended to tie customers to Commodore's line (?) of printers. Obviously, with the rest of the world producing ASCII printers, it should not have been too much of a surprise that many converters appeared on the market ... and they didn't even capture the market they were looking for.

The next worst mistake was putting the formatting commands in the printer instead of the computer. I think the motivation was the same for that decision. There are not many features of the TRS-80 family that I like better than the Commodore's, but output formatting is one of them. It would be so nice to use the same commands to get the same output on screen or printer!

Sorry, Ralph. It was a good article for those who got the CBM printer, but you stepped on some of my sore toes!

Machine Langauge Articles

The three machine language articles by Bressler were all excellent. As mentioned previously, his perspective and approach looks like it should answer most questions for novices. Anyone wanting more information should first get a copy of Lenvanthal's book "6502 Assembly Langauge Programming". If you're a collector, purist, or snob, you may also want to get the "6502 Programming Manual" by MOS Technology, but the Levanthal book is much more readable, and oriented toward teaching the language rather than just being a reference manual.

Please, when you include machine language in your articles, give us two listings. The first, as in Ralph's articles, is a commented assembly listing. This form is easiest to understand when you're trying to figure out what the routine does and how it works. Actually, this form presents all the information that's needed but a second kind of listing is helpful. This second form, shown in Doug Haluza's article, is a printout in hexadecimal as it appears when you're using the monitor. The reason for showing it that way is that the monitor is the fastest way for readers to get the routine into their computers.

... By Any Other Name

COMAL sounds interesting, too! (Wish I had time to try all the new, neat, and interesting stuff I see showing up!!!) Old time PET users will recognize Len Lindsay's name from the original PET Gazette. Certainly the price of $47.50 is a bargain as far as computer language systems go. For those who object to paying for public domain software, try using any sophisticated package without any documentation. If the paper instructions are any good at all, they alone can be worth the price of the entire package. Since it looks like diskettes are provided with the package, too, the price is clearly not out of line. Of course, when you go to another language (even a better one), you leave the mainstream ... so be prepared to limit your horizons to a smaller group of programmers with whom to share software, hints, and condolences.

Program Indexing

Doug Haluza's program is certainly superior to cassette indexing routines done in BASIC. Here are two other alternatives, from my own bag of tricks:

1. Use C-10 cassettes, and record only one program on each side. Write the name of the program on the outside of the cassette. Find the right cassette, insert it with the name up, and the desired program will be the first one you come to. Just type LOAD and RUN.
2. Buy the new C2N cassette unit. It only costs $74.95, and has a tape counter.

There is a third alternative which I omitted because of cost, but eventually every serious user should aspire to move up to diskettes for storage. When you've filled all available drawers with cassettes (as I did), and have trouble finding the cassette you're looking for, that's a good hint you're ready.

Odds and Ends

My experience confirms the nice words about Jim Strasma ... he knows PETs, and is responsive.

Message for Jerry Key: using @ to replace files, according to what I've read, does not damage the file that's being saved ... the risk of damage is to other files on the same diskette. The danger is worse because you don't suspect the damage, and don't know where to look for it. If you do find it later, chances are that you won't connect the problem with the event that caused it.

Jerry's point about entering the monitor via calls (64785/54386) vs. breaks (1024) is well-taken. Follow his advice, and strengthen good habits. Someday it could save your program.

Command-O is really good. Don't know how I got by with a plain old ToolKit!

Notice that page 44 contains the commented assembly language listing for Doug's article (pp 26-28). A kudo for giving us both listings! Note: the monitor listing on page 28 has the first section on the right and the last on the left ... follow the addresses in the left column of each section

That's it for now (it's after 11 PM). It's been fun ... but I hope it's not so exciting the next time around!

Vic Sticks

It has been determined that the stock Atari joystick replacement part number CX-40 distributed by Atari Inc, Consumer Division, Sunnyvale, CA 94086 work just fine with the VIC-20. They can also be purchased for $9.96 from Toys R Us. I haven't figured out what controls it yet but I have four programs from Japan that work just fine with them - Racecars, Rugby, UFO, etc. - Ken Barroll

Bridge Hands

by Alex Breed

This article and accompanying program are a follow up to Bill Batcher's Card Shuffle (Sept/Oct 1981). It shuffles a standard deck of 52 playing cards, deals them into four hands as in contract bridge, sorts each hand by suit and denomination, and displays the hands as north, east, south and west.

The program uses a type of shuffle given by David Levy in his series of articles "Intelligent Computer Games" in Creative Computing. It is different from the three types described in Bill's article. Each card in the deck is taken in succession and exchanged with another card at random. It is simple, effective, and quick. In this program the cards are manipulated not as strings but as numerics, which are translated into suit notation at the time of display.

Some bridge players like to deal hands to themselves to practice bidding and play, and the program could be used for that purpose. It could also be used as the preliminary work for a real bridge-playing program.

```
1 REM BRIDGE HANDS BY ALEX BREED 9/16/81
99 REM ASSIGN VARIABLES ETC.
100 DIM A(52), B(4,13): R=RND(-TI)
110 DATA "(shift X)(shift Z)(shift S)(shift A)","11121314"
115 DATA "JQKA",4,"NORTH",0,"WEST",10,"EAST",4,"SOUTH"
120 READ S$: FOR I=1 TO 4: S$(I)=MID$(S$,I,1): NEXT
130 READ F$: FOR I=1 TO 7 STEP 2: F$((I+1)/2)=MID$(F$,I,2): NEXT
140 READ F1$: FOR I=1 TO 4: F1$(I)=MID$(F1$,I,1): NEXT
150 FOR I=1 TO 4: READ P(I),P$(I): NEXT
199 REM CREATE NUMERIC CARD DECK
200 FOR I=100 TO 400 STEP 100: FOR J=2 TO 14: N=N+1: A(N)=I+J: NEXT J,I
299 REM SHUFFLE
300 GOSUB 1000: PRINT TAB(14)"(6 down)SHUFFLING"
310 FOR I=1 TO 52: R=52*RND(1)+1: H=A(I): A(I)=A(R): A(R)=H: NEXT
399 REM DEAL AND BUBBLE SORT HANDS
400 GOSUB 1000: PRINT TAB(9)"(6 down)DEALING AND SORTING"
410 N=0: FOR I=1 TO 4: FOR J=1 TO 13
430 N=N+1: B(I,J)=A(N): NEXT J: M=12
450 T=0: FOR J=1 TO M: C=B(I,J)
460 IF C<B(I,J+1) THEN B(I,J)=B(I,J+1): B(I,J+1)=C: T=1
470 NEXT J: IF T THEN M=M-1: GOTO 450
480 NEXT I
499 REM CONVERT NOTATION AND PRINT DEAL
500 GOSUB 1000: FOR I=1 TO 4: IF I<>1 THEN PRINT: PRINT
510 PRINT TAB(P(I))P$(I): J=1: S1=4: GOSUB 1100
520 IF J>13 THEN 590
530 C$=RIGHT$(STR$(B(I,J)),3): S=VAL(LEFT$(C$,1)): C$=RIGHT$(C$,2)
540 IF LEFT$(C$,1)="0" THEN C$=RIGHT$(C$,1)
550 FOR K=1 TO 4: IF C$=F$(K) THEN C$=F1$(K): GOTO 570
560 NEXT K
570 IF S<>S1 THEN S1=S1-1: PRINT: GOSUB 1100: GOTO 570
580 PRINT C$" ";:J=J+1: GOTO 520
590 IF S1>1 THEN S1=S1-1: PRINT: GOSUB 1100: GOTO 590
600 NEXT I
610 PRINT "(home)(6 down)"TAB(33)"PRESS": PRINT TAB(32)"SPACE"
615 PRINT TAB(33) "FOR": PRINT TAB(33)"NEXT": PRINT TAB(33) "DEAL"
630 GET A$: IF A$="" THEN 630
640 GOTO 300
1000 PRINT "(clr)";: FOR I=1 TO 1000: NEXT: RETURN
1100 PRINT TAB(P(I))S$(S1)" ";:RETURN
```

Mindstretcher Series
A Product Review

Type: Software
Model PET: BASIC 1.0, 2.0 or 4.0
Source: Island Software
Box 300
Lake Grove, NY 11755
Price: 5 program sets at $20 each

by Ralph Bressler

Island Software is a new company which I'm sure is headed for many successes if their first series of programs is any indication of what they will produce. The Mindstretcher series is NOT remediation or drill in any particular subject. Instead, Bill Batcher has produced a series of programs which cause children, and for that matter adults, to think and solve problems. It is a package that I feeı no school should be without.

The Series contains 5 sets of programs with two to four programs on a tape. Each set costs $20 and comes complete with an explanation of the program, a description of how to use it with students, a history of the idea and a bibliography. All programs make use of some great PET graphics. Although the series is meant for "gifted and talented students in grades 3 through 9", many adults seem fascinated by the simple yet thought provoking concepts. Jerry Burke of Island Software says that one of their biggest problems is the unfamiliar names some of the programs have. So that you don't miss out, here is a brief summary of the programs.

Jigsaw

This tape contains a total of four programs with four jigsaw puzzles each. A picture using the PET graphics is displayed on the screen and is divided into 24 squares. The picture then disappears leaving only the numbered positions. The program then displays the pieces at random one at a time. The user chooses the position number for that piece and continues until all pieces are correctly positioned. I have seen people of all ages do these puzzles over and over. They are enjoyable and encourage the development of many different skills in the younger user.

Traffic Jam/Chain Reaction

In Traffic Jam, there are ten towns connected by one way roads. At the start of the game 4 cars are placed in certain towns and players alternate moving the cars. The object is to move the cars so that your opponent cannot make any move. This is done by moving a car into a town with only incoming roads. This program can be very challenging and requires a lot of thought about possible winning strategies. The suggested activities provided are excellent.

I have played Chain Reaction with several of my friends and usually can beat them. However, I generally fall prey to my eighth grade class since I, apparently, have not developed a good winning strategy. The game consists of a number of boxes each with an "explosive limit". On a turn a player may claim a box by placing a marker in one that is empty or add another marker to a box which already has some of his. When the "limit" is reached an explosion takes place sending particles in all directions. Where ever these particles land the player captures a square whether it was empty or occupied by an opponent. This may in turn cause more explosions, thus Chain Reaction. A player wins when the opponent loses all his markers. The board can vary from 3 by 3 to 6 by 6 thus making the game easy to learn but challenging once you know it.

Rubik/Candles

Rubik is a 2-dimensional representation of the 3-dimensional aggrevation that has swept the country. Six different graphics symbols are substituted for the six colors and rotation is limited to a clockwise direction. This doesn't seem to bother students in the least since they work with the program easily. Bill has included the option of entering Q to reset the cube.

Candles displays two cakes with a random number of candles on each. Players alternate blowing out candles trying to blow out the last one to win. On a turn a player may blow out any number of candles on one cake or the same number on both. The student can choose to play against the PET at four levels. At level 1 you can beat it since it chooses the best move only 60% of the time. Level 2 is harder since the safest move is chosen 80% of the time and at level 3 you can only win if you make no mistakes and get the first move. On Level 4 the PET always goes first and wins but this allows students to analyze the strategy used.

Black/Kayles

Black is not easy to explain but is a lot of fun to play. A board is filled with empty squares and players choose which one of three tiles they will place in the square. The tiles have different pathways on them so that when placed on the board they create a "road". After a player has placed his tile a heart begins moving from the upper left hand corner. If the heart goes off the board or ends up in the lower right hand corner at the end of its trip the player who placed the tile loses. A player must place the tile at the end of the path in each round. The board can be from 4 by 4 to 8 by 8 providing practice for the beginner and challenge for the seasoned pros.

Kayles is perhaps my favorite program since I figured out the strategy all by myself. A series of bowling pins is displayed in a row on the screen with a ball moving beneath them. A player throws the ball to knock down the pins by hitting the space bar. It is possible to knock down one or two pins with one ball and you try to make moves that will allow you to hit the last pin. You play against the PET which will always win if it goes first. After mastering the beginners level which has a solid row of pins you may move to the advanced level which has a gap at some point in the row making things more difficult.

Jinx/Welter

The interest caused by Jinx in a group of students is as hard to describe as the program itself. The user sets up a pattern on the Jinx board and then the PET moves the markers that have been placed according to a set of rules. When no markers can be moved the PET stops leaving a pattern behind. Points can be awarded for the number of "generations" an initial pattern lived or students may try for a certain final arrangement. The nature of the game means that people will experiment with it long after they know how it works.

Welter is as simple as Jinx is complex. It consists of a row of squares some of which have markers placed in them. Players take turns moving markers to the left until one player can no longer move and loses. The strategy is like Nim and the random placement of the markers makes things interesting even after you know how it works.

These programs are protected so that unauthorized copies cannot be made but this should not deter anyone from purchasing them. All the programs are done in such a way as to make them interesting, entertaining and educational. Even though most would be considered games, they teach analytical thought and planning as few other programs do. The price is more than fair and I highly recommend them to everyone.

THE MONITOR

by Jerry Key

This column will attempt to give relatively new users of the Machine Language Monitor a small understanding of what we can do with the MLM even if we won't normally get into machine language programming. The column will normally deal with Basic 4.0 but every attempt will be made to show the equivalent Basic 2.0 (03 ROM) commands or values in parentheses.

What is the MLM ? It is nothing more than a reflection of the code that is used by the PET, the code that is its real native language. If you are looking at a Basic program by way of the monitor you are seeing the program in Hexidecimal form. In Basic you use the Decimal form. Therefore it helps to have a conversion program to compare the difference. The PET/CBM Personal Computer Guide by Adam Osborne is an excellent reference for this but the locations in normal Read/Write Memory (RAM) are not converted except for start and end. Most user groups have a program available. Let's type in the following Basic program first:

```
10 REM WE WILL PUT THIS LITTLE
20 REM TEST PROGRAM IN JUST TO
30 REM TRY SEEING WHERE SOME OF
40 REM THE POINTERS ARE BY
50 REM USING THE MONITOR
```

Now let's look at the Monitor and see some things that are there. In direct mode type.: SYS54386 (SYS64785 for Basic 2.0) and the return key(rtn). The screen should look something like this:

```
*C
     PC   IRQ   SR AC XR YR SP
.;  B780 E455 34 33 38 36 FA
.
```

Basic 2.0 will have different numbers. For now we won't worry about what these mean except the '.' is where the input starts. Now type M 0028 0035(rtn). Your screen should now look like this:

```
     PC   IRQ   SR AC XR YR SP
.;  B780 E455 34 33 38 36 FA
.M 0028 0035
.:  0028 01 04 90 04 90 04 90 04
.:  0030 00 80 FE 7F 00 80 00 FF
.
```

What we have done is to display a number of locations that the PET uses to keep track of things. First off, in Hexidecimal, locations are counted from 0 to 9 then A to F then 10 to 1F, 20 to 2F etc. So in our example starting at the first TWO numbers after .: 0028, we count the 01 as location 0028, the 04 as 0029, the 90 as 002A, the next 04 as 002B and the last 04 on that line as 002F. Now look at the first two again, the 01 04. 0028 and 0029 is the pointer to the start of Basic stored low half first. We reverse these numbers to get 0401 (0400). This is the start of our Basic program. The next two numbers at 002A and 002B are 90 04 or 0490 which is the pointer to the start of variables which is also the first location PAST the end of our Basic program. The next four locations in our example are repeats of the 0490. They represent the end of variables and end of arrays respectivly. They point to the same location because our example program has not defined any variables or arrays yet. The 00 80 at 0030 and 0031 is the pointer to the start of strings. It moves from the top of memory down because that is how strings are stored. In our example this is the top of RAM in a 32K machine. The next two, FE 7F, or 7FFE, is the bottom of strings or top of available RAM memory, and the next 00 80=8000 is the limit of memory (for Basic). In our example, these are all essentially the top of RAM since no strings have been defined yet. We will talk about most of these in a later column.

So what is here that we can use? Let's go back to 0028-0029 and 002A-002B. These were 0401 (01 04) and 0490 (90 04). Now type M 0401 0490(rtn). This should get you a screen full of code that starts something like this:

```
.:  0401 1F 04 0A 00 8F 20 57 45
.:  0409 20 57 49 4C 4C 20 50 55
```

And ends like this:

```
.:  0481 20 54 48 45 20 4D 4F 4E
.:  0489 49 54 4F 52 00 00 00 AA
.
```

For now, we will deal only with the last line. Count starting with the 49 which is at location 0489, we have 0489,048A,048B,8C,8D,8E,8F and 0490 displayed. The AA in 0490 is left from when the PET did a memory test on power up. 0490 is the first location PAST the end of our Basic program. A BASIC program is always terminated with the 00 00 00 sequence. To check, let's decode the Hex numbers just before the three 00's into ASCII. The 52 is an R, 4F is O, 54 is T, 49 is I and starting at the end of our previous line, the 4E is N, 4F is O, 4D is M. If we put these in the correct order they should spell MONITOR which is the last word in our sample program!

So what is here that we might use right away? Have you ever tried loading a machine language program and then saving it to find that you really didn't have it saved? Well now you probably can. First load it, SYS54386 (SYS64785), type M 0028 0035, take the first two numbers as the starting location and the second two numbers as the ending location then type: .S "0:Filename",08,0401,0490(rtn) and see what happens (if you are using tape omit the 0: and change the 08 to 01). What you have told the PET is to Save(S) Drive# and filename on Device number 08 (disk) starting at location 0401 and ending with location 048F (0490-1). Always use commas as delimitors between parameters.

Just bear in mind that the 00 00 00 will not be present if the end of the program is in machine language or has machine language with it. But who has trouble saving a straight Basic program anyway? Play around with this and next time we will use this to save a straight machine language program. Have fun as the worst you can do is hang the machine and have to reset it!

If you want to see this column continued, send comments and suggestions to: Jerry Key, 141 Flint Ridge Dr, Gahanna, OH 43230 or to The PAPER.

Ken's Komputer Corner

by Ken Barroll

Hi! My name is Ken Barroll and I reside in Washington state. When I'm not flying a 747, I am working (playing?) with my computer. Maybe I should say computers. My home line consists of a) the CBM 8032 with an 8050 disk drive and the Olympia ES-100 printer which I have had for about a year and use mostly as a word processor with WordPro 4; b) the PET 2001 with 8K and the 'munchken' keyboard and a 2022 printer; and c) the VIC 20 and VIC 1211M. I also have a VIC 1001 in the Orient which is quite similar to the USA version (the VIC 20). Yes, my wife does think that anyone who owns four computers is nuts!

The CBM 8032 gets used almost daily and does a great job as a word processor and general business machine to print labels, customer lists, etc. This column is being typed with it and WordPro 4. I have the Dataface GRQ 10 card plus the

TNW 1000 CBM-IEEE to RS.232 unit in a stock Olympia ES-100. The latter is a nifty electronic DAISYWHEEL typewriter from Germany that costs about $1600 hooked up and running. If you know of any other daisywheel, letter quality RO printer that even comes close in price, let me know. This one is built with the same standards of quality and precision we expect from the land of Porsche and Mercedes! It is a popular solution in Europe for those who can't afford NEC's or inkjets. It's fairly unknow here but with 1700 dealers and several companies now making interfaces I wager it won't be for long. Just as a stand alone self correcting typewriter it's out of sight!!

I purchased the PET 2001 in 1978. It was my introduction to computers and has been a total joy. One of its prime purposes is to be a Morse code/radioteletype terminal for my ham radio station (W7OP - Radio Free Seattle) and, of course, for all the games. Both of my kids were taught to read and write on it and it was used by me to learn BASIC programming. This machine is getting well known and I think I must have about 300 programs for it. Dick Ball has over 100 programs. Around here the favorite game is 'CAPTURE!' from Cursor and their cartoons are always prime entertainment. Anyone who owns a PET and doesn't subscribe to Cursor is missing out on something great. The 'in-house' record with CAPTURE! is four out of ten with a difficulty of 9!!

The VIC, which is primarily a 4.0 version 2 ROM machine, is at first glance a cheap entry level computer. This impression is quite deceiving since it has a very advanced brain which can be expanded in many ways we haven't even seen yet. Many accessories are now available in the mystic Orient but the single 170K disc drive and the modem have yet to be seen. Of the Commodore equipment I own it has the MOST FEATURES, some of which I don't understand even now. I do have some 40 programs and am finding new ones almost daily. Several of the big software houses in Japan have already released about 50 programs for VIC, mostly games. ADCOM and Systems Formulate really have some good stuff - like PACMAN and Racecars just like the arcades! Some of the programs require the use of a joystick and/or larger memory units. These are already available in department stores in Japan.

The VIC 20/1001 is where most of my time is spent these days. (Half of every month I'm overseas.) It doesn't work well with all color TV's but is quite satisfactory with the Maganovox 19" we have in the basement. So, what's all the excitement about? It's just a color PET that works off the TV, right? WRONG!! It is quite a different bug with features not even the 8032 can compete with.

The VIC consists of the keyboard unit (about $300) with a seperate power supply that generates the 11 volts AC the VIC uses. You also get an RF converter but the VIC exhibits brilliant performance when fed into a Zenith color monitor, otherwise your results will be excellent to poor, depending on the TV selected. There doesn't seem to be any relationship between the quality of the TV and how it will work with the VIC. The black stripe crystal locked TV's including some SONY's seem to work poorly with VIC but are outstanding TV's by themselves.

The colors are controlled by POKing a number between 1 and 255 into 36879. There is a border color, one for the background and the characters are also in color. On power up you can PEEK at 36879 and find 27 which corresponds to blue border, white background and very dark blue characters. If you want to put a red heart somewhere, you can either use a print statement with the appropriate control character to color it or else you can POKE it onto the screen. To POKE you give the position and the number for the character AND the color position for that spot and the color you want. Screen memory is in one range and screen color memory in another. It may sound hard but is quite easy once you get used to it. For those of us who couldn't remember the old imbedded cursor controls, we now have a whole new set of color controls.

VIC has three 'voices' and one 'noise' channel which play over the TV speaker. POKEs control the sound including the volume of each voice. The VIC can read tapes from the PET and list them on the screen. If they don't use a lot of POKEs to the screen or print too wide, they will run just fine. To load VIC tapes on a PET type POKE 41,16:NEW first, or use APPEND with the toolkit. The VIC has a built-in relocator which allows it to handle both kinds of tapes. Most of the BASIC commands are the same as the PET. A few 'mistakes' in PET BASIC have been corrected. Hitting RETURN without any input will not drop you out of a program since the VIC just goes onto the next line. To stop you must use STOP. If you hold down the stop key and press RESTORE everything is reset except the program in memory. Shifting and pressing the Commodore logo key toggles the character set from uppercase/graphics which is how it powers up to upper/lower case. There are four function keys and shifting makes 8. These keys will be used in some upcoming applications. A joystick or dual paddle interface works great with games like Racecar. There will soon be an Alien and a Space Invader cartridge of top quality which will make use of the plug-in system.

The back of the VIC has a big hole in it for one of the 13 expansion units or the plug for the expansion unit interface holder, for lack of a better phrase. The latter is the size of a tissue box and can take up to 6 units at once. You pick exactly which units you want. With everything plugged in you can have a complete 32K computer that has a resident Toolkit and Hires color graphics. This package includes extra commands such as 'circle' which draws one perfectly and 'paint' which colors areas of the screen only. The VIC printer is a special version of the GP-80M from Seikosha and is a well-known and reliable unit. It costs about $300 in the Orient and their prices are roughly comparable to ours.

The VIC 1010, the aforementioned interface, holds the six cartridges and costs about $150 over there. Many of these plug-in modules are already available in Japan. Three of these units are straight RAM expansion which are useful since the VIC powers up with only 3583 bytes free. Actually this is enough to do alot but more memory is nice. The VIC 1210 is 3K of RAM so you end up with 6655 bytes when you plug it in. It costs $43. The VIC 1110 8K unit gives you 11775 and costs $73 while the 16K VIC 1111 gives you 19967 and is $108. So we can have a 20K ROM hires color machine that can be expanded to 32K RAM and uses a 6502A running at 1.0227 MHz. The TV screen will have 22 characters across (BIG ones) and 23 down but in hires mode it is 176x160 dots that can be individually controlled. You should see some of the games from the Orient like PAC-MAN with a display very close to the arcade version using the VIC's buildable character set. Exactly how that works I haven't figured out yet, but see the July COMPUTE. If you key in the program called 'Rocket Command' that is listed in the back you will see small space ships and smile buttons flying across the screen. These are all made up using the buildable character set. 'Rocket Command' is clever and addicting and our record is 90 points.

Another nice unit is the VIC 1212 which costs about $64 and is a small 'toolkit'. It contains the commands: RENUMBER, MERGE, FIND, CHANGE, DELETE, AUTO, TRACE, STEP, DUMP and HELP. The VIC 1540 is the disk unit not yet shown but said to hold 170K.

The VIC 1112 makes an IEEE 488 port for the normal PET accessories such as the 8050 one megabyte disk drive and the 2022/4022 printers. There is a serial port on back of the keyboard. The VIC 1530 is the old cassette recorder but in white and gold with brown keys and a tape counter! It costs about $80. That's what's available now for the VIC and when you add it all up you could easily have a 32K color computer with advanced commands AND printer for under $1300.

In summary, the VIC is a very capable computer at a tremendously low price. Its only major drawback is the 22 character line. Rumor says that one outfit is looking at a word processor package that will convert it to a 66 to 88 column screen. Several companies have made special cards for other computers like the Apple that make the screen 80 columns wide. The VIC has many features in common with the new color PET too. So, it shouldn't be too difficult to make a serious business computer of the VIC, at least on a part time basis. Anyhow, it's absolutely marvelous and a wonderful machine that deserves a lot of attention! I fully expect it to be "KING" in a very short time providing Commodore doesn't botch the marketing.

Mikro
A Product Review

by Ralph Bressler

Type: Firmware
Model PET: BASIC 1.0, 2.0 or 4.0
Source: Skyles Electric Works
231E South Whisman Rd
Mountain View, CA 94041
Price: $80

Bob Skyles is slowly gathering the best firmware for the PET available anywhere. First, there was Toolkit, then Disk-O-Pro and Command-O and now Mikro. Mikro is a 4K ROM which houses a symbolic assembler (imported from England) that lets you type assembly source code just as you would type in a BASIC program. Just one SYS command and you're ready to go. The chip comes with the usual excellent Skyles documentation. It runs to a total of 49 pages, not the 12 pages mentioned in some advertisements.

All you do is plug in the chip as explained in the manual and then type SYS 40960 and you're ready to go. Mikro allows you to type in assembly source code using the PET's own screen editor. This means you don't need to remember any odd syntax or new commands to enter or change your program. The numbers you type are used for references. In fact, Mikro will work with Toolkit or Command-O so that you can RENUMBER a program, DELETE sections or FIND specific op codes.

Mikro adds three new commands to BASIC which are preceeded by an "!". !F will format your assembly listing in the proper LABEL, OP CODE, OPERAND and COMMENT fields. The normal PET LIST shows everything packed together, so the source code is hard to read. When you want to assemble your source code you simply type !A and Mikro does the assembly. I really didn't try a VERY long program but I felt it assembled very fast. You may choose to have a printout sent to the screen or you may skip this step. The listing includes line number, location, object code and formatted source code plus comments. Object code is automatically stored where you told Mikro to put it. No loader is necessary. The !N command will convert any number to binary, octal, decimal or hex when you specify the original base. This is very handy if you've ever had to do these calculations on the fly.

The manual is very comprehensive and not only outlines the operation of Mikro but gives hints for its use and a few cautions about some bugs in Mikro. The manual also gives a nice, short tutorial on 6502 programming, as well, as several mnemonic and op code charts. The manual also explains the 12 error messages that Mikro has and details their meaning.

Mikro uses a standard set of operators such as * for program origin and ; for comments. Plus and minus signs may be used in expressions. Since Mikro is a symbolic assembler you may also set up labels for commonly used addresses. It also has 5 pseudo-ops. The "=" sign is used to define labels and set the program's origin. TXT allows messages to be included as a string of bytes coded in ASCII. BYT allows the insertion of 8-bit data bytes into object code. These may be of almost any form including decimal, hex, ASCII characters, binary and expressions. WOR is the pseudoo-op that allows you to insert tables of 16-bit addresses into the object code. END is optional but is handy since it will stop the assembly when Mikro finds it. This allows you to assemble up to the point where you think a mistake may be occuring. It also means you can add a BASIC program at the end of your source and the manual explains how to do this.

Mikro has some weaknesses which Skyles points out. It will not do conditional assembly or handle macros. These two abilities are nice but not mandatory. There is another problem that I didn't find in the manual. Any BASIC keywords used in labels will get tokenized and will be unusable. This may not seem like a big problem and it probably isn't in many cases. However, when trying to use standard labels, like CHR<u>GET</u> or CHR<u>GOT</u>, this can be bothersome. Some may consider the $80 price tag a bit steep also.

Mikro's strongest point is the fact that it is always only a SYS away. It also uses the PET's screen editor which means no memorizing editor commands. It is fast, easy to use, and comes with a great manual. Really, $80 is not a lot for the capabilities included in the chip and its versatility. I highly recommend it to anyone who needs to do a lot of short assembly programming quickly and easily.

```
 10  033A             ; SCREEN FILL ROUTINE
 20  033A             ;
 30  8000             SCREEN=32768
 40  0001             CHR='A-64
 50  033A             ;
 60  033A  A901                   LDA  #CHR
 70  033C  A200                   LDX  #0
 80  033E  9D0080     LOOP        STA  SCREEN,X
 90  0341  9D0081                 STA  SCREEN+256,X
100  0344  9D0082                 STA  SCREEN+512,X
110  0347  9D0083                 STA  SCREEN+768,X
120  034A  E8                     INX
130  034B  D0F1                   BNE  LOOP
140  034D  60                     RTS
150  034E                         END
```

This example of an assembly listing from Mikro gives you some idea of how nice it is to use any assembler. Using one saves so much work, allows so many shortcuts and preserves your program so it can be easily changed. Lines 50 and 60 show how a label can be equated to almost any value. SCREEN is given the decimal value 32768 while CHR becomes the ASCII value of A minus 64 to get the screen code for A. As it assembles MIKRO places the object code in the listing as shown in lines 60 to 140. Notice that in lines 90 to 110 we are storing the accumulator on the screen but that we can use labels like SCREEN+256 which is very handy. We can easily change parts of the program to assemble the program at a totally different address or fill the screen with a new character.

New Software Sources

Everybody wants a piece of the action. Large book publishers and small audiovisual firms are beginning to see the microcomputer software market as just right for them. The end result of the assault of these companies is still unclear. Since large companies and even smaller ones that sell products other than software, can advertise extensively their products will probably sell quite well. Many schools will buy from a company they have dealt with before for other materials rather than consider a small, exclusively computer software company. The big question here is what the effect may be on the overall quality of the software available.

I have seen some of the programs being offered by FOCUS, Sunburst and EME and think that some of it is really good. Some is OK but has been done before and some is trash. I have not seen much software from large publishers yet except the math series from Millikan. Believe me it is just a matter of time. I believe that purchasers of software must make it clear to the vendors that they will demand quality. A good company name or high priced advertising should not be enough to make us buy poor or only good software. These companies have the resources if not the experience to produce excellent, comprehensive packages. Let us make sure we educate them - Publisher

Machine Language for the Near Beginner: Part 3

by R Bressler

Last session we concentrated almost entirely on branch instructions. Branch instructions use relative addressing which tells the PET how many memory locations to skip ahead or behind. These instructions test the various flags which are found in the status register. The status register is a one byte register which must be converted to binary to be understood. We will move a little faster in this installment as we set up and use program loops and introduce a new addressing mode.

Before we go on let me make a point. Some of these programs that follow may be interesting if they are just typed in and executed. To really learn what is happening you should study the text. Try to understand the logic of the program and write the machine language translation yourself. I have provided this translation in every case so that you may CHECK yourself. Notice that in this installment I have given the translations in "hex dump" form. This is done to save space and because it is easier to type into the monitor this way. You should also try to make any suggested changes.

Setting up simple program loops is not hard. The most difficult thing is to calculate the backward branch necessary to set up the loop. Let's write a program which will calculate the sum of the hex numbers from 1 to F. To do this we need a counter. We can use the X register and set it to F. Each time through the loop we can decrement X until it gets to zero and then leave the loop. The assembly program would look like this:

```
Label     Mnemonic      Operand     Comments
=====     ========      =======     ========
          CLC                       * clear the carry for addition
          LDA           #$00        * clear the accumulator
          LDX           #$0F        * set the counter to F
 LOOP     STX           LOC1        * store XR at LOC1
          ADC           LOC1        * add LOC1 to accumulator
          DEX                       * subtract one from XR
          BNE           LOOP        * go back to LOOP if XR not 0
          STA           LOC2        * if at end store XR at LOC2
          BRK                       * return to monitor
```

We have used LDX to set our X register counter to $0F. We then have stored the current value of the X register at LOC1 by using STX. The DEX instruction simply subtracts 1 from our X register counter. BNE means BRANCH NOT EQUAL which means the branch WILL occur if the previous operation DID NOT result in zero. In this case the branch is to the location with the label loop. When we hand assemble this program we will have to calculate the backward branch from the BNE instruction to LOOP. Now let's do the assembly:

```
0398 18 A9 00 A2 0F 8E 90 03
03A0 6D 90 03 CA D0 F7 8D 91
03A8 00
```

Notice that the offset for the BNE at $03A5 was calculated to be F7. We want to branch back to $039D if the counter is not zero. There are many ways to calculate this backward branch. I find the easiest way is as follows:

1) Call the address immediately after the offset 0.
2) Count backwards by one for every memory location. So in this case we count FF, FE, FD, FC, FB, FA, F9, F8, and F7.
3) Enter this number as the offset after the branch instruction.

Most of us have seen a program that instantly fills the screen with some character. This, of course, is in machine language and is not to hard to do. All you have to do is load the accumulator with the code for your favorite character and then store it at EVERY location on the screen, all 1000 of them. It takes two bytes to load the accumulator and three bytes to store it at each location. Add one more for the BReaK at the end and you have a 3003 byte program. Of course, there is a much easier way. What we really want to do is store the character at the first screen location and then move over by ONE and store it again. We will set up a loop using the X register as a counter and we will execute the loop 255 times. But I thought the screen had 1000 locations? Well, we will divide the screen into 4 quarters each with 256 characters. But that's 1024 locations! Who cares when it makes things easier. To do this magical trick we use INDEXED ADDRESSING. In this case we will use STA $8000,X to put the characters on the screen. This means we store the accumulator at $8000 PLUS whatever is in the X register. If we continue to increment the register the screen will fill. The program then is:

```
Label    Mnemonic  Operand      Comments
=====    ========  =======      ========
         LDA       $A0          * load accumulator with decimal 160
         LDX       #$00         * zero the X register for a counter
LOOP     STA       $8000,X      * store accumulator in 1st quarter screen
         STA       $8100,X      * store accumulator in 2nd quarter screen
         STA       $8200,X      * store accumulator in 3rd quarter screen
         STA       $8300,X      * store accumulator in 4th quarter screen
         INX                    * increment X register by one
         BNE       LOOP         * do it again until back to 0
         BRK                    * back to monitor
```

Notice that the 'quarter' screens are at $8000, $8100, $8200 and $8300. Remember that BNE means Branch Not Equal to zero. During the first iteration of the loop X is 1, 2 during the second and so on until it gets to FF on the 255th. One more increment and we're back to zero since $FF + 1 is $100 but only the 00 fits in the one byte that is the X register. The translation of this would be:

```
0398 A9 A0 A2 00 8D 00 80 8D
03A0 00 81 8D 00 82 8D 00 83
03A8 E8 D0 F1 00
```

As you can see indexed addressing makes some operations practical which might be impossible without it. With this type of addressing a whole range of memory locations can be accessed with just one command. Remember than the routine above could be used in a BASIC program simply by changing the BRK to an RTS (60 instead of 00). In the BASIC program you could then simply use SYS 920 to fill the screen with the character of your choice. To change the 'fill character' from the BASIC program just use POKE 921,X where X is the value of the character you want to fill the screen with.

Another nice effect to call from a BASIC program is the reversal of the characters on the screen. Reverse characters turn to normal field while those in normal field are reversed. A nice effect when a student gets a right answer. This is relatively easy to do using the indexed addressing we just spoke about. The reverse characters are simply the normal characters with the 7th bit set to 1 instead of 0. For example A is $01 in the PET screen code while reverse A is $81. When we convert to binary we have:

```
normal  A   0000 0001 = $01
reverse A   1000 0001 = $81
```

The only difference is the "set" seventh bit. This means that all we need to change to reverse a character is this bit. When its 1 we should change it to 0 and vice versa. To accomplish this we will use Exclusive OR which has the mnemonic EOR. We will "exclusive or" the contents of each screen location with $80 and then store the result back at the same location. Exclusive or is a logical operation which says that 1 and 1 result in 0, 0 and 0 results in 0 but 1 and 0 gives a 1. For an example let's say that the location contained an A and we "exclusive or" it with $80. The result would look like this:

```
  A is   0000 0001 = $01
$80 is   1000 0000 = $80
result   1000 0001 = $81
```

It doesn't take a very careful look to realize that the result is a reverse A. Now we can write our program which will reverse all the characters in the first 255 screen locations.

```
Label    Mnemonic  Operand    Comment
=====    ========  =======    =======
         LDX       #$00       ; load X with 0 to clear counter
LOOP     LDA       $8000,X    ; load A from location plus X
         EOR       #$80       ; exclusive or contents to do reversal
         STA       $8000,X    ; store result of EOR back on screen
         INX                  ; increment counter by one
         BNE       LOOP       ; do it again until X is 0
         RTS                  ; when through return to BASIC
```

Notice that we have used RTS instead of BRK so that we can call the routine from BASIC using SYS 920. Can you make the changes necessary to do the entire screen? Here is the machine language translation:

```
0398 A2 00 BD 00 80 49 80 9D
03A0 00 80 E8 D0 F5 60
```

A few simple changes to this program would allow us to, say, reverse all characters except the spaces. Basically, all we have to do is add a way to check to see if we have a space. If we don't have a space we then skip over the EOR. To do this we use the CoMPare mnemonic we encountered before. Remember the code for a space is 32 or $20 in hex. Here is the modified program:

```
Label    Mnemonic  Operand    Comment
=====    ========  =======    =======
         LDX       #$00       ; load X with 0 to clear counter
LOOP     LDA       $8000,X    ; load A from location plus X
         CMP       #$20       ; check to see if it's a space
         BEQ       OVER       ; if it is skip the reversal
         EOR       #$80       ; exclusive or contents to do reversal
         STA       $8000,X    ; store result of EOR back on screen
OVER     INX                  ; increment counter by one
         BNE       LOOP       ; do it again until X is 0
         RTS                  ; when through return to BASIC
```

This leads to other interesting possibilities. Can you alter things so that the changes occur in the last quarter of the screen? Could you change ONE byte to change only the spaces? Here is the translation:

```
0398 A2 00 BD 00 80 C9 20 F0
03A0 02 A9 80 9D 00 80 E8 D0
03A8 F1 60
```

We could now write a program to change any character to any other. The program below will change all spaces to stars.

```
Label     Mnemonic  Operand    Comment
=====     ========  =======    =======
          LDX       #$00       ; load X with 0 to clear counter
LOOP      LDA       $8000,X    ; load A from location plus X
          CMP       #$20       ; check if it's character to change
          BNE       OVER       ; if it is skip the reversal
          LDA       #$2A       ; change it to the character you want
          STA       $8000,X    ; store result of EOR back on screen
OVER      INX                  ; increment counter by one
          BNE       LOOP       ; do it again until X is 0
          RTS                  ; when through return to BASIC
```

Here is the translation to machine language:

```
0398 A2 00 BD 00 80 C9 20 D0
03A0 05 A9 2A 8D 00 80 E8 D0
03A8 F1 60
```

Why not try changing the program above so that it will change all the O's on the screen to 0's? The limitation of this method is that we can only change one character. What if we wanted to change a group of characters? How could we change all vowels to stars? To do this we would just have to do a series of comparisons. After each comparison for a vowel we would branch to the part of the program which would do the change only if we found a match. The program would then look like this:

```
Label     Mnemonic  Operand    Comment
=====     ========  =======    =======
          LDX       #$00       ; set counter to 0
LOOP      LDA       $8000,X    ; load A from screen location plus X
          CMP       #$01       ; compare to screen code for A
          BEQ       OVER       ; if found branch to change
          CMP       #$05       ; compare to screen code for E
          BEQ       OVER       ; if found branch to change
          CMP       #$09       ; compare to screen code for I
          BEQ       OVER       ; if found branch to change
          CMP       #$0F       ; compare to screen code for O
          BEQ       OVER       ; if found branch to change
          CMP       #$15       ; compare to screen code for U
          BEQ       OVER       ; if found branch to change
          JMP       SKIP       ; jump over change if no match made
OVER      LDA       #$2A       ; load A with code for a *
SKIP      STA       $8000,X    ; store A at a screen location plus X
          INX                  ; increment counter by one
          BNE       LOOP       ; do it again until X is 0
          RTS                  ; return to BASIC
```

Here is the translation to machine language:

```
0398 A2 00 BD 00 80 C9 01 F0
03A0 13 C9 05 F0 0F C9 09 F0
```

```
03A8 0B C9 0F F0 07 C9 15 F0
03B0 03 4C 03 B6 A9 2A 9D 00
03B8 80 E8 D0 DE 60
```

Well there it is. It should be obvious that the larger the number of comparisons we want to make the longer the program. There should be an easier way and there is. We have used indexed addressing to access the screen locations but we can also use it to access the elements of a data table. A table in this case is simply a set of values stored in consecutive memory locations. Here the table would be the vowels or other characters we wish to substitute the * for. We will continue to use X as the index to the screen but now we will use Y as the index to the data table. We will store our program beginning at $0398 but our data will be at $0393 to $0397. To enter the data simply use the monitor to change the contents of these locations to the hex values for the codes for the vowels; $01, $05, $09, $0F and $15. Here is the program:

```
Label    Mnemonic  Operand    Comment
=====    ========  =======    =======
         LDX       #$00       ; set counter to 0
LOOP     LDA       $8000,X    ; load A from screen location plus X
         LDY       #$00       ; zero pointer to data table
CHECK    CMP       $0393,Y    ; point to element in table based on Y
         BEQ       DUN        ; if a match do the change to *
         INY                  ; if no change increment Y and ...
         CPY       #$05       ; do it again if we have not done...
         BNE       CHECK      ; all 5 vowels
         JMP       OVER       ; if no match at all skip the change
DUN      LDA       #$2A       ; load A with code for *
OVER     STA       $8000,X    ; store A at screen location plus X
         INX                  ; increment X by one
         BNE       LOOP       ; and do again until X is 0
         RTS                  ; return to BASIC
```

As you can see this makes things neater but there is also an extra bonus. Now if we want to check for more characters and make changes the program only gets longer by one byte. All we have to do is add an element to the data table. In the previous methods each new comparison would add 4 bytes.This does require some advanced planning since we must leave enough room for our data table. Here is the translation which you should be able to do by now:

```
0393 01 05 09 0F 15 A2 00 BD
039B 00 80 A0 00 D9 93 03 F0
03A3 08 C8 C0 05 D0 F6 4C AE
03AB 03 A9 2A 9D 00 80 E8 D0
03B3 E6 60
```

This session has run longer than I thought it might but I hope it was useful. I know I keep promising subroutines and next time they will be included along with a strange new addressing mode called INDIRECT addressing. At this point I am nearing the limit of what I know and have used myself. You will see more machine language articles by Doug Haluza and Jim Fowler in upcoming issues. Many of us learned BASIC by looking at the programs others had done. Machine language is no different. Examining well commented programs helps us learn new techniques and gain insight into old ones.

Please make this a dialogue rather than a lecture. Those of you that are more experienced should criticize what I do since I freely admit I am no seasoned pro. Less experienced readers should pull me up short on points that are poorly explained. In any case, please let me know how I am doing.

VIGIL
A Product Review

Type: Software
Model PET: BASIC 1.0, 2.0 or 4.0
Source: Abacus Software
PO Box 7211
Grand Rapids, MI 49510
Price: $35

by Bill Batcher

I teach programming to 5th and 6th grade students enrolled in a gifted program. For ten weeks, each pupil learns the various commands and statements in BASIC and the rudiments of programming. Those students who are highly motivated can opt to sign up for a second ten week course in programming. Here, each pupil proposes a program he plans to write, and then spends each session solving problems he encounters as the program takes shape.

A few of these advanced students have proposed word puzzles, math drills, card games and other such programs wherein print statements are used to display all the game elements on the screen. Things sit quietly and patiently waiting for responses to input statements. But 90% of these kids invariably propose to write the triple A type programs (Atari-Aliens-Asteriods). In these programs lots of things are moving on the screen, colliding with each other and racking up scores. Thus, I have to bring these kids into the whole realm of screen poking and get statements keyed to the numeric pad. They also are never quite satisfied that the program is complete until it has sound.

When Ralph first showed me his copy of VIGIL, it seemed to me that this could be a tool to use with my advanced kids. It looked like an ideal language for AAA programs. Numeric pad inputs are built into the program as is the scoreboard. It contains a cue for collisions and sound. But could these students cope with a second language? Would they be thrown by the non-English looking commands? Would it be easier to stick to BASIC?

With some hesitation I had my district buy a copy. I introduced it to a group of sixth graders in September who had had their introduction to BASIC the previous spring. Happily, I found the kids took to VIGIL very readily. They could handle its routines easily. And in the process, they are learning that there are different ways to accomplish a goal.

What is VIGIL? The acronym stands for Video Interactive Game Interpretive Language. This is descriptive. VIGIL is tailor-made for writing interactive video games. It is no good at all for data processing chores. It could be used for a random math drill, but BASIC does that more efficiently. But for AAA type games VIGIL is perfect.

The word language needs some qualification. You do need to load the VIGIL interpreter into memory before you can write a program. You do need to learn the unique vocabulary and syntax of the VIGIL language. But VIGIL is not an independent language in the same sense as FORTH or PASCAL or whatever. VIGIL uses the BASIC text editor that is built into the PET's ROM. It lets the BASIC interpreter begin to read each statement and then it steps in and reinterprets it. Thus, while a few statements are identical in BASIC and VIGIL (like GOTO, GOSUB, CLR), others are completely different. For instance, READ in VIGIL will assign a character from a depressed key to a register (something like a variable in BASIC).

The primary joy of VIGIL is its economy. Every statement in a program is very short. For example, compare the following routines for moving a ball on the screen in response to the numeric pad. Notice that the VIGIL version seems to be a puzzle with its unfamiliar statements. However, this is a new language and once you understand the commands the logic is quite simple.

```
    VIGIL              BASIC

3000 &2                3000 GET K$
3010 IV1               3010 IF K$="2" THEN V=V+1
3020 &8                3020 IF K$="4" THEN H=H-1
3030 IV255             3030 IF K$="6" THEN H=H+1
3040 &6                3040 IF K$="8" THEN V=V-1
3050 IH1               3050 P = 33268+H+(V*40)
3060 &4                3060 POKE P,81
3070 IH255
3080 DHV1
```

I found that my kids could understand the logic of the VIGIL routine as easily as the more "English-like" BASIC routine. We'd read a line like 3010 as "increase the vertical by one". We don't need to worry about multiplying by 40. Line 3030 shows the use of 255 as equal to -1. We did a little modular arithmetic and the kids could get the idea that 255 was -1 in mod 256. VIGIL does allow other ways to decrease a register so you don't have to go into mods.

Another joy of VIGIL is its use of an 80 by 50 screen. The ball in the above routine is a quarter square, and it has a lot of space to maneuver with 4000 screen locations. All the pictures in a VIGIL program are made of quarter squares, somewhat like Trash-80 pictures. If you want to use the excellent PET graphics, forget VIGIL and go back to PET BASIC. But with those quarter squares, my kids were able to design airplanes, cars, space creatures, trains, parachutes, etc. Once you have defined a picture in a VIGIL program there is no problem to get it to move. My kids design their pictures on graph paper because the pictures themselves do not show up when you list the program. After the picture is designed, the artist converts it into a series of two digit hexadecimal numbers. Sounds complicated? Actually, I copied a little chart from the guide and provided each kid with a copy, and with that they have no trouble finding the numbers for any picture they draw. A small by-product is that they are subliminally getting accustomed to hex notation.

One disappointment I've had is that I can't get VIGIL to work on a FAT 40. This is too bad since the sound is already built in and since there is a whole generation of users growing out there who only know this machine. Perhaps, ABACUS will be making another version available.

When you purchase VIGIL, you get three VIGIL interpreters, one for each version of BASIC, 9 illustrative game programs, and the guide. The latter is well written and, it seems to me, completely definitive.

One has to admire Roy Wainwright, the author, of VIGIL, especially after reading the biographical note in the guide's preface. I heartily recommend VIGIL as a teaching tool for programming classes at all levels.

VIGIL for the Fat 40

Bill's point about not being able to use VIGIL on the Fat 40 was well taken. It would be particularly nice to use VIGIL on this machine with its built in sound and larger display. Two major problems present themselves though. The $E000 ROM which controls parts of the screen display has been changed on the new boards so that any references to it by VIGIL would have to be altered. Also, VIGIL starts at $033A and extends to $1300 and so uses the taboo locations from $03E9 to $03F9. These are used by the Fat 40 and are constantly changed.

Somehow I just knew that the folks at ABACUS had at least seen the problem and must be working on it. A short call to Arnie Lee confirmed my suspicions. A version now exists for the Fat 40. You may upgrade an older version by sending back the origional tape. There is no fee involved and Arnie promised prompt return of the program. When a company supports their software in this manner they deserve our support. - Ralph

Techniques for Better BASIC Programs

by R Bressler

I feel that it is not the purpose of a newsletter like The PAPER to teach beginning BASIC. If you want to know how a PRINT statement works or what a FOR...NEXT loop is there are many books which you can use. However, once you know most or all the commands in BASIC there is still the matter of hooking them together into a good program. Technique is often not taught in basic BASIC books and this is what a newsletter should work on. Some of the most useful techniques for the PET are still not mentioned in Commodore's manuals or in many books. This article concentrates on a few of these techniques.

In general, the first requirement for most programs is that they be easy to read. The logic should be easy to follow and free from unnecessary tricks. One way to improve readabilty is to include an adequate number of REMarks. These can explain the function of a part of your program or the use of a certain variable. Another way to make things easier to follow is to make your program top down and structured. This simply means that there is a main program loop near the beginning of the program that calls various subroutines. The subroutines then return to the main program loop.

Other good programming techniques might be the inclusion of a menu in your program, intelligently named variables and the proper use of GET and INPUT with default values for the latter. A menu is simply a list of functions from which the user makes his selection. This is usually done with a GET statement followed by a series of GOSUBs. When deciding what letters to use for variables try to use a combination of two letters or a letter and a number that remind you of what values the variable will have. For example, LN$ for last names or TG for the total of a set of grades. PET BASIC will allow longer names but only the first two letters are recognized. Longer names also take up a lot of memory space so stick to the method I just mentioned. When a user needs to hit a key to continue or to make a yes or no choice a GET statement may be appropriate. On the PET hitting RETURN alone with no input will cause a program to terminate leaving the naive user scratching his head. Providing a default value will speed up data entry and prevent unwanted fallout.

Rather than give isolated examples of these techniques I have decided to present a program which will integrate many of these techniques into one unit. Let us suppose that we want to keep a small inventory of items in a school store. We could store the names of our items in several different ways. We could type everything into DATA statements. This would avoid using data files but would mean we would have to be sure that the students using the program could change the data statements and make no mistakes. We could use a cassette tape data file. This is slow and somewhat unreliable since, again, students would have to be familiar with how the tape system operates. We could use disk data files but this is fraught with problems. One big problem is the initial cost of the disk drive. I would be nice if we could use INPUT statements to manage the data and then have the changes we make automatically entered into DATA statements. This, of course, has been explained in many articles. This first one I saw was in the PET Gazette and the latest in COMPUTE. I will try to explain it again here with a practical example.

We will develop this program step by step, rather than looking at the finished product and trying to decipher it. I have a finshed program that I know works and will use the line numbers from it. Remember that a line like

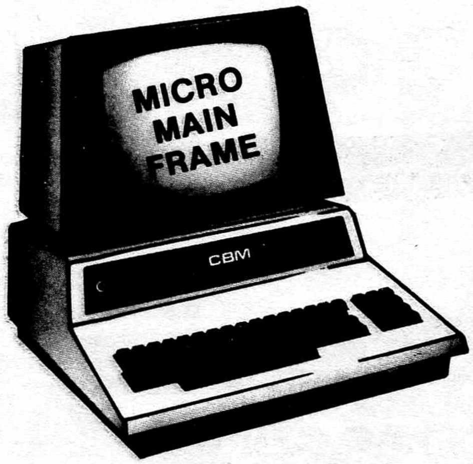
```
10 PRINT "(clr)"
```

means that you would press the SHIFT and CLR keys at the same time but would not include the (clr) in actual program.

American Peripherals

122 Bangor Street • Lindenhurst, N.Y. 11757 • 516-226-5849

SUPER PET! MICRO - MAINFRAME



$1995

APL
FORTRAN
COBOL
PASCAL
BASIC
6809 Assembler

All included at this incredibly low price.

134K of memory.

Schools use timeshare systems so they can access other languages. Why timeshare at $5000 rental per terminal per year when you can purchase a multi language system at a small fraction of the rental price. At this price, timesharing has become obsolete.

Disk Drive: $1795
1 Megabyte disk drive: $1795

Printers: from $795

This system was developed by Commodore in conjunction with the University of Waterloo for use in educational settings.

AMERICAN PERIPHERALS

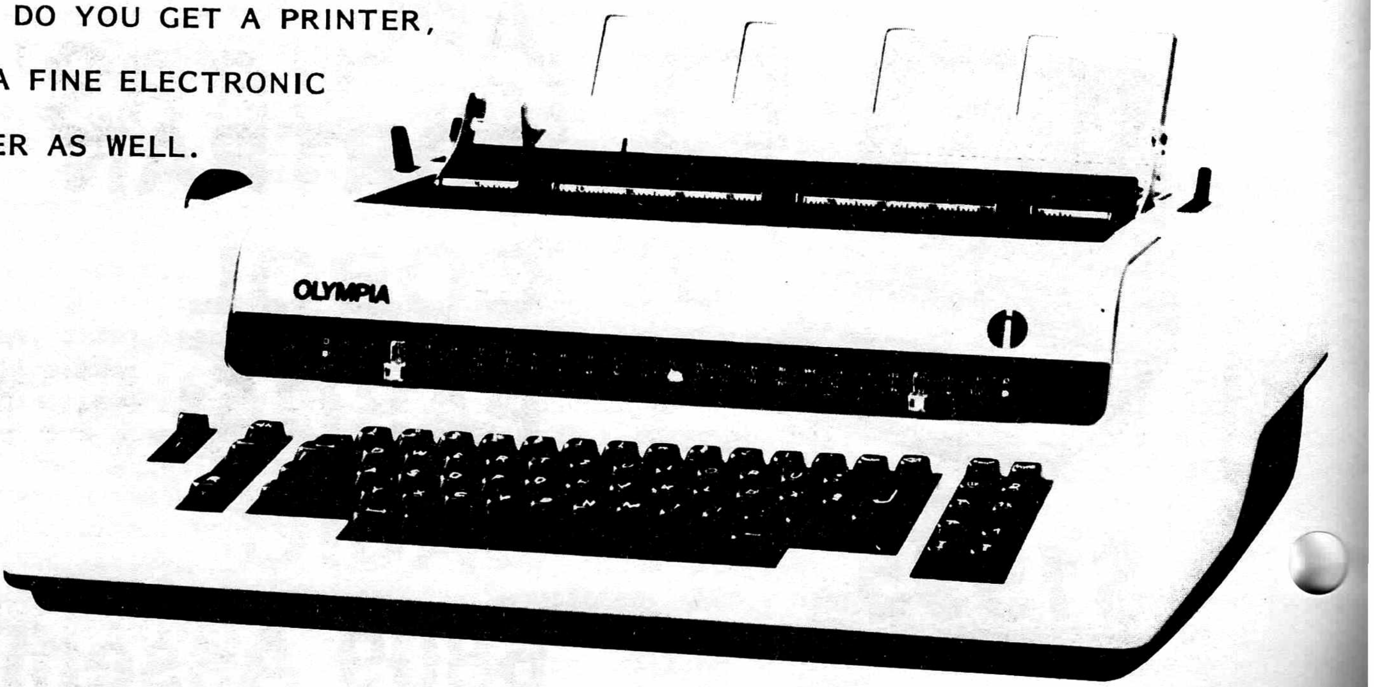
122 BANGOR STREET • LINDENHURST, N.Y. 11757 • 516-226-5849

Olympia ES100

Type Wheel Electronic Typewriter

WORKS GREAT WITH THE PET!

NOT ONLY DO YOU GET A PRINTER,
YOU GET A FINE ELECTRONIC
TYPEWRITER AS WELL.



Performance Highlights

New sensational microprocessor technology that is totally electronic — there are no more levers or elements.

Extremely refreshing quiet operation.

Easily exchangeable typing wheels that are picture sharp, weigh only 5.8 grams and are electronically guided.

The keyboard is pleasantly familiar and simplified to greatly reduce typing stress.

It has a special understanding 8-step correction memory that removes typing errors painlessly.

It has a constantly correct 8-step buffered keyboard.

There are 88 REPEAT typing and function keys.

There is a visual signal diode to indicate the position of the typing wheel.

The paper insertion and ejection is automatically controlled.

The correcting system utilizes two different types, lift off or cover up.

The ribbon system utilizes five different types of cassettes: carbon, multi-strike, lift-off carbon, fabric or offset print.

There is a platen regulator that enables adjustments according to the number of copies required.

There is an acoustical signal to indicate when margin instructions are carried out.

The pitch key brings the pitch into line with the selected typestyle.

Its modular construction makes it extremely simple to service.

The ES 100 is ergonomically designed so that the typist sits in a natural, comfortable, relaxed position.

The written text is always visible.

It weighs 1/3 less than a normal electric typewriter.

It indexes vertically in either direction.

18 characters per second (180 Baud)

commodore

PET interface and cables included $1500.00

AMERICAN PERIPHERALS

commodore

122 BANGOR STREET • LINDENHURST, N.Y. 11757 • 516-226-5849

Now with 12" Screen & Bleeper!

Get a 16K PET for $663[1] (or a 16K APPLE for $2,500[2])

Limited Time Offer

Good for 16K PET, 32K PET, and 8032 CBM.

Here are the rules:

1. Send purchase order for:
   A. 2 units at regular price
   B. 1 free unit
2. After receipt of payment, free unit is shipped direct to you.

[1] This price applies for the 3 for 2 deal only. See details on back.

[2] This price assumes the most common Apple configuration: 16K computer ($1330), single disk drive and controller ($645), and 13 inch color monitor ($550).

HARDWARE & ACCESSORIES

CODE	PRICE	DESCRIPTION
4095	$450.00	High Resolution Graphics or Visible Memory.
5000	$110.00	Terminal Interface. This allows a Pet to be used as a smart terminal.
5005	$225.00	TNW-2000 RS232 two way interface.
5010	$169.00	ADA 1450. One way interface.
5020	$295.00	Sadi. Two way RS232 one way par. Most flexible interface for most printers to date.
5024	$1900.00	Datronics Test Scorer with Pet interface.
4037	$395.00	Voice Synthesizer by CBM. Features phoneme synthesis for vocabulary construction (rather than memory limited digital techniques).
4038	$279.95	Modem by CBM. High performance 300 BAUD IEEE interfaced modem features accurate teleprocessing communication for Pet systems.
4045	$50.00	Programmers Toolkit. This is a Rom that allows resequencing, block deletions, tracing, searches, merging programs. Built in 8K $79.00.
4050	$39.00	Space Saver. Allows use of toolkit and word processor simultaneously.
4055	$40.00	Tractor paper. Box of 3500 sheets. ($10.00/1000 sheets) $5.00/box for shipping.
4060	$15.00	Plastic cover for PET.
4005	$1.00	C-10 Cassettes in hard plastic cases. Box of 250 for $220.00
4015	$1.25	C-20 Cassettes in hard plastic cases. Box of 250 for $275.00
4020	$50.00	Floppy disks. Price for 10
4025	$6.00	Tape organizer tray. Carries 15 tapes.
4027	$25.00	Desk top organizer. Holds 36-72 tapes.
4028	$4.50	Disk holders. Holds 10 disks.
4030	$35.00	Sound for your PET. Includes amplifier, connectors, cables and sample programs.

First, a little planning. We know we will want to be able to add items to our inventory and see a list of what we have. In addition we will want to be able to order more of an item, sell an item and change the prices. Our data should include the name of the item, its wholesale and retail price, the number we have in stock and a reorder level. The latter is a number that tells us when we should order more of an item. It is a good practice to put the data at the end of a program but let's enter an example data item now.

```
10000 REM NAME WPRICE RPRICE STOCK REO
10001 DATA WIGETS, 11, 19, 10, 9
```

Near the beginning of the program we should have an initialization section to set important variables.

```
10 CLR: PRINT "(clr)"
30 NI= 1: DL= 10001
40 :
```

NI is the number of items we have in the data and DL is the last data line that has been used. Line 40 is a nice way to leave space between program lines. Next we need our menu with the options we discussed.

```
160 REM MAIN MENU
170 PRINT "(clr)(rvs)O(off)RDER
180 PRINT "(rvs)S(off)ELL
190 PRINT "(rvs)A(off)DD
200 PRINT "(rvs)C(off)HANGE PRICE
210 PRINT "(rvs)L(off)IST ALL ITEMS
220 PRINT "(rvs)E(off)ND
230 INPUT "(2 down)CHOICE   X(3 left)";C$
```

The use of (rvs) and (off) simply "light up" the first letter of each choice to show that the user need only type the one letter in response to CHOICE. The INPUT statement in line 230 has the default value X. This is not a command but will prevent the program from ending if RETURN is accidentally pushed. To use a default like this type your normal prompt such as CHOICE and then three spaces. Now enter the default value which may be a number, letter or word. After this hit SHIFT and LEFT CURSOR for every character in the default you entered and then hit two more. Try experimenting some with this.

We now must transfer control of the program to various subroutines based on the user's response. Let's do this first and then I'll offer some explanation.

```
240 IF C$="E" THEN PRINT "GOODBYE": END
250 FOR I=1 TO 5: IF C$=MID$("OSCAL",I,1) THEN 270
260 NEXT: PRINT "THAT COMMAND DOESN'T EXIST": GOTO 230
270 ON I GOSUB 1000, 2000, 3000, 4000, 5000
280 GOTO 160
```

This small block of code contains many important techniques. Line 240 ends the program if that was the user's choice. Line 250 and 260 check to see if the user has entered a valid command. Another way to do this is with an IF...THEN statement for every command but the way given is much shorter. A loop is constructed with an upper limit equal to the number of possible commands. We then use ONE IF...THEN statement to search a command string using the MID$ command. In this case the command string is OSCAL. MID$ takes this string goes over I characters and looks at 1 character at a time. If it finds a command character then control is transfered to line 270. If it goes through the entire loop with no match then we know the user has entered a non-existent command and we say so and ask him to enter it again. Line 270 uses the powerful ON...GOSUB structure which again takes the place of many IF..THEN statements. Say we entered A for our command. Since A is the third command, the value of I when we got to line 270 would be 3. ON...GOSUB works by counting through the list of line numbers following the GOSUB. It counts based on the value of the variable following the ON. In this case it counts over three line number and goes to the subroutine starting at line 3000. When we return line 280 sends us back to the main menu.

Let's now set up all our subroutines as simple skeletons with a label and RETURN but nothing else at this time.

```
1000 REM ORDER AN ITEM
1110 RETURN
1120 :
2000 REM SELL AN ITEM
2120 RETURN
2130 :
3000 REM ADD AN ITEM
3150 RETURN
3160 :
4000 REM CHANGE PRICES
4130 RETURN
4140 :
5000 REM LIST ALL ITEMS
5080 RETURN
5090 :
```

This should allow us to run our program and make choices in the main menu. Nothing really will happen since all the routines are really not there. The simplest one to do is the LIST that starts at 5000 so we will start with that. We will print a heading, the information for each item and a line to separate one from the other.

```
5010 PRINT "(clr)NAME","WHOLESALE","RETAIL"
5015 PRINT "(down)","IN STOCK","REORDER"
5020 RESTORE: FOR I=1 TO NI
5030 READ N$, WP, RP, IS, RE
5040 PRINT "(down)"N$,WP,RP
5045 PRINT "(down),IS,RE
5050 FOR J=1 TO 39: PRINT ".";: NEXT: PRINT
5060 NEXT I
5070 GET SP$: IF SP$="" THEN 5070
```

The word RESTORE in line 5020 resets the data so that all of it can be read from the beginning more than one time in the same program. We know that we have NI items and so we use this for the upper limit of our loop. In line 5030 we read the name of the item (N$), the wholesale price (WP), the retail price (RP), the number of items in stock (IS) and the reorder number (RE). These are then printed followed by the separator created in line 5050. Line 5070 uses GET to wait until a key is pushed at which time control is returned to the main loop.

The next routine we will add is the one that allows us to add new items to our inventory. This routine begins at line 3000. The procedure is quite straightforward. We enter the item's name and the information about it using INPUT. A check should be included to see that we are not trying to add an item we already have. After we are through the new item must be added into its own data statement. This means we must now have two new subroutines, one for checking for duplicate items and one for the data entry.

```
9000 REM SEARCH FOR DUPLICATE ENTRY
9010 RESTORE: FOR S=1 TO NI
9020 READ N$, WP, RP, IS, RE
9030 IF I$=N$ THEN RETURN
9040 NEXT S
9050 RETURN
```

I$ here is the variable I will use for the name of the item being added. Notice that tnere are two ways to return from this routine. If we return as a result of line 9030 then a match was found, S is less than or equal to NI, and we do not want to enter I$ as a new item. If we return from line 9050 then S will be greater than NI, no match was made, and we will enter I$ as a new item in the inventory. Now to flesh out the add subroutine itself.

```
3010 PRINT "(clr)ADD A NEW ITEM(down)
3020 INPUT "NAME OF ITEM   X(3 left)";I$
3030 IF I$="X" THEN RETURN
3040 GOSUB 9000: IF S<=NI THEN PRINT"THAT ITEM WAS FOUND": GOTO 3020
3050 INPUT"WHOLESALE PRICE   X(3 left)";WP$
3060 IF WP$="X" THEN RETURN
3070 INPUT"RETAIL PRICE   X(3 left)";RP$
3080 IF RP$="X" THEN RETURN
3090 INPUT"# TO ORDER   X(3 left)";IS$
3100 IF IS$="X" THEN RETURN
3110 INPUT"REORDER #   X(3 left)";RE$
3120 IF RE$="X" THEN RETURN
3130 N$=I$: WP=VAL(WP$): RP=VAL(RP$): IS=VAL(IS$): RE=VAL(RE$)
3140 GOSUB 8000: REM AUTO DATA ENTRY
```

Note tnat is usually best to enter all data using a string variable. This will avoid tne annoying ?REDO FROM START that occurs if a letter is entered for a numeric variable. Line 3130 sets things up for the automatic data entry by setting numeric variables such as WP equal to the value of the string variable that was entered. The value of a number is just that number while the value of letters is 0.

We are now ready to write the data entry routine at line 8000. We will start with a semi-automatic routine first. This means that we will print a new data line with the item we are dealing with and all its information. We will also print line 30 which tells us how many items we have and what data line was last used. Both of these will be incremented by one to reflect the new item. Finally we will print RUN and home the cursor. If we now hit RETURN three times we can enter the two program lines and then RUN the program. Let's do this first and then automate the process so we do not have to hit the returns.

```
8000 REM AUTO DATA ENTRY
8010 NI=NI+1: DL=DL+1
8020 PRINT "(clr)(down)"
8030 PRINT DL"DATA "N$","WP","RP","IS","RE
8050 PRINT " 30 NI="NI":DL="DL
8060 PRINT "RUN"
8070 PRINT "(home)";
8090 END
```

To free us from hitting RETURN we tell the PET that three keys were pressed and that tnose three keys were carriage returns. The PET has an area of memory called the keyboard buffer. As you type the characters are placed in the keyboard buffer and then on the screen. This prevents losing characters if a typist is fast. There is one location that keeps track of how many keys have been hit and 9 others that record what these keys are. In BASIC 1.0 the number of keys is stored at 525 while what the keys are begins at 527. In BASICs 2.0 and 4.0 these locations are 158 and 623. First, we add a line to the initialization part of our program which sets two variables equal to the values for the machine that the program is operating on.

```
   20 KB=525: K1=527: IF PEEK(50003) THEN KB=158: K1=623
```

Location 50003 is 0 in BASIC 1 but is not zero in BASICs 2 and 4. This line makes the program "smart" so that it will work on all three versions. We now add one line to our data entry routine and everything is automatically done for us.

```
   8080 POKE KB,3: POKE K1,13: POKE K1+1,13: POKE K1+2,13
```

Here we have said that three keys have been hit and we have stored a carriage return (code 13) in the first three locations of the keyboard buffer. When the routine is accessed the lines will automatically be added to the program, the screen will clear and the program will run again.

All that remains now is to complete the routines for ordering, selling and changing the prices on items. Note that here we want to check to make sure we DO already have the item. When we access the data entry routine at line 8000 we DO NOT want to increment NI since we are NOT adding a new item.

First we'll order an item thereby increasing the number in stock (IS).

```
   1000 REM ORDER AN ITEM
   1010 PRINT "(clr)ORDER AN ITEM(down)"
   1020 INPUT "NAME OF ITEM   X(3 left)";I$
   1030 IF I$="X" THEN RETURN
   1040 GOSUB9000: IF S>NI THEN PRINT"THAT ITEM WAS NOT FOUND": GOTO 1020
   1050 PRINT "(down)NAME","IN STOCK","REORDER(down)"
   1060 PRINT N$, IS, RE
   1070 INPUT "(down)NUMBER TO ORDER   X(3 left)";O$
   1080 IF O$="X" THEN RETURN
   1090 IS=IS+VAL(O$)
   1100 DL=10000+S: OF=1: GOSUB8020
```

Here we use the subroutine call in line 1040 to check to see if the item we are ordering is already in our inventory. If it is not found then S will be greater than NI and our error message will be printed. If it is found we go on to print a heading and the name of the item along with the number presently in stock and the reorder number. In line 1090 we add the number we just ordered to the previous stock. Now the next part is a little tricky. Since this item already exists we only want to change its present data line. We do not want to add another data line or increment NI, the number of items. So, we set DL and then set a flag (OF) to tell the subroutine at line 8020 that we are ordering. Notice that we call the subroutine at 8020 instead of 8000 to avoid changing NI and DL in line 8010. We also must add a line to the automatic data entry routine so that line 30 will not be changed when no new items have been added.

```
   8040 IF OF=1 THEN PRINT: OF=0: GOTO 8060
```

The sell routine is much like that for ordering except we decrease our stock. We still do not want to add a new item or a new data line nor do we want to disturb line 30.

```
   2000 REM SELL AN ITEM
   2010 PRINT "(clr)SELL AN ITEM(down)"
   2020 INPUT "NAME OF ITEM   X(3 left)";I$
```

```
2030 IF I$="X" THEN RETURN
2040 GOSUB9000: IF S>NI THEN PRINT"THAT ITEM WAS NOT FOUND": GOTO 2020
2050 PRINT "(down)NAME","IN STOCK","REORDER(down)"
2060 PRINT N$, IS, RE
2070 INPUT "(down)NUMBER TO ORDER   X(3 left)";O$
2080 IF O$="X" THEN RETURN
2090 IF IS-VAL(O$)<0 THEN PRINT "CANNOT SELL MORE THAN YOU HAVE": GOTO 2070
2100 IS=IS-VAL(O$)
2110 DL=10000+S: OF=1: GOSUB8020
```

The only real difference here is that we use line 2090 to check to see if we have in stock the number we wish to sell. This could easily be turned into a back order!

The last major routine we need is one to change the prices. This is essentialy the same as the last two, except the prices will change, not the number in stock.

```
4000 REM CHANGE PRICES
4010 PRINT "(clr)CHANGE A PRICE(down)"
4020 INPUT "NAME OF ITEM   X(3 left)";I$
4030 IF I$="X" THEN RETURN
4040 GOSUB9000: IF S>NI THEN PRINT"THAT ITEM WAS NOT FOUND": GOTO 4020
4050 PRINT "(down)NAME","WHOLESALE","RETAIL(down)"
4060 PRINT N$, WP, RP
4070 INPUT "(down)WHOLESALE PRICE   X(3 left)";WP$
4080 IF WP$="X" THEN RETURN
4090 INPUT "(down)RETAIL PRICE   X(3 left)";RP$
4100 IF RP$="X" THEN RETURN
4110 WP=VAL(WP$): RP=VAL(RP$)
4120 DL=10000+S: OF=1: GOSUB8020
```

Again we set the flag in line 4120 and call the subroutine at 8020. This avoids changing NI or line 30 since we have not added any new items.

I want to add one final routine that has not been discussed. Everytime I run the program I want it to warn me if an item is at the reorder level. For each item that is at that level the program should give me a chance to order more but not force me to. Since this will be done at the beginning of each run we'll put it near the beginning of the program. All we have to do is check the number in stock (IS) against the reorder number (RE) for each item. For any item that is low we call the order subroutine.

```
50 REM AUTOMATIC REORDER WARNING
60 FOR S=1 TO NI
70 READ N$, WP, RP, IS, RE
80 IF IS>RE THEN 140
90 PRINT "(clr)REORDER WARNING(down)"
100 PRINT "NAME","IN STOCK","REORDER"
110 PRINT N$, IS, RE
120 PRINT: GOSUB 1070
130 NEXT
```

Our program could use many more useful additions many of which came to mind as I wrote this explanation. We could keep track of total sales and costs for one session. We could do the same for a longer period of time using the automatic data entry method. A routine to delete an item would be interesting but somewhat more complex. If you have other suggestions please send them to me and I'll try to incorporate them in the program. Also send other suggestions for other programs and techniques you would like to see.

High Resolution Bar Graphs

by Anselm Wachtel

The following short program prints (actually POKEs) bar graphs in either a horizontal or vertical direction with single pixel resolution. The algorithm is similar to that used in COPLOT (COMPUTE! April 1981), but the characters cannot conveniently be extracted from a string via the MID$ function. However, they can be read from an array. The demo program is practically self-explanatory.

The salient point of the algorithm is that the quantity J (lines 80, 160), which can take values from 0 to 8, points to the specific character to be used from the array. If J=0, this part is skipped. I have also provided for rounding (rather than truncating) of the fractional part. This improves precision perceptibly.

Lines 70 and 150 generate the quantities to be plotted. I have purposely made the increments slightly smaller than the resolution which is 1/8 or .125. This causes some of the bars to be plotted with equal length. The reader can experiment by modifying these lines.

My Axiom 801P printer provides additional paper feed with each line, and this cannot be varied. This causes the vertical bars to consist of separate segments. To make graphs look really good a little work with scissors and tape is needed. The CBM printers can be adjusted to print contiguous graphics vertically. Since POKEs go directly to the screen and not to paper, you must use some sort of screen dump like KEYPRINT to get your results on the printer.

I am indebted to Mark Zimmerman for pointing to the use of the appropriate POKE symbols.

```
0 PRINT "(clr)   HIGH RESOLUTION GRAPH DEMO"
5 REM A. WACHTEL, PITTSBURGH, PA  1981
20 DATA 101,116,117,97,246,234,231,160
30 DATA 100,111,121,98,248,247,227,160
40 FOR J=1 TO 8: READ F(J): NEXT J
50 FOR J=1 TO 8: READ G(J): NEXT J
52 INPUT "(down)(rvs)H(off)ORIZONTAL OR (rvs)V(off)ERTICAL BARS";A$
54 IF A$="V" THEN 140
60 PRINT "(clr)";
70 FOR K=0 TO 20: X=20+0.1*K
80 XI = INT(X): J = INT(8*(X-XI)+0.5)
90 PRINT K;TAB(4)"(shift +)"
100 FOR I=1 TO XI: POKE 32773+40*K+I,160: NEXT I
110 IF J=0 THEN 130
120 POKE 32773+40*K+I,F(J)
130 NEXT K: END
140 PRINT "(clr)(20 down)"
150 FOR K=0 TO 15: X=15+0.1*K
160 XI=INT(X): J=INT(8*(X-XI)+0.5)
170 FOR I=0 TO XI-1: POKE 33573+2*K-40*I,160: NEXT I
180 IF J=0 THEN 200
190 POKE 33573+2*K-40*I,G(J)
200 NEXT K
205 PRINT"      ";:FOR Q=1 TO 3
210 PRINT "(shift 1)(9 shift @)";
215 NEXT Q: PRINT "(shift 1)"
220 PRINT "     0         5         10        15"
```

(Note: We apologize for the strange way of presenting lines 205-215. Whenever possible we try to type short programs directly into WordPro. This reduces work for us but, as you know, WordPro will not print PET graphics. - pub)

Adding Commands to BASIC with CHRGET

by Doug Haluza

In the article "Caling all ML Programs" on page 21 of the last issue of The PAPER Ralph Bressler only briefly touched on how to use CHRGET with machine language programs. In this article I hope to explain what CHRGET is and how to use it with aa simple example.

BASIC is an interpreted rather than a compiled language. This means that rather than translating the BASIC code you give it into the machine language the 6502 microprocessor understands, the PET runs a large program written in maching language that interprets the BASIC code you give it and performs the necessary operations. This large program, called the BASIC interpreter, is contained in ROM (Read Only Memory). ROM memory's main advantage is that it is non-volatile, that is it doesn't lose its contents when the power is turned off, so every time you turn on your PET the BASIC interpreter is 'READY' to execute commands and run programs for you.

Unfortunatly ROM's contents are permanently etched in silicon and cannot be changed, so Microsoft, the authors of PET BASIC, put a small piece of the BASIC interpreter in RAM (read/write memory). This short section of code is the CHRGET routine. BASIC uses this routine each time it needs another character of BASIC code or command to interpret. Beacuse CHRGET is in RAM it can be changed. By linking CHRGET to our own machine language routine we can examine each character of BASIC code that is about to be interpreted, and if that character part of one of the commands we wish to add to BASIC our machine language routine can take over to interpret the command and produce the desired result.

The Programmers Toolkit, Command-O, Univerdal DOS (The Wedge) and other utility programs that add commands to BASIC intercept CHRGET and check each character to be interpreted to see if that character is part of a utility command. Because checking each character takes time these utilities often use a key character (i.e. >,@,/,],!, etc.) that precedes the command to save time. By only checking for the key character(s) the utility can quickly check for its commands and return other characters to BASIC.

Before we try to use CHRGET let's take a look at the BASIC 3 & 4 versions of CHRGET to see how it works (BASIC 1.0 is identical, but it starts at $00C2):

```
0070: E6 77     CHRGET INC TXTPTR    ;Increment low byte of pointer
0072: D0 02            BNE CHRGOT    ;No carry?, don't...
0074: E6 78            INC TXTPTR+1  ;Increment hi byte of pointer
0076: AD YY XX CHRGOT LDA $XXYY      ;Get character
0079: C9 3A            CMP #':'      ;Set zero flag if = ':'
007B: B0 0A            BCS EXIT      ;If > '9', RTS with carry set
007D: C9 20            CMP #' '      ;Is it a space?
007F: F0 EF            BEQ CHRGET    ;Yes, ignore it and get another
0081: 38               SEC
0082: E9 30            SBC #$30
0084: 38               SEC
0085: E9 D0            SBC #$D0      ;Clear carry if >= '0' (and <= '9')
0087: 60        EXIT   RTS           ;RTS w/Z Flag set if ':' or end of line
```

The pointer to the current BASIC character to be interpreted is stored at $0077 and $0078. If the BASIC interpreter calls CHRGET, the pointer is incremented, and the next BASIC character is loaded into the accumulator (A) and some flags are set. If CHRGOT is called by BASIC, the current character is reloaded into A and the flags are reset. After some tricky maneuvering the character is returned in A (if it's not a space) with the Z flag set if it's a colon or the end of a line, and the carry flag cleared if it's numeric (0-9).

To intercept CHRGET with a machine language routine all we have to do is insert a 'JMP UTILITY' instruction at the beginning of CHRGET that jumps to our utility, checks for utility commands, and then returns to CHRGET. Unfortunatly with this method The utility must increment the pointer and then do an indirect load to get the character. Remember to do an indirect load either the X or Y register must be used, but since CHRGET is called while BASIC is running, we must not disturb either the X or Y registers. Either the X or Y register will have to be saved and zeroed before the load, and restored after the load. This is not too hard to do, so many utilities intercept CHRGET here.

If CHRGET is intercepted right after CHRGOT these problems are eliminated. The pointer is incremented and the character is already loaded into A by the time the JMP to the utility is executed. The 'CMP #':' and 'BCS EXIT' instructions are overwritten by the 'JMP UTILITY' instruction that was inserted to intercept CHRGET, so these instructions must be performed before the utility returns to CHRGET. Most utilities intercept CHRGET in this way.

The example utility below intercepts CHRGET after CHRGOT, and does a little more. Rather than just inserting a JMP into CHRGET the old CHRGET routine is swapped out and another routine is put in its place. This method has several advantages. It allows this utility to be used with other utilities that intercept at the beginning of CHRGET (i.e. The Wedge), if they return to CHRGET at or right after CHRGOT. Utilities that intercept after CHRGOT will be able to check characters after they are checked by this example utility, and therefore will remain enabled. Also since the two CHRGET routines are simply swapped, trying to reenable the example utility will actually disable it and leave everything as it was before.

This example utility does PEEK and POKE in hexadecimal. The backarrow (<-) is used as the key character because it is never used in BASIC syntax. Typing <-8000,01 will POKE $01 to location $8000 (the screen). Typing <-8000 will PEEK location $8000 and print the value in hex.

In short, it works like this (see the assembly listing): The SETUP routine swaps part of the two CHRGET routines. The new routine, which starts at CHPCH, is put in after CHRGOT, and the code that was there is put in after CHPCH. This swaps out a 14 byte block of CHRGET and swaps in a new block that checks for the backarrow and returns to the swapped out code if none is found.

The utility code starts at LETSGO which calls a subroutine to get a 4 digit hex number. This subroutine, GET4HX, calls GET2HX and then falls into GET2HX which calls GET1HX and then falls into GET1HX. GET1HX reads a BASIC character and converts it from ASCII to hex if it's a valid hex digit, otherwise it gives a 'SYNTAX ERROR'. GET2HX gets two digits in one byte and returns this byte in location $00 and the Y register. GET4HX gets a word (two bytes) and returns the high part in loacton $02 and the low part in Y. The net result of all this is to get a four digit (two byte) address which is then stored in $01 and $02.

The next character is checked to see if it's a comma. If it is a comma then the POKE routine is called. POKE gets a 1 byte value and stores it at the specified address before returning to BASIC with the next character.

If the next character is not a comma then the PEEK routine loads the contents of the specified address and prints it out. To print this value in hex the hi nibble (4 bits) must be shifted into the low nibble for conversion, then the hi nibble must be masked out to allow the low nibble to be converted. The conversion routine uses BCD (Binary Coded Decimal) arithmetic to convert the hex digit to ASCII for printing by the WRT routine.

This assembly listing is for BASIC 3.0. The program is the same for BASICs 1.0 and 4.0, except for the address declarations at the beginning and the start address. The 1.0 and 3.0 versions are located in the second cassette buffer, while the 4.0 version is located in the first cassette buffer to avoid conflict with 4.0 DOS variable storage. The assembly changes and hex dumps for all three versions are given below so you can try this example on any machine. Use the machine language monitor to enter and save the proper code for your machine.

Assembly Listing

```
100:   033A           QTFLAG    =    $CD        ;BETWEEN QUOTES FLAG
110:   033A           CHRGET    =    $70        ;BEGINNING OF CHRGET
120:   033A           CHRGOT    =    $76        ;RE-GET LAST CHAR
130:   033A           TXTPTR    =    $77        ;POINTER TO CURRENT CHAR
140:   033A           CHRPCH    =    $79        ;PATCH LOCATION
150:   033A           READY     =    $C389      ;WARM START
160:   033A           SNERR     =    $CE03      ;SYNTAX ERROR EXIT
170:   033A           WRT       =    $FFD2      ;PRINT A  CHARACTER
180:   033A                     *=   $033A      ;START IN SECOND CASS BUF

190:   033A A2 0E     SETUP     LDX  #14        ;SWAP 14 BYTES
200:   033C B5 79     SWAP      LDA  CHRPCH,X   ;STARTING AT CHRPCH
210:   033E BC 55 03            LDY  CHPCH,X    ;WITH THE ROUTINE AT CHPCH
220:   0341 94 79               STY  CHRPCH,X
230:   0343 9D 55 03            STA  CHPCH,X
240:   0346 CA                  DEX
250:   0347 D0 F3               BNE  SWAP       ;SWAP ALL 14
260:   0349 4C 89 C3            JMP  READY      ;AND THEN WARM START

280:   034C E6 77     CHGET     INC  CHRGOT+1
290:   034E D0 02               BNE  CHGOT
300:   0350 E6 78               INC  CHRGOT+2
310:   0352 4C 76 00  CHGOT     JMP  CHRGOT
320:   0355 C9 5F     CHPCH     CMP  #"<-"
330:   0357 F0 03               BEQ  ARROW
340:   0359 4C 55 03  NOGOOD    JMP  CHPCH
350:   035C A5 CD     ARROW     LDA  QTFLAG
360:   035E D0 F9               BNE  NOGOOD
370:   0360 4C 64 03            JMP  LETSGO
380:   0363 EA                  NOP

390:   0364 A0 00     LETSGO    LDY  #0         ;UTILITY STARTS HERE
400:   0366 20 91 03            JSR  GET4HX     ;GET ADDRESS
410:   0369 85 01               STA  1          ;IN ($01,$02)
420:   036B 20 70 00            JSR  CHRGET     ;GET NEXT CHARACTER
430:   036E C9 2C               CMP  #","       ;IS THE NEXT CHAR A COMMA
440:   0370 F0 17               BEQ  POKE       ;YES, MUST BE A POKE
450:   0372 B1 01     PEEK      LDA  (1),Y      ;OTHERWISE DO PEEK
460:   0374 48                  PHA             ;SAVE RESULT FOR LATER
470:   0375 4A                  LSR             ;SHIFT HI NIBBLE
480:   0376 4A                  LSR             ;INTO LOW NIBBLE
490:   0377 4A                  LSR
500:   0378 4A                  LSR
510:   0379 20 7F 03            JSR  PNTHEX     ;AND PRINT IT
520:   037C 68                  PLA             ;RETRIEVE ORIGINAL VALUE
525:   037D 29 0F               AND  #$0F       ;SEND LOW NIBBLE
530:   037F F8        PNTHEX    SED             ;CONVERT HEX TO ASCII
540:   0380 18                  CLC             ;WITH A FANCY BCD TRICK
550:   0381 69 90               ADC  #$90       ;FROM LEVENTHAL'S 6502 BOOK
560:   0383 69 40               ADC  #$40
570:   0385 D8                  CLD
580:   0386 4C D2 FF            JMP  WRT        ;THEN PRINT IT AND RTS

600:   0389 20 96 03  POKE      JSR  GET2HX     ;GET VALUE TO POKE
610:   038C 91 01               STA  (1),Y      ;POKE IT IN
620:   038E 4C 70 00            JMP  CHRGET     ;RETURN TO BASIC W/ NEXT CHAR
```

```
630:    0391 20 96 03 GET4HX   JSR  GET2HX     ;GET FIRST TWO HEX DIGITS
640:    0394 85 02             STA  2          ;STORE THEM
650:    0396 84 00    GET2HX   STY  0          ;GET TWO (MORE) DIGITS
655:    0398 20 A1 03          JSR  GET1HX     ;GET FIRST DIGIT
660:    039B 0A                ASL             ;SHIFT IT INTO HI NIBBLE
670:    039C 0A                ASL
680:    039D 0A                ASL
690:    039E 0A                ASL
700:    039F 85 00             STA  0          ;AND STORE IT
710:    03A1 20 70 00 GET1HX   JSR  CHRGET     ;GET A HEX DIGIT IN ASCII
720:    03A4 90 02             BCC  NUMERC     ;AND CONVERT FROM ASCII TO HEX
730:    03A6 E9 08             SBC  #8
740:    03A8 E9 2F    NUMERC   SBC  #$2F
780:    03AA 2C AD 03          BIT  MEMF0      ;CHECK IF LEGAL HEX DIGIT
790:    03AD F0 03    MEMF0    BEQ  OKHEX      ;YES, STORE IT
800:    03AF 4C 03 CE          JMP  SNERR      ;NO, GIVE SYNTAX ERROR
810:    03B2 05 00    OKHEX    ORA  0
820:    03B4 85 00             STA  0          ;ADD (TO EXISTING) DIGIT
830:    03B6 60                RTS             ;AND RETURN
```

BASIC 3.0

```
.:   033A A2 0E B5 79 BC 55 03 94
.:   0342 79 9D 55 03 CA D0 F3 4C
.:   034A 89 C3 E6 77 D0 02 E6 78
.:   0352 4C 76 00 C9 5F F0 03 4C
.:   035A 55 03 A5 CD D0 F9 4C 64
.:   0362 03 EA A0 00 20 91 03 85
.:   036A 01 20 70 00 C9 2C F0 17
.:   0372 B1 01 48 4A 4A 4A 4A 20
.:   037A 7F 03 68 29 0F F8 18 69
.:   0382 90 69 40 D8 4C D2 FF 20
.:   038A 96 03 91 01 4C 70 00 20
.:   0392 96 03 85 02 84 00 20 A1
.:   039A 03 0A 0A 0A 0A 85 00 20
.:   03A2 70 00 90 02 E9 08 E9 2F
.:   03AA 2C AD 03 F0 03 4C 03 CE
.:   03B2 05 00 85 00 60 77 57 77
```

BASIC 4.0

```
.:   027A A2 0E B5 79 BC 95 02 94
.:   0282 79 9D 95 02 CA D0 F3 4C
.:   028A FF B3 E6 77 D0 02 E6 78
.:   0292 4C 76 00 C9 5F F0 03 4C
.:   029A 95 02 A5 CD D0 F9 4C A4
.:   02A2 02 EA A0 00 20 D1 02 85
.:   02AA 01 20 70 00 C9 2C F0 17
.:   02B2 B1 01 48 4A 4A 4A 4A 20
.:   02BA BF 02 68 29 0F F8 18 69
.:   02C2 90 69 40 D8 4C D2 FF 20
.:   02CA D6 02 91 01 4C 70 00 20
.:   02D2 D6 02 85 02 84 00 20 E1
.:   02DA 02 0A 0A 0A 0A 85 00 20
.:   02E2 70 00 90 02 E9 08 E9 2F
.:   02EA 2C ED 02 F0 03 4C 00 BF
.:   02F2 05 00 85 00 60 06 05 05
```

BASIC 1.0

```
.:   033A A2 0E B5 CB BC 55 03 94
.:   0342 CB 9D 55 03 CA D0 F3 4C
.:   034A 8B C3 E6 C9 D0 02 E6 CA
.:   0352 4C C8 00 C9 5F F0 03 4C
.:   035A 55 03 A5 CA D0 F9 4C 64
.:   0362 03 EA A0 00 20 91 03 85
.:   036A 01 20 C2 00 C9 2C F0 17
.:   0372 B1 01 48 4A 4A 4A 4A 20
.:   037A 7F 03 68 29 0F F8 18 69
.:   0382 90 69 40 D8 4C D2 FF 20
.:   038A 96 03 91 01 4C C2 00 20
.:   0392 96 03 85 02 84 00 20 A1
.:   039A 03 0A 0A 0A 0A 85 00 20
.:   03A2 C2 00 90 02 E9 08 E9 2F
.:   03AA 2C AD 03 F0 03 4C 1C CE
.:   03B2 05 00 85 00 60 77 57 77
```

6502 Assembler Package
A Product Review

Type: Software
Model PET: BASIC 1.0, 2.0 or 4.0 - cassette or disk
Source: Human Engineered Software
3748 Inglewood Blvd. Room 11
Los Angeles, CA 90066
Price: $23.95

by Ralph Bressler

In recent months many companies have sent me a variety of products. No company has been more cooperative or sent better material than HES. Some company names describe their location while others are the names of the owners. The name Human Engineered Software actually tells you what this company is trying to do and I can tell you that they succeed in every product they make.

The assembler package they sent was incredibly complete and included the editor program (HESEDIT) for all three ROM sets, the assembler (HESBAL) and the public domain Micromon with instructions. All programs were designed to work with a disk and, where appropriate, were properly constructed to direct output to a printer. The programs were accompanied by around 70 pages of some of the best documentation I have ever seen. The manual not only tells you how to operate the programs but also details the variables and major routines used, shows how to modify the programs for disk or cassette, gives other suggestions for custom modification and includes complete listings of the programs and assembly routines used. Incredible!

HESEDIT

This program is a simple, generalized text editor which could be used to maintain a mailing list or another similar function but whose main function is to edit the source code for the assembler. When the program is run it divides the screen into 3 lines of "Command Area" with line 4 to 25 being the data or text area. A handy reference line is drawn across the screen to help you tell which column you are in. In the text area there are line numbers after which you type up to 35 characters. The line number area is also used for various text modification commands. A special input routine redefines some keys to help in the human engineering. All the cursor controls act normally except CLEAR which has been disabled. The problem of using cursor controls right after inserting is eliminated and the up and left arrows have been defined as forward and backward tabs. RETURN is only pressed when you have given a command that needs to be acted upon.

Files are saved or loaded using the familiar BASIC commands but are enhanced so that only parts of source text may be stored or recalled. This is handy when you want to add or save certain routines but not an entire program.

Commands are provided to move forward or backward in a text file by an entire page, half a page or by single lines. Other commands allow changing from upper to lower case, examining the number of bytes free, ending the program and, of course, inserting lines in the source file. Line commands to manipulate text include insert, repeat a line a number of times and delete a single line or a range. All of these commands are easy to use and become quite natural. I also feel that the commands were kept simple and few in number, yet leave the program very flexible. A series of 11 error messages prevent critical errors and clearly tell you what the problem is. If you end the program accidentally a simple GOTO 99 straightens things out.

Some editors for assemblers will automatically format your source text into the proper label, op code-operand and comment fields if you simply delimit them by spaces. HESEDIT does not do this so you must plan ahead for your longest label and leave space in the label field for it. It would also be nice to be able to freely scroll through text as with Word pro. A search and change function would also enhance many applications. Overall though, remember the price and the fact this is an extremely flexible program. It is available alone for $12.95.

HESBAL

This program comes as a one-pass symbolic assembler which means you must define ALL variables before they are used. This also means that the assembly will be faster than a two-pass version. If you should wish a two-pass version as I did the excellent instructions will show the few changes that are needed. HESBAL reads a data file from cassette or disk. This means that it is much more pleasant to use it in conjunction with the latter. The data file is saved right after tne HESBAL program on tape and takes about 90 seconds to load each time you use the assembler.

As far as I can tell from my limited knowledge of assembly language, HESBAL accepts all standard mnemonics and creates no strange syntaxes of its own. It would be ridiculous for me to try to summarize the manual but here are some comments which may help.

In the one-pass version all symbols must be defined before they are used. Constants may be in hex or decimal. Four pseudo-ops make using the assembler easier. ORG sets the value of the location counter and can be used to change the address at which the code will be assembled. HESBAL will start at $033A if not told otherwise. ORG can also be used to reserve space for a certain number of bytes of data or to position data anywhere in memory. A * is used for the current value of the location counter. EQU is used to assign a value to a label. The value can be a number in either base, the * for present location counter value or a previously defined symbol. Addition and subtraction are allowed in the value. BYT is used to set up tables of data which may be any hex or decimal value which will fit in one byte. END is used to end the program assembly and check for remaining undefined labels. There is no specific pseudo-op to reserve a specific number of bytes of memory for use later. However, using ORG to set the location counter ahead solves this problem.

The manual goes on to explain how to reserve an area of memory at the top of memory. It also shows a convenient way to correct minor errors such as LAD instead of LDA. Using this method allows you to immediately restart the program again without rereading the data file. The instructions also give a nice routine for saving the machine code you generate as DATA statements. Finally, the manual explains the various error codes that may be generated.

HESBAL immediately POKEs the machine code it generates into memory but does check to see you're trying to put the code in ROM. It also will output the assembly listing to the printer.

HESBAL is somewhat slower than many other assemblers I have used. This might be a problem on longer programs. Since it POKEs code directly into memory long programs might be impossible to store since they would overwrite the assembler. The program will work well with a tape but reading the data file from tape is painfully slow

Overall, this package is an excellent value for any beginner in assembly language and will be very adequate for many of the jobs more advanced programmers will want to do. Any tape based package suffers from the general drawbacks of cassettes and this system is no exception. Switching from the editor to the assembler and waiting for the data file to load can seem to take hours. Still, consider the alternative of a $1295 purchase for a disk drive. I prefer using the PET's own screen editor to do my editing but if I can't have that HESEDIT is a friendly alternative. The strongest points of this system are its unsurpassed documentation and its human engineering. Watch for more products from this company! What's in a name? In this case plenty.

IDPC fullFORTH+
A Product Review

by Ralph Bressler

Type: Software
Model PET: BASIC 2.0 or 4.0
Source: IDPC Co.
        Box 11594
        Philadelphia, PA 19116
Price: $75 for fullFORTH+
       $28 for Target Compiler

If you think that a review of this product looks familiar you are right. fullFORTH+ was reviewed several issues ago in these pages. I was very pleased with the package then but several things have prompted me to review the package again. To eliminate the suspense, this package is still one of the best values around but now it's even better.

I recently attended a conference where a presentation of FORTH was featured. The version used was one for the Apple but I was impressed with the fact that it looked almost identical to my version on the PET. Some of the unusual and interesting parts of FORTH were discussed. FORTH is a language of "words". There is a dictionary of words supplied from which you can build new words which can be added to this dictionary. The beauty is that you actually TEACH the computer what you want it to do. There is no rigid set of commands. Most versions of FORTH are compiled or pseudo-compiled as soon as you define a word and most have a built in assembler. This means FORTH runs much faster than BASIC and, in some cases, almost as fast as machine language. There are no programs but screens of information or defined words which can be stored. FullFORTH+ is a full version of FORTH with words added for Commodore systems. Words are provided for numeric and string handling and disk and printer operations. Words are included to facilitate structured programming which improves readability and allows writing very compact code. It is said that FORTH is a language which allows extremely fast program writing. One feature that initially bothers some, is the reverse Polish or stack oriented notation that is used. For example, to just add two numbers and print the result you would type:

```
4 5 + .
```

This would place the values 4 and 5 on the stack. The '+' adds the top two stack values and places the result on the stack. Finally, the '.' outputs the top value on the stack. Defining a word to cube a number and print out the result might look like this:

```
: CUBE DUP DUP * * . CR ;
```

and you would call it by saying:

```
3 CUBE
```

The word CUBE would take the 3 since it would be the top number of the stack and duplicate it twice so that the stack would then be 3 3 3. It would then multiply the top two numbers and return the result to the stack leaving 3 9. It would do the multiplication again and leave the value 27 on top of the stack. Finally, it would output the top value on the stack and then do a carriage return. Look for some more detailed articles on FORTH in upcoming issues.

Version 1.5 of fullFORTH+ has some improvements over the older versions. First, an increase in efficiency has been obtained since many words have been rewritten. Also, many important new words and structures have been added to the core which is automatically in place when FORTH is loaded. This includes an

IF-DO logic structure, printer control commands, a decompile routine, and numeric input routines. The -OK message which is equivalent to BASIC's READY now indicates the base currently in use. This is handy since doing an operation in the wrong base can cause confusion. Best of all the system will no longer crash when certain keys are accidentally struck. fullFORTH will now run on any Commodore system and on 40 or 80 columns. Updating an old version costs just $7.50

IDPC has also rewritten parts of the manuals and reformatted them to a handier, easier to use size. They are very good and help a lot. If this isn't enough IDPC now has a hotline to answer technical questions not addressed in the manual. I have used it several times and have found return calls prompt and helpful.

IDPC now has a target compiler which allows you to produce a stand alone FORTH program. This program can operate without the FORTH system so that programs may be developed for resale. A manual and implementation notes are provided and do an adequate job of explaining how the compiler works and how to use it. I did find I had to read some things twice and perform some operations more than once to get the hang of it. The compiler always turns FORTH into 6502 machine code. In one mode it will produce a simulated BASIC program complete with a one line SYS to activate it when run. The biggest drawback to the compiler is that its vocabulary is much more limited than the FORTH vocabulary. Also certain procedures that can be accomplished in FORTH cannot be done during compilation. This means that not just any FORTH program may be immediately compiled. Some careful planning will have to be done with new programs, and some old ones may have to be restructured. This probably seems like a bigger problem to me than it really is due to my naivete in FORTH programming.

This package is one of the cheapest full language systems around. It is well documented and supported by a well written newsletter. Updates are reasonably priced and the company is committed to producing an ever better product. A quote from the newsletter might be appropriate here:

> "When making significant improvements in existing programs, many software firms release it as a new product. If you have an older version of their software, you either do without the new features, or pay full price for the new product. We do not believe this is fair - improvements to fullFORTH+ will always be offered to current registered users. We promise that we will never release a version of the FORTH language for Commodore, APPLE, OSI, or CGRS systems that will not be offered to current registered users of fullFORTH+ for the cost of the replacement disk, plus shipping and handling."

With this kind of attitude and support for their customers it is a shame that IDPC has been the victim of an undeserved derogatory review in the newsletter of the FORTH Interest Group, FORTH Dimensions. The reviewer admitted that he had not tried fullFORTH+ and was basing his review on the manual only. This manual was an incomplete version that FIG had promised to just look at and NOT to use as the basis of a review. Furthermore, IDPC was never sent any notification that the review would be published and none after is was published. This is not reviewing but an apparent attempt to discredit a product and one can only wonder about the motives. The FORTH Interest Group is supposed to promote FORTH for all systems and this kind of thing can only damage their own reputation and that of what they seek to promote.

In any case, I hope some of you will purchase these products and let me know of the experiences you have. Do you get the kind of support I get? Are your questions answered and do you get technical help? Is the newsletter worthwhile? If anyone has used this or another FORTH, I would appreciate some articles or comments on the language and the version you use.

CGRS Microtech PEDISK II

A Product Review

Type: Hardware by Doug Haluza

Model: PET/CBM new model, Basic 3.0 or 4.0, 16K or 32K

Source: CGRS Microtech
P. O. Box 102
Langhorne PA 19407

Price: $595 for Model 540: 5.25" Single Drive, Doube Density (143K)
$795 for Model 580 5.25" Single Drive, Quad Density (286K)
$1095 for Model 877 8" Single Drive, IBM Standard (295K)

PEDISK II is a low cost IBM compatible disk drive for the PET/CBM, AIM and SYM. It interfaces to the PET through a small interface board (similar to the Spacemaker) that plugs into the $E000 socket inside the PET. The BASIC ROM that was in that socket plugs into the board. A ribbon cable connects the interface board to up to three 5.25" or 8" disk drives. The drives are housed in a grey metal case that sits beside the PET.

The board has a 2K DOS (Disk Operating System) ROM that adds extra commands to BASIC like the Universal DOS program for the CBM disk does, but PEDISK commands are preceded by an '!' instead of '@' or '>'. These commands are:

!LOAD loads a program from the disk into memory
!SAVE saves a program on the disk
!OPEN forms a sequential or relative data file on the disk.
!INPUT reads a data record from the disk
!PRINT stores a data record on the disk
!CLOSE ends a sequential or relative file on the disk
!LIST gives a directory of files on the disk
!RUN loads and runs a program from the disk
!SYS calls the DOS monitor which contains extra utilities

Niether the disk commands, nor the diskettes themselves are compatible with the 2040/4040 or 8050. The disks use IBM format, which is the same format used on the TRS-80 and many minicomputers. A software routine is available to read or write TRS-80 5.25" diskettes. CP/M and 8" IBM compatible data files can also be read and written by the PEDISK. This allows the PET to be used for remote entry of information that is later thransfered to another larger computer for analysis. For example, PETs equipped with PEDISKs are being used by a parking lot franchise to log parking data and cash tendered to the PEDISK. At the end of the day the diskette is brought to the company's main computer for financial accounting, trend forcasting, etc.

The incompatibility in disk commands is a big problem for people who want to use 'canned' software packages. Although it is not too hard to modify a BASIC program that uses the CBM disk for the PEDISK, it is not trivial either. If the package is written in machine language it is nearly impossible to convert it. Fortunatly the people at CGRS Microtech recognize this and are committed to making software packages available for PEDISK. Currently the MAE assembler from Eastern House Software as well as the Papermate Word Processor, Flex File Data Base Manager, KMMM Pascal, and EARL assembler, all from AB computers, are available for PEDISK. Wordpro 3+ and 4+ will also be available soon.

PEDISK is a practical alternative to the CBM disk drive with several advantages. PEDISK is moderatly less expensive for a single drive system. If a second drive is needed the PEDISK system is significantly less expensive than the CBM dual disk. The second drive and even a third drive can be added at any time; 5.25" and 8" drives can be used at the same time with the same controller. PEDISK loads programs directly to memory, this makes it up to four times faster than the CBM disk. Although the disks are not compatible with CBM disks, their compatibility with other computers can be very useful for many applications.

A Useful "Infinite" Loop

by Jim Fowler

Did you read "Change Your GOTOs to FOR...NEXT Loops" by David Carew (BYTE, Jan. 1981, p 334)? If you tried the first listing on a PET you found it wouldn't work. But the idea is a good one and will run on a PET with a small but important change. Here is the idea: Every time PET does a GOTO it must translate the line number and then look through the whole program until it matches up the line number with the one translated. This takes a significant amount of time. If you have a "backward" GOTO making a loop that is to be executed many times, you will have an annoyingly long dealy in your program. For example: You are going to READ through a table of DATA until you find a certain value:

```
    100 READ X
    110 IF X=100 THEN 200
    120 GOTO 100
```

Here's the good idea: When PET does a FOR...NEXT loop it "makes a record" of the beginning of the loop and, as long as the loop index is not outside the given range, it jumps directly to that point without having to look for it. You can repeat the READ over and over in a loop but you don't know how many times you will have to execute it. You could just set the high number high enough so that you will always have enough, but there is another way and that is to change the index by 0!

```
  100 FOR I = 1 TO 2 STEP 0
  110 READ X: IF X=100 THEN I=2
  120 NEXT I
```

This runs fast and without any hitch. The published listing didn't work because of a peculiarity of PET BASIC. Some of you may have noticed that changing the index, I, to 2 should cause the loop to execute once more! But think of this: You can make the index go upwards (say, 1 to 10) or downwards (10 to 1 step -1) and PET will always do the loop until I is given a value outside the range given. It tests I against the final value and, depending on whether STEP is positive or negative, it tests whether I exceeds or is less than that value. But if the STEP is 0 then the only way to pass the test (ie. to leave the loop) is when the two are equal. That is not what the article said; it set I to 3 to exit from the loop and PET won't do that. You might well ask why not just leave the loop by GOTO: 110 READ X: IF X=100 GOTO 200 and thereby forget all three complications? Read the article by W.D. Maurer, "A Bug in BASIC" on page 188 of the same issue of BYTE. Maybe you will see why leaving a loop except by the NEXT test is a programmer's NO-NO (even though BASIC allows you to do it). Actually there is another reason, too. You can generate an ?OUT OF MEMORY error if you do it often enough. I'll be glad to write another article on this problem if there is any demand.

New 2 Megabyte Floppy

Watch for a new disk drive for the PET that is being developed by an independent company. The box will house two 5 1/4" double sided disk drives with a total of 2 Megabytes of storage, yet it should cost much less than the 8050! Although it will probably have its own data recording format, it will be compatible with the familiar CBM disk commands. In addition it will have ISAM (Indexed Sequential Access Method) data files with alpha-numeric keys. Keep in mind this is only in the prototype stage but things could progress very rapidly.

ML Utility Pac
A Product Review

by Ralph Bressler

Type: Firmware
Model PET: BASIC 2.0 or 4.0; series 2000, 4000, 8000
Source: Competitive Software
        21650 Maple Glen Drive
        Edwardsburg, MI 49112
Price: $79.95 + $2 shipping

There must be at least 3 dozen 'chips' on the market today for the PET/CBM. Some are protection chips with no useful code and others contain useful utilities. With so many chips and only two slots to put them in the user has a real problem. Even with slot expansion devices to allow plugging multiple chips into one space an eventual limit is reached. I must admit that I tend to judge firmware by the PAIC's Toolkit. This is the first chip I ever bought and it is the one I used the most. The ML Utility Pac has also won a place on my PC board in a valuable ROM socket.

This firmware contains so much that it is hard to really describe it all even in a long review. Let me say first that even though it says machine language utilities that should not deter even the beginning BASIC programmer. This chip has enough goodies to suit everyone's needs. Remember that all of these features are in ROM and do not take up any user memory. I feel that the chip's features fall into two major categories. First, there is a greatly expanded machine language monitor and second, a set of useful routines that may be called by SYS commands.

The machine language monitor is truly comprehensive and very useful. It has maintained the basic commands of Commodore's TIM monitor but has added many commands to the list. I must admit that I was a little skeptical since I have several monitors in RAM that are available free from user's groups and contain many of the same features as the chip. However, this firmware goes far beyond any monitor I have seen. Here is a summary of some of the commands:

Simple Asembler: allows you to type in assembly mnemonics and it converts them to op codes

Disassembler: prints a disassembly in any address range and you may then change the code

Transfer: moves code from a range to a new location

New Locater: adjusts the address after a transfer. also takes care of word values in data tables

Relocate: moves a program and updates the addresses in a program. very useful in opening or closing space in a program.

Integrate: dumps memory with the ASCII values at the right

Fill Memory: fills any range with any value

Hunt Memory: searches any range of addresses for any series of bytes or ASCII characters

Auto Repeat: a quick way to turn the repeat keys on and off

Printer: any output may be directed to the screen or to the printer

Screen Dump: the screen may be sent to the printer in normal sized characters or in enhanced characters

Memory Test : a built in test allows testing in any range and prints 'bad' locations

Break Set: allows the programmer to stop a program after a certain number of occurances of a specific instruction

Quick Trace: allows program execution but checks and updates the break set

Walk Code: the user may single step, slow step, or fast step through a program as a disassembly is output

Compare Memory: compares two ranges and prints locations that do not compare

Also included are hex math and conversion features that allow the following features with hex numbers:

1. addition
2. subtraction
3. ANDing
4. ORing
5. Exclusive ORing
6. One's complement
7. Inverse
8. Hex to decimal 2 or 4 digits
9. Decimal to hex 2 or 4 digits
10. Decimal and Hex of any ASCII character

Remember that this is just a brief summary and that all these commands are in addition to the regular monitor commands. All commands are a single letter or shifted letter related to what the command does. One thing I miss that other monitors have is the ability to hold down the up or down cursor key and scroll through a program.

The rest of the features are accessed by SYS commands and a handy reference sticker is enclosed so you don't forget them. Here is a brief description.

The complete Wedge or DOS is available by typing a single SYS command. This is VERY useful if you forget to load DOS and really need it.

A screen dump and enhanced screen dump faithfully reproduces your screen on a printer. This function can be called from a BASIC program.

With two POKEs and a SYS you may display any page of memory on top of the screen. It is interesting to watch page zero change as you type.

Auto repeat may be turned on or off with a SYS command.

If you NEW a program accidentally another SYS command gets it back.

A simple SYS command reverses the entire screen for a nice effect.

With one SYS command you may list programs to the printer exactly as they appear on the screen. The upper and lower case letters will appear properly even when found in quotes.

Finally, this chip includes a routine to properly link two basic files which have been appended by the DOS COPY command. A single SYS updates the chains.

It must be pointed out that the machine language monitor is very much like a monitor in the public domain called variously Extramon, Supermon, or Micromon. I have used all of these and Micromon in particular includes all the chip features and a few more. Also, most of the functions available by using SYS have appeared in some form in different magazines. However, Competitive has changed the code to operate on either BASIC 2.0 or BASIC 4.0 and have relocated all the routines to a nice, safe location. The fact that all these functions are now in a chip and do not have to be loaded from disk or tape makes life a lot easier. So, you could have most of the capabilities of the chip for next to nothing but you might have to relocate some of them and would still have to load them every time you needed them or burn them into an EPROM yourself. I prefer the chip even at the price.

A good manual is included and shows just how and where to install your chip. A new version is ready for the 8000 series and uses the entire 80 columns of the screen. It plugs into the $9000 or $A000 slot. Competitive also offers a BASIC Utilities chip with the same commands as the PAICS Toolkit. It is available for BASIC 2 or BASIC 4 for $39.95. The same chip with the addition of the extra features such as DOS, screen reverse, screen dump, and so on is only $79.95 for the BASIC 2 or BASIC 4 version. A 30-day free trial is also offered but I bet few people send this useful utility chip back.

Programs for Commodore's PET®

Present this ad from THE PAPER and receive $2 off your purchase price. Valid at your local dealer or when ordered direct.

• PROFESSIONAL TOOLS

• Business Researcher (Rvsd. Simplex)	(16k)	$50
• RNAV3 Navigator (Western U.S.)	(16k) $30 (8k)	$25
• Education Pack (High School Math/Sci)		$15

• DISK BOWLING SYSTEM (PET/CBM)

• Leaguebowl-24	(Disk 32k)	$145
• Archivebowl	(for above)	$40
• Allsweepbowl	(for above)	$40
• Tournamentbowl	(Cass. 16k)	$30

• HOME & OFFICE DATA MGMT.

• Deluxe Address (16k)	$40	• Home Inventory	$20
• Home Address Book	$25	• Shopper	$20
• Grocery Mart	$15	• Dinner's On!	$15

• GAMES & ADVENTURES

• Mansion!	$15	• Fur Trapper	$15
• Museum!	$15	• High Seas	$15
• Pentagon!	$15		

Write for details or ask your local dealer.

BRILEY SOFTWARE
P.O. Box 2913
Livermore, CA 94550
(415) 455-9139

Dealers: Letterhead inquiries invited. Photocopies of this ad are NOT valid coupons. One coupon per purchase. This coupon may be redeemed for face value plus 15¢ for handling if it was received from customer upon purchase of one of the above programs. Offer void where restricted by law.

The Good Books from *Cow Bay Computing*

FEED ME, I'M YOUR PET
(Book 1)

LOOKING GOOD WITH YOUR PET
(Book 2)

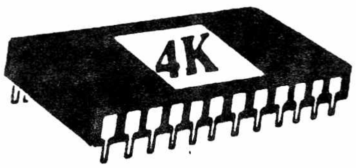
TEACHERS' PET
(Lesson Plan, Answer Key)

Instruction, Classwork, homework worksheets, quizzes for classroom use.

Workbooks are $4.95. TEACHERS' PET is $4.00

CBC COW BAY COMPUTING
BOX 515
MANHASSET, N.Y. 11030

For the Commodore PET/CBM



PLUG IN MORE POWER!

IMMEDIATE DELIVERY

MACHINE LANGUAGE UTILITY-PAC

Rom based firmware includes 43 COMMANDS to ENHANCE use of your computer including D.O.S (WEDGE)!, ASSEMBLER, DISASSEMBLER, HUNT MEMORY, QUICK TRACE, COMPARE MEMORY, TRANSFER MEMORY, RELOCATOR, WALK CODE, INTEGRATE MEMORY (Hex Code and Ascii), VIDEO SCREEN DUMP (STANDARD OR ENHANCED), FILL MEMORY, FAST TYPE HEX ENTRY, HEX TO DECIMAL & ASCII CONVERSIONS and VISE VERSA! Most functions to screen or printer. Makes handling and understanding of machine code programming easier. Also included are these programs accessible from Basic. D.O.S.(WEDGE), LOW CASE LIST, SCREEN DUMPS (STANDARD & ENHANCED), RE-NEW, AUTO REPEAT, DISK APPEND, REV.SCREEN, DISPLAY. AVAILABLE FOR 3.0, 4.0 & 8032 COMPUTERS IN LOCATIONS $A000 or $9000; SPECIFY WHEN ORDERING. MANUAL included. Does not lower user memory. A MUST for new or advanced programmers alike! We accept VISA & MASTERCARD. 30 DAY MONEYBACK TRIAL! SEE REVIEW IN COMPUTE! JUNE 1981 ISSUE! ORDER NOW!

4K ROM for 3.0	(A000) or (9000)	$79.95 + $2 S&H
4K ROM for 4.0	(A000) or (9000)	$79.95 + $2 S&H
4K ROM for 8032	(A000) or (9000)	$79.95 + $2 S&H

DEALER INQUIRIES INVITED!

BASIC UTILITIES 3.0 or 4.0

This 4K Rom contains 19 COMMANDS for Basic programming. INCLUDED are AUTO - RENUMBER - DELETE - FIND - APPEND (TAPE) - DUMP - HELP - TRACE - STEP - OFF - D.O.S. - SCREEN DUMP - ENHANCED SCREEN DUMP - RE-NEW - LOW CASE LIST - AUTO REPEAT - APPEND (DISK) - REV.SCREEN - DISPLAY - THIS ROM IS LOCATED AT $9000. These programs do not lower user memory and will greatly enhance your programming ability through use of the automatic disk & printer routines! 30 DAY MONEYBACK TRIAL, ORDER NOW!

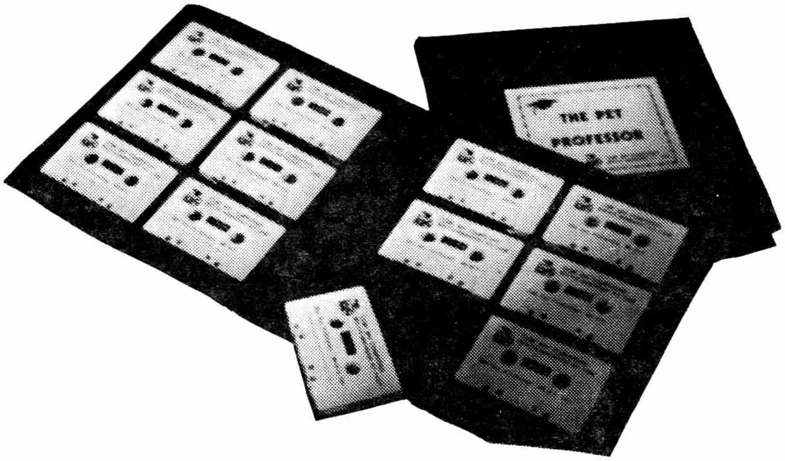
4K ROM	$79.95 + $2 S&H
2K ROM W/FIRST 10 COMMANDS ONLY	$39.95 + $2 S&H

PLEASE SPECIFY WHICH ROM SET YOU HAVE.

SEND $1 FOR CATALOG AND $5 OR $10 OFF OF YOUR NEXT ORDER!

COMPETITIVE
SOFTWARE
21650 Maple Glen Drive
Edwardsburg, MI 49112

Introducing the Pet Professor
All you need to do is decide what you need.



The Pet Professor Arithmetic Software

If you need to teach division of a 2-digit decimal by a 1-digit whole number, we have D-D-1. This program teaches the concept step by step.

Since you probably need to keep student interest high, we go very slowly with a moving cursor. The student is comfortable and involved.

Do you need to drill subtraction of a fraction from a mixed number? Just bypass the teaching part of program F-S-2 and go directly to drill. The nice part is if the student happens to forget, the teaching is still available.

If a test on subtraction of whole numbers with 4-digits, multiple zeros and borrowing is your need, program WN-S-6 is your answer.

You probably also need just about every arithmetic concept that there is. We have them all -- 77 programs.

The directions are simple. Use the **Pet Professor** for all the arithmetic you teach.

CBC COW BAY COMPUTING
BOX 515
MANHASSET, N.Y. 11030

For more information send $5 for a sample tape or write and tell us what you need.

Microphys Programs

PET

Software Specialists
Science and Education

APPLE

Microphys is pleased to announce the availability of its educational software for use with the Commodore PET/CBM and Apple/Bell & Howell microcomputers. These programs have been successfully employed in **Chemistry, Physics, Calculus** and **Mathematics** classes on both the high school and college levels.

The programs are supplied on C-10 cassettes and are accompanied by complete instructions so that even those with little or no computer experience may immediately utilize the software in their classrooms. Each cassette retails for $20 and may be obtained from leading computer dealers or directly from Microphys.

Each **Physics** and **Chemistry** cassette has both a computer-assisted and individualized-instruction program recorded on opposite sides of the cassette. The **CAI program** guides the student through interacts with the computer and receives immediate evaluation of his responses and/or assistance when needed. The **I/I program** generates a unique set of problems for each student. The computer can supply answers so that the student may check his own work. If the teacher directs the computer to suppress these answers, the student completes his work at home and then feeds his results into the computer which grades his work, supplying the answers to those questions incorrectly solved by the student. **NOTE:** each time a particular program is run, a different set of numerical values is generated. In most instances, an entirely new problem is presented. The Mathematic and Calculus cassettes have only the individualized-instruction feature.

For those using disk drives, the programs have been coherently grouped and are available on diskettes. The price of each diskette is $180 which represents a considerable savings with regard to the individual cassette price.

A partial list of the programs available appears below. Please write for the Microphys Winter Catalog which describes the complete line of educational software for use on the PET/CBM and Apple/Bell & Howell microsystems.

CALCULUS CASSETTES

PC726-Differentiation of Algebraic Functions
PC727-Maxima/Minima Problems: Part I
PC728-Maxima/Minima Problems: Part II
PC729-Relative Rate Problems: Part I
PC730-Relative Rate Problems: Part II
PC731-Integration of Algebraic Functions
PC732-Differentiation of Trigonometric Functions
PC733-Integration of Trigonometric Functions
PC734-Integration: Areas of Plane Figures
PC735-Integration: Volumes of Solids
PC736-Integration: Arc Lengths
PC737-Integration: Surface Areas of Solids

M4 **Calculus I Diskette** · contains 726 737

PHYSICS AND CHEMISTRY CASSETTES

1. Linear Kinematics
2. Projectile Motion
3. Momentum and Energy
4. Energy and the Inclined Plane
5. Inelastic Collisions
6. Centripetal Force
7. Pulley Systems — Machines
8. Specific Heat Capacity
9. Calorimetry
10. Heats of Fusion/Vaporization
11. Specific Gas Laws
12. General Gas Laws
13. Thermodynamics I
14. Thermodynamics II
15. Transverse Standing Waves
16. Longitudinal Standing Waves
17. Lenses and Mirrors
18. Refraction of Light
19. Series Circuit Analysis
20. Parallel Circuit Analysis I

20A. Parallel Circuit Analysis II

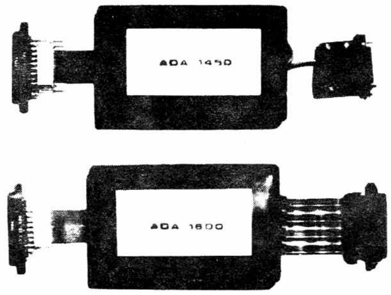
21. Series Parallel Circuit Analysis
22. Faraday's Law
23. Gram-Molecular Mass
24. The Mole Concept
25. The Molarity Concept
26. The Normality Concept
27. The Molality Concept
28. Stiochiometry: Mass/Mass
29. Stoichiometry: Mass/Volume
30. Stoichiometry: Volume/Volume
31. Stoichiometry: General
32. Percent Concentration
33. pH Concept
34. EMF of Electrochemical Cells
35. Electric Field Analysis
36. Photoelectric Effect
37. Symbols and Valence Drill
38. Names of Compounds Drill
39. Formulas of Compounds Drill
40. Total Internal Reflection

P1 **Physics I Diskette** contains the following programs
1 2 3 4 5 6 7 8 9 10 301 302 305 306
P2 **Physics II Diskette** contains the following programs
11 12 13 14 15 16 17 18 19 20 20A 21 35 36 40 303 304

Microphys Programs

2048 Ford Street
Brooklyn, New York 11229
(212) 646-0140

CBM/PET INTERFACES



RS-232 SERIAL PRINTER INTERFACE – addressable – baud rates to 9600 – switch selectable upper lower, lower upper case – works with WORDPRO, BASIC and other software – includes case and power supply.
MODEL – ADA1450 149.00

CENTRONICS NEC PARALLEL INTERFACE - addressable – high speed – switch selectable upper/lower, lower upper case – works with WORDPRO, BASIC and other software – has Centronics 36 pin ribbon connector at end of cable.
MODEL – ADA1600 129.00

CENTRONICS 730/737 PARALLEL INTERFACE – as above but with Centronics card edge connector at end of cable.
MODEL – ADA730 129.00

COMMUNICATIONS INTERFACE WITH SERIAL AND PARALLEL PORTS – addressable – software driven – true ASCII conversion – selectable reversal of upper-lower case – baud rates to 9600 – half or full duplex – X-ON, X-OFF – selectable carriage return delay – 32 character buffer – centronics compatible — much more.
MODEL – SADI 295.00

ANALOG TO DIGITAL CONVERTER – 16 channels – 0 to 5.12 volt input voltage range – resolution is 20 millivolts per count – conversion time is less than 100 microseconds per channel.
MODEL – PETSET1 295.00

REMOTE CONTROLLER WITH CLOCK/CALENDAR – controls up to 256 devices using the BSR X10 remote control receivers – 8 digital inputs, TTL levels or switch closure — 8 digital outputs, TTL levels.
MODEL – PETSET2 295.00

All prices are in US dollars for 120VAC.
Prices on 220 VAC slightly higher.
Allow $5.00 shipping & handling, foreign orders
add 10% for AIR postage.
Connecticut residents add 7½% sales tax.
All prices and specifications subject to change without notice.
Our 30 day money back trial period applies.
MASTER CHARGE VISA accepted.
MENTION THIS MAGAZINE WITH YOUR ORDER
AND DEDUCT 5% FROM TOTAL.

IN CANADA order from: Batteries Included, Ltd., 71 McCaul Street, F6 Toronto, Canada M5T2X1, (416)596-1405.
IN THE USA order from your local dealer or direct: Connecticut microComputer, Inc., 34 Del Mar Drive, Brookfield, CT 06804, (203)775-4595.

Dealer inquiries invited.

CmC

Connecticut microComputer, Inc.
34 Del Mar Drive, Brookfield, CT 06804
203 775-4595 TWX 710 456-0052

Perfectly Balanced.

educational software

from

MICRO-ED

for

PET®

and

VIC®

Send for our free catalog*

*please specify PET or VIC

MICRO-ED, Inc. • P.O. Box 24156
Minneapolis, MN 55424
or telephone us at (612) 926-2292

VIC and PET are the registered trademarks for Commodore Business Machines.

WATCH

THIS

SPACE

WATCH

THIS

SPACE

NEW! IBM COMPATIBLE 8" FLOPPY DISK for CBM/PET,* Aim, Sym

PEDISK II

LOW COST – HIGH PERFORMANCE FLOPPY DISK

5¼" MINI FLOPPY DISK SYSTEMS:

Model 540-1 Single Drive, Double Density (143K)	$595.00
Model 540-2 Dual Drive, Double Density (286K)	$895.00
Model 580-1 Single Drive, Quad Density (286K)	$795.00
Model 580-2 Dual Drive, Quad Density (572K)	$1,195.00

8" MINI FLOPPY DISK SYSTEMS:

Model 877-1 Single Drive, IBM standard (295K)	$1,095.00
Model 877-2 Dual Drive, IBM standard (590K)	$1,695.00

Payable in U.S. Dollars

PEDISK II is a high performance mass storage peripheral to enhance your computer's storage capability. Total storage to 850K bytes is available. The PEDISK II system consists of a small disk controller electronic board that mounts inside the computer and an external disk drive assembly. PEDISK II offers the fastest mass storage system available for the Commodore PET. With a data transfer rate of 250000 bits per second, the PEDISK II loads data directly to memory. This is up to four times faster than any 488 bus-type mass storage device.

One, two, or three drives connect to the PEDISK controller board. The user can choose 5¼" or 8" disk drives. An important feature of the PEDISK II system is its ability to exchange data with other computers. The standard 8" IBM 3740 format allows data exchange with large computers. Data files, mail lists, etc. available on large computers can be processed by the PET/PEDISK system. Remote PET/PEDISK computers can provide data entry for large machines.

PDOS II software links directly to the standard BASIC and operates with BASIC-type commands for easy interfacing. A full set of disk utility commands completes the powerful disk operating firmware.

DISK COMMANDS:

BASIC

!LOAD – reads a program file to the computer
!SAVE – stores a BASIC program file to the disk
!OPEN – forms a sequential or relative data file
!INPUT – reads a data record from a file on the dis
!PRINT – stores a data record to a file on the disk
!CLOSE – ends a sequential or relative data file
!LIST – displays a directory of all files on the disk
!RUN – reads a program file and executes

MONITOR-DOS

D – displays contents of memory or diskette.
G – go to program and execute.
H – help user with listing of all commands.
K – kill a file on the diskette (erase file).
L – read program to the computer memory.
M – memory examine and change monitor.
N – name a file differently (rename).
P – print directory of all files on the disk.
R – return to BASIC mode.
S – save program or data from memory to the disk.
U – utility: format, copy, compress, patch diskette.
X – execute program after loading.

fullFORTH +

FULL FEATURE "FORTH" FOR 6502 SYSTEMS

INTERPRETER – can be executed directly in an interpretive mode to speed testing and debugging.

CROSS-COMPILER – words can be individually compiled and tested, the entire program can also be cross-compiled for maximum efficiency.

COND. ASSEMBLER – Machine language modules can be intermixed and conditionally assembled to fullFORTH.

STRING HANDLING – variable length constants and variables are allowed. Processes compare, move, concatenate and sub-string words.

FLOATING POINT – process 5 or 9 digit integer and floating point numbers for arithmetic operations.

SCREEN EDITOR – contains a unique full cursor visible screen editor.

SPECIFY PEDISK II, PET 2040 or 4040 DISK, OR APPLE $75.00

cgrs
MICROTECH
DEALER INQUIRIES INVITED

FOR INFORMATION, SEE YOUR DEALER OR:

P.O. BOX 102 • LANGHORNE, PA 19047 • (215) 757-0284
U.S.A.

*PET IS A TRADEMARK OF COMMODORE

COLLEGE BOARD SAT PREPARATION SERIES

TRS-80, APPLE, PET, OSI, ATARI, CP/M, PDP-11

Each program confronts the user with a virtually limitless series of questions and answers. Each is based on past exams and presents material of the same level of difficulty and in the same form used in the S.A.T. Scoring is provided in accordance with the formula used by College Boards.

S.A.T., P.S.A.T., N.M.S.Q.T. — Educator Edition set includes 25 programs covering Vocabulary, Word Relationships, Reading Comprehension, Sentence Completion, and Mathematics. Price **$229.95**

Independent Tests of S.A.T. series performance show a mean total increase of 70 points in students' scores.

G.R.E. Series – Educator Edition includes 28 programs covering Vocabulary, Word Relationships, Reading Comprehension, Sentence Completion, Mathematics, Analytical Reasoning and Logical Diagrams. Price **$289.95**

COMPETENCY EXAM PREPARATION SERIES

This comprehensive set of programs consists of simulated exam modules, a thorough diagnostic package, and a complete set of instructional programs. It is designed to teach concepts and operations, provide drill and practice and assess achievement levels through pre and post testing. The Competency Exam Preparation Series provides a structured, sequential, curriculum encompassing mathematical, reading and writing instruction.

This program is designed for individual student use or use in a classroom setting. Programs provide optional printer capability covering worksheet generation and performance monitoring. C.E.P.S. are available in three software formats.

National Proficiency Series	$1,299.00
N.Y.S. Regents Competency Test, Preparation Series	$1,299.00
California Proficiency Assessment Test, Preparation Series	$1,299.00

If desired separate Mathematics and Verbal packages are available for $799.00 ea. A Spanish language version of the Mathematics Instruction Package is available at no extra charge.

INQUIRE FOR UNIQUE M.I.T. APPLE™ LOGO APPLICATIONS SOFTWARE

Time Traveler

The best of the adventure games. Confronts the player with complex decision situations and the demand for real time action. Using the Time Machine, players face a challenging series of historical environments. To succeed you must build alliances and struggle with the ruling power. Each game is unique.

$24.95

Odyssey In Time

This spectacular adventure game adds a new dimension of excitement and complexity to **Time Traveler**.

Odyssey In Time includes all the challenges of **Time Traveler** plus 10 additional eras. Each game is different and may be interrupted and saved at any point for later play.

$39.95

Sword of Zedek

Fight to overthrow Ra, The Master of Evil. Treachery, deceit and witchcraft must be faced in your struggles as you encounter wolves, dwarves, elves, dragons, etc. Each of the twelve treasures will enhance your power by making you invisible, invulnerable, etc. Each game is unique in this spectacular and complex world of fantasy.

$24.95

★ *NEW* *MICRO-DEUTSCH* ★

Micro-Deutsch set includes 24 grammar lessons, covering all material of an introductory German course. Four test units also included. Grammar lessons use substitution transformation drills, item ordering, translations and verb drills. Drill vocabulary based on frequency lists. Suitable for use with any high school or college textbook. Extensively field tested at SUNY Stony Brook. Available for Apple II and PET/CBM. (PET version includes a special foreign language character chip.) Also available soon: MICRO-FRANCAIS, MICRO-ESPAÑOL, MICRO-IVRIT, MICRO-YIDDISH, MICRO-CHINESE, MICRO-JAPANESE. **$179.95**

☆ *NEW* ☆

☆ Pythagoras and The Dragon ☆

Mathematics in a fantasy game context. Based on **The Sword of Zedek**, **Pythagoras and The Dragon** introduces Pythagoras as a mentor to the player. When called on for aid, Pythagoras poses math questions, and depending on the speed and accuracy of the player response, confers secret information. With Pythagoras as an ally, the quest to overthrow Ra, The Master of Evil, assumes a new dimension of complexity. Depending on the level chosen, problems range from arithmetic through plane geometry.

32K **$39.95**

PROGRAMS AVAILABLE FOR
TRS-80, APPLE II, PET & ATARI

Disk or cassette, *please specify*. N.Y.S. residents add sales tax.

All programs require 16K ● TRS-80 programs require LEVEL II BASIC ● APPLE programs require Applesoft BASIC

Krell Software Corp.
21 Millbrook Drive, Stony Brook, NY 11790
(516) 751-5139

New York/New Jersey's Largest Pet® Dealer

Sensational School Offer:

BUY 2 COMMODORE PET® COMPUTERS. GET 1 FREE!*

NOW IN OUR 4TH YEAR

PLUS THESE EXTRAS:

- FREE 13 MONTH WARRANTY (on premises service)
- FREE SOFTWARE WORTH UP TO $2,000 (210 programs with each computer purchased)
- FREE DELIVERY
- BUY TWO PET® CASSETTE RECORDERS, GET ONE FREE!
- AND: if you add on a Floppy Disk Drive, you get an additional $1,000 worth of software!

COMMODORE PET COMPUTERS COST LESS AND GIVE YOU MORE FOR YOUR MONEY. READ OUR CHART.

FEATURES	COMMODORE PET 4016	APPLE II	IBM
Base Price (16K)	$663*	$1,330	$1,565
12″ Green Screen	Standard	299	345
IEEE Interface	Standard	300	NO
TOTAL	$663*	$1,929	$1,910
Upper & Lower Case Letters	Standard	NO	Standard
Separate Numeric Key Pad	Standard	NO	Standard
Intelligent Peripherals	Standard	NO	NO
Real Time Clock	Standard	NO	NO
Maximum 5¼″ Disk Capacity Per Drive	500K	143K	160K

*Based on Commodore's 3 for 2 School Offer, average price when bought in multiples of 3. Prices are as of the most recent published price lists, September, 1981 and approximate the capabilities of the (16K) PET® 4016. Disk drives and printers are not included in prices. Models shown vary in their degree of expandability.

PET® computers—world's leading educational computers—now a standard in American classrooms. Compact, self-contained, excellent resolution 12″ screen, with fantastic graphic possibilities. No wonder, the PET is first in flexibility, versatility and value.

AVAILABLE ONLY AT COMPUTER STRATEGIES:

- Daisy Link Monitor™, which permits instructor to monitor up to 16 computers in classroom and make individual corrections and comments on each student's screen.
- Our exclusive 216-page "No Nonsense Pet Educational Software Catalog."

We also supply complete CBM Business Systems for Word Processing, Tax Preparation, Projection Planning (VisiCalc), Legal Time Accounting and Accounts Receivable/Payable.

COMPUTER STRATEGIES, INC.

NEW YORK-NEW JERSEY'S LARGEST PET® DEALER
300 N. Main St. (Hillcrest Prof. Bldg.), Spring Valley, N.Y. 10977
N.Y. (914) 356-7770 N.J. (201) 664-2294

JUST OUT, "MICRO/MAINFRAME" SIX LANGUAGES!

COMPUTER STRATEGIES, INC.

80 x 25

PET/CBM™

2000/3000/4000 Series

not using a CRT, or display controller chip

$275.00*

Select either
80 x **25** or **40** x **25**

On The
Built-in
Display

From the keyboard or program

Displays the full, original character set

Available from your local dealer or:

EXECOM CORP.
1901 Polaris Ave.
Racine, WI 53404
Ph. 414-632-1004

*Plus installation charge of $75.00

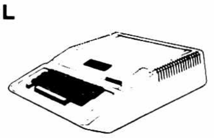
Available only for Basic 3.0 & Basic 4.0

PET& CBM™a
trademark of Commodore Business Machines

PET & APPLE II USERS

Tiny Pascal

Plus +
GRAPHICS



The TINY Pascal System turns your APPLE II micro into a 16-bit P-machine. You too can learn the language that is slated to become the successor to BASIC. TINY Pascal offers the following:

- **LINE EDITOR to create, modify and maintain source**
- **COMPILER to produce P-code, the assembly langauage of the P-machine**
- **INTERPRETER to execute the compiled P-code (has TRACE)**
- **Structured programmed constructs: CASE-OF-ELSE, WHILE-DO, IF-THEN-ELSE, REPEAT-UNTIL, FOR-TO/DOWNTO-DO, BEGIN-END, MEM, CONST, VAR ARRAY**

Our new TINY Pascal PLUS+ provides graphics and other builtin functions: GRAPHICS, PLOT, POINT, TEXT, INKEY, ABS AND SQR. The PET version supports double density plotting on 40 column screen giving 80 x 50 plot positions. The APPLE II version supports LORES and for ROM APPLESOFT owners the HIRES graphics plus other features with: COLOR, HGRAPHICS, HCOLOR, HPLOT, PDL and TONE. For those who do not require graphics capabilities, you may still order our original Tiny Pascal package.

TINY Pascal PLUS+ GRAPHICS VERSION·	
PET 32K NEW Roms cassette	$55
PET 32K NEW Roms diskette	$50
APPLE II 32K/48K w/DOS 3.2 or 3.3	$50
TINY Pascal NON-GRAPHICS VERSIONS·	
PET 16K/32K NEW Roms cassette	$40
PET 16K/32K NEW Roms diskette	$35
APPLE II w/ROM Applesoft 32K w/DOS	$35
APPLE II w/RAM Applesoft 48K w/DOS	$35
USER's Manual (refundable with software order)	$10
6502 Assembly Listing of INTERPRETER-graphics	$25
6502 Assembly Listing of INTERPRETER-non graphics	$20

master charge

VISA

FREE postage in U.S. and CANADA. Orders may be prepaid or by bankcard (include card number and expiration date). Michigan residents include 4% state sales tax. Orders accepted via THE SOURCE - CLO852.

ABACUS SOFTWARE
P. O. Box 7211
Grand Rapids, Michigan 49510

CENTERBROOK SOFTWARE DESIGNS

"Learning is Fun"

Remedial Math and Reading
Science, Utilities, and Games

We have over 70 programs in the topics listed above. All programs work in 8k and there are versions for all PETs. Prices for most programs are in the $10 to $20 range and special packages are available. New programs are produced every month and custom programming is available.

SPECIALS
(Until June 15, 1981)

Any 6 Math Programs
normally $90, ***now only $60***

Science Package - biology & chemistry
8 programs sell for $95
now only $50

NEW Programming Pack SCARDIAC-simulates Bell Tel CARDIAC-intro to machine language programming
POGOG - a LOGO-like or turtle graphics language
Regular $50 ***NOW ONLY $20***
Both work on 8K of memory

For a catalog and more information write or call:

Centerbrook Software Designs
98 Emily Drive
Centereach, NY 11720
516-585-2402

VIGIL

INTERACTIVE GRAPHICS/GAME LANGUAGE

FOR THE PET/CBM

VIGIL is an exciting new interactive language for your PET/CBM micro. **VIGIL** - Video Interactive Game Interpretive Language - is an easy to learn graphics and game language that lets you quickly create interactive applications.

- More than 60 powerful commands permit you to easily manipulate graphics figures on the screen
- Double density graphics give you 80 X 50 plot positions on your 40 column PET/CBM
- Large number display capability, access to two event timers and tone generation (if you have ext. speaker)
- Load and save your **VIGIL** programs to cassette or diskette
- Nine interactive programs demonstrate the power of **VIGIL** - Breakout, SpaceWar, AntiAircraft, U.F.O., SpaceBattle, Concentration, Maze, Kaleidoscope & Fortune
- Comprehensive user's manual with complete listings of enclosed programs

VIGIL comes on cassette, or diskette ready to run on any 40 column PET/CBM micro with at least 8K of memory. Specify ROM-set when ordering. 6502 listing of the **VIGIL** Interpreter available separately.

	US & Canada	Foreign
VIGIL FOR Pet/CBM on Cassette or Diskette (w/9 programs)	$35	$40
VIGIL User's Mannual (refundable with software)	$10	$12
VIGIL Interpreter listing (6502 Assembly language)	$25	$30
PET MACHINE LANGUAGE GUIDE	$8	$10

VISA

ABACUS SOFTWARE
P. O. Box 7211
Grand Rapids, Michigan 49510

master charg

Prices include postage. Michigan residents include 4% sales tax. Orders must be prepaid or via bankcard (Mastercard, VISA, Eurocard, Access, etc.). Include card number and expiration date.

(C) 1981 by Roy Wainwright

MICRO SOFTWARE SYSTEMS

P.O. Box 1442, Woodbridge VA 22193

HARDWARE	LIST	MSS
SuperPET	1995	1676
CBM 8096 Upgrade to 8032	500	420
CBM 8032 Computer	1495	1256
PET 4032 Computer	1295	1089
PET 4016 Computer	995	836
VIC-20 Computer with Modulator	325	289
CBM 8050 Disk Drive	1795	1495
CBM 4040 Disk Drive	1295	1088
CBM 2031 Disk Drive	695	584
C2N Cassette Drive	75	69
CBM 4022 Printer	795	669
EPSON MX-80 Printer	649	499
GRAFTRAX 80 graphics ROMs ...	95	75
EPSON MX-80 F/T Printer	749	649
EPSON MX-100 Printer	995	799
IEEE Interface	59	51
ITOH STARWRITER Printer	2074	1695
XYMEC HY-Q 1000 Printer	2885	2499
NEC SPINWRITER 7730 Printer	3195	2695
IEEE Interface & Cable	149	125
TALLY 8024 Printer	1995	1689
DIGI-PLOT PLOTTER	1495	1295
DIGI-PLOT 6-COLOR PLOTTER	1995	1676
CBM 8010 MODEM	279	234
DC HAYES SMARTMODEM	299	259
SMARTMODEM + McTERM software	494	399
ESCON IEEE/Selectric IF & Cable	695	635
ESCON IEEE/Selectric 50/60/70 ..	595	499

SOFTWARE	LIST PRICE		MSS PRICE	
TITLE	CASS	DISK	CASS	DISK
Billboard (8032)	39.95	44.95	33.56	37.76
Disk Librarian	29.95	34.95	25.16	29.36
Word Pro 4 Plus		450.00		378.00
Word Check		195.00		164.00
Visi Calc (CBM)		199.00		168.00
Ozz		395.00		332.00
File Cabinet		69.95		59.00
Create-A-Base		295.00		248.00
JINSAM SYSTEM #8.		795.00		668.00
JINSAM SYSTEM #4.		695.00		584.00
JINSAM SYSTEM #1.		395.00		332.00
MAGIS Business Pkg		2495.00		1995.00
PASCAL (TCL)		295.00		248.00
PASCAL (WI)		75.00		63.00
fullFORTH+		55.00		46.20
Program Toolkit		39.95		35.60
Command-O		79.95		67.20
Matrix Sort ROM		54.95		49.20
Spacemaker II		39.95		35.60

MIN ORDER $25. FREE CATALOG. ADD 5% S&H. VA ADD 4%. VISA/MC OK

The PAPER
Box 460
Livingston Manor, NY 12758

ADDRESS CORRECTION REQUESTED