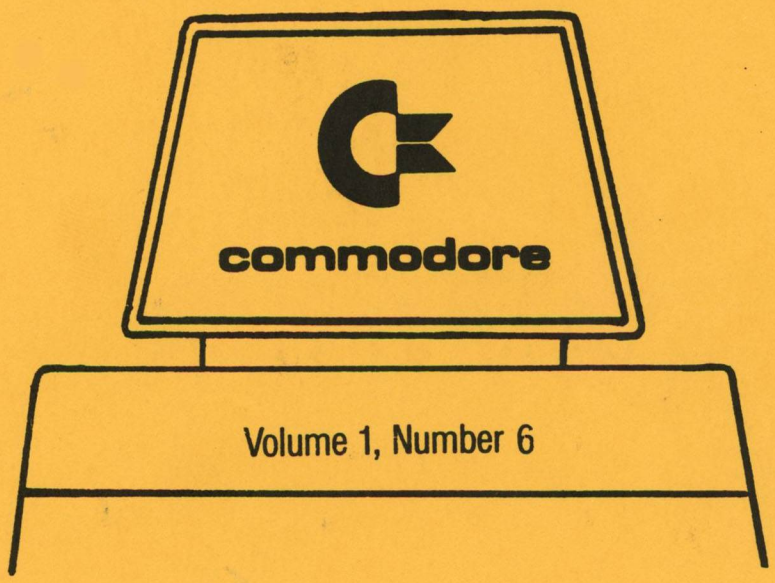
PET
USERS CLUB



NEWSLETTER

COMMODORE BUSINESS MACHINES, INC.

Newsletter Contents

~~~~~~

The Charter of the COMMODORE PET USER CLUB is to provide a method of sharing up to date information, and programs relating to the PET Computer between the many PET owners and Users. Membership charges in the United States and its possessions are $15.00 annually, while subscriptions outside the U.S. are $25.00 yearly.

We would like to publish features from PET Users concerning specific applications, interesting discoveries or even bits worthy of sharing. If you would like to contribute to future NEWSLETTERS, please send your article, letter or comments to: THE EDITOR, COMMODORE U.S. PET USERS CLUB, COMMODORE BUSINESS MACHINES, INC., 3330 SCOTT BLVD., SANTA CLARA, CALIF. 95051.
~~~~~~

Editor Notes

Dear PET User Club Readers:

As you can tell by the size of this NEWSLETTER (60 pages) we have an abundance of information to give to you! It was delayed though to guarantee accurate information.

We have found that some questions PET Users have are quite similar, therefore our Data Exchange Section has been expanded to answer many more of the most commonly asked questions. As promised, the new 8K PET ROMs are available for certain PCB board serial numbers - see our Commodore News Section for specifics.

For those of you who are working with our four part feature program, Break-Even Analysis, its' concluding article leads off the Software Section this month. Let us know the benefits received from this program series documentation; i.e. Would you like to see more? What concepts or programming techniques would you like to read about, and in what level of detail?

After reviewing numerous programs for our Software Applications contest, we have chosen a winner! To discover who this months winner is, or if you would like to participate in future contests please see page 16.

The majority of our Programming Section has been devoted to the Utility Disk operation. Once this base line is established, future application articles will be more meaningful.

If you want the PET to be a cost efficient solution for areas in your business, take a look at our Frequency Counter article in the Peripherals and Attachments Section. Also in our "P & A" Section you will discover a preview of the new CBM Printers AND we've accompanied a sales demonstration print-out!

In rounding out this issue, you'll find a list of our newest Commodore Dealers. Each month scan through the Dealer List to see it any New Dealers are in your area. Remember it's one more source for more service.

The EDITOR

P.S. In Issue 4-5 we have a correction on page 10 in the article titled 4/8K Versus the 16/32K, in the listing. It should read as follows:

line 110 reads FORE should read POKE
line 120 reads SY8 should read SYS826
line 1030 reads 133,202,76,66,2, should read 133,202,76,66,3

On page 2 line "90" should read; 30 GET#1,A$:IF ST 0 THEN 100

On page 9 the CHR$ values- 223,225,173,172 and 188 should be located on the bottom of page 8.

Data Exchange

IN PURSUING BETTER COMMUNICATION WITH OUR USERS, THIS SECTION WILL COVER ANSWERS TO YOUR INQUIRIES NOT COVERED ELSEWHERE IN THIS NEWSLETTER.

Mr. Buz Overbeck of Dallas, TX., will start off this month's Data Exchange with a very common problem.

→ **Q.** I have just received my new 32K PET, (small keyboard/build in cassette model), and find that the majority of program tapes written on 8K PETs will either not run or not load. These include commercial tapes as well, including one from COMMODORE. Could you explain the problem?

A. Your cassette deck may be the problem. If you cannot load the tape on someone else's PET then the tape is bad. Check to see if your cassette is plugged in. If you don't get a 'SEARCHING' message, this is likely the case.

Next, Mr. Richard H. Ball of Seattle, Washington asks:

→ **Q.** When you run a program on the 2023 printer that has graphics, the print-out looks as if you were looking through venetian blinds - in otherwords the printer lines are too far apart. Is there any way to vary the line spacing? If no way now, is there any plans in the far distant future to change this? Is this also the same on the 2022?

Also the unit I saw had the blank paper in a stack on the outside, does the unit have any provisions for paper on the inside?

A. The 2023 printer mechanism has a fixed line spacing, and there is no way to vary it. The 2022 Tractor feed printer does, however, have a programmable line spacing feature. The printout which follows represents normal line spacing on the Model 2023. If line spacing in your printout differs contact your Commodore Dealer for inspection.

```
10 OPEN4,4
20 FOR I=32 TO 95 : A$=A$ + CHR$(I) : NEXT I
30 FOR I=160 TO 223 : B$=B$ + CHR$(I) : NEXT I
40 C$="■" + A$
50 D$="■" + B$
60 E$="■" + A$
70 F$="■" + B$
80 G$="■" + C$
90 H$="■" + D$
```

```
100 PRINT#4,CHR$(1)"□CBM□ 2022 & 2023 □P□RINTER □C□HARACTER □S□ET"
110 PRINT#4
120 PRINT#4,A$
130 PRINT#4,B$
140 PRINT#4,C$
150 PRINT#4,D$
160 PRINT#4,E$
170 PRINT#4,F$
180 PRINT#4,G$
190 PRINT#4,H$
200 CMD4
210 :PRINT:PRINT:PRINT:PRINT
220 LIST
```

In response to your second question; The Printer is designed, like most other commercial printers, to have the paper external to the unit. This allows the User to accommodate any amount of almost any type of paper that he desires.

Finally Mr. Orrin S. Edwards of Mineola, New York, has questions on the version 2 ROMS.

→ Q Will programs with the POKES for data file writing run with the new ROMS, or must the POKES be removed?

A. The version 2 ROMs will support programs with the POKES for data files, but the actual POKE statements should be removed as they are now unnecessary, and will cause problems since the affected addresses have been redefined. (see Issue 3, page 17 of this Newsletter)

→ Q. Will the jump table in the high addresses remain the same in the new ROMS, or will programs utilizing these addresses have to be modified?

A. The Jump table in high ROM is the same as that in the old ROMs so those programs that refer to it should still work.

→ Q. Will the ROMS come with sufficient documentation, including and update to, or a revised users manual?

A. The new Version 2 ROMs do come with a revised 2001 User Manual which includes the changes in the BASIC.

→ Q. Is the Machine Language Monitor in the ROMs the same as the one now on tape?

A. Yes, the Machine Language Monitor in the ROM is the same as on tape, EXCEPT that the command format for the LOAD and SAVE are slightly different. For example:

```
NEW<"NAME",DD
   S "NAME", DD,SSSS,EEEE

DD    Device Number
SSSS  Start Address
EEEE  End Address
```

The register display command has been improved to allow the User to view and modify the RAM IRQ Vector. The old Machine Language Monitor tape will LOAD but will not RUN in a PET with the new ROMs -- but it is not needed since the monitor is already in ROM. In reference to the documentation you should be referring to would be our revised USER MANUAL for the 16/32K.

~~~~~~

BACK TO BASICS FOR THE 2001-8, 16/32K

Many of you have written or phoned in with a number of general questions. In this section,we will try to answer a lot of the more general "wutduzzitdo" and "how do I find out" type of questions that many of you may be wondering about, even if you haven't asked yet. This is not a substitute for our regular letters and questions column, however. If you need a specific piece of information don't hesitate to write.

QUESTION: Will COMMODORE help me design a program or a system for my specific application?

ANSWER: Not directly. Manuals and bulletins are available which can help you, not to mention this NEWSLETTER! Or you may wish to contact your Dealer, in many cases he has Software capabilities or knows who to contact in your area.

QUESTION: How do I get an array of graphics to print on the CRT?

ANSWER: You may be erasing the screen when you try to print CHR$ (146), which is a "clear screen" character.

QUESTION: Can I write my own tape header?

ANSWER: Yes. Just SAVE"FILENAME", then LOAD"FILENAME". Or, to write data files, OPEN 1,1,1,"FILENAME".

QUESTION: Do you have to Fast Forward past the cassette leader before saving a program?

ANSWER: No, you shouldn't. The operating system software provides about 7.5 seconds to move the tape off the leader before starting to record data.

QUESTION: How fast is cassette data storage?

ANSWER: Data rate is approximately 50 characters per second.

QUESTION: What does PET look for on tape when it searches?

ANSWER: The header block on the tape file.
~~~~~~

QUESTION: How many files can be open at one time?

ANSWER: Ten.

QUESTION: Where are the cassette buffers?

ANSWER: Cassette #1 from $027A to $0339

Cassette #2 from $033A to $03FE

These are Hexadecimal

QUESTION: How is End-of-Memory determined by BASIC?

ANSWER: On power-up reset, a checkerboard pattern is written and read back while incrementing a pointer until failure occurs.

QUESTION: How do you delete a line?

ANSWER: Type the line number then press RETURN.

QUESTION: What will happen if I try mixed mode arithmetic?

ANSWER: All arithmetic is performed in floating point. If an operation is performed on an integer, it is first converted to floating point, and on assignment to an integer variable, the result is appropriately truncated or left alone.

QUESTION: Can you program in machine language from BASIC and not use a monitor?

ANSWER: Yes. By using the POKE command, it is possible to load a program into RAM. The process can be automated with a BASIC loader program which contains the bytes of the machine code program in DATA statements. To be safe, POKE into cassette buffer #2.

QUESTION: How is the SYS function used?

ANSWER: The parameter for SYS is a decimal address. This is evaluated and used as a target for JMP instruction. Return to BASIC via RTS.

QUESTION: Where are variables stored, and can they be passed from one program to another?

ANSWER: During program execution, strings are created and stored downward from high end of memory. Integers and real numbers are stored upward from the end of BASIC text. They may be passed to an overlay program if the overlay is less than or equal in size to the program which initiated the LOAD.

QUESTION: Does PET have a SORT function?

ANSWER: No. SORTing must be done by a BASIC program. See Knuth, "The Art of Computer Programming" for a variety of algorithms.

QUESTION: How does PET compare strings?

ANSWER: In alphabetical order according to ASCII code. For example:

"A"<"AA" and "ABCD"<"ABCE"

QUESTION: Is the screen refreshed from a specific 1K of memory?

ANSWER: Yes, starting at $8000.

QUESTION: Can I POKE the locations for cursor control?

ANSWER: Not unless you're willing to risk all sorts of unidentified flying glitches. We do not recommend using POKE to control the cursor. The cursor control keys, and from BASIC.

QUESTION: Can PET be reset without destroying RAM content?

ANSWER: No.

QUESTION: What is the PET's power consumption?

ANSWER: 1.1 amp or 150 watts.

QUESTION: Why is the PET only expandable to 32K RAM?

ANSWER: Because the upper 32K is reserved for O.S., I/O, and ROM. The 6502 can only address 64K.

QUESTION: Why won't my PET load and save my programs:

1. Are you using good tapes?
2. Have you fully rewound the tape before a save or load?
3. Have you recently cleaned and demagnetized the deck heads?
4. If these questions are answered "yes" and PET still won't read tapes, it could be due to poor alignment to the read/record heads. Check with your local dealer.

QUESTION: If the RETURN key stops working in the middle of a program, how can I save myself? Do I HAVE to reset?

ANSWER:
1. If the cursor can be seen, press RETURN.
2. If the cursor can't be seen, press the RUN/STOP key.
3. If neither works, you must reset. Check for possible hardware malfunction. Is the keyboard connector firmly attached to the main board?
4. And, if all else fails, check to be sure you haven't left any tape or printer files open. PET may be sending the RETURN to a file.

~~~~~~
~~~~~~

Commodore News

CBM BUSINESS KEYBOARD

For those of you who own the 16/32B Model CBMs here is some further information about your keyboard and the graphic characters. Included here is a chart listing the graphic symbols, and how to access them.

First enter the following command, POKE 59468,12 and depress [RETURN] - your screen will change from the LOWER CASE/UPPER CASE keyboard mode, to the UPPER CASE/GRAPHIC SYMBOL keyboard mode. In this latter mode, the upper case representation of the key dpressed is displayed. When shifted alphabetical keys (A-Z) are depressed and the associated graphic symbol is displayed; as shown in the attached GRAPHIC SYMBOL TABLE.

The remaining GRAPHIC SYMBOLS, as shown on this table, can only be accessed in the programming mode. For example programs in the CBM USER MANUAL, which use any of these symbols, simply substitute CHR$ (nnn) where 'nnn' is the numeric value assigned to the particuliar symbol.

Whenever you wish to exit this "GRAPHIC SYMBOL" mode, either reset your machine (power off/on) or enter: POKE 59468,14 and depress [RETURN] .

There are three keys on your Business Keyboard which are not described in the User Manual, namely TAB, ESC and REPEAT.

The first two are user programmable keys. That is, a software program could be developed to define a custom usage specially for your application. Without such Software, the CBM's operating system will ignore depression of these keys.

The third key [REPEAT] displays the abreviated PRINT command, "?" for repeated handling of your direct mode operations. For example, if you want to calculate 2+3, you would enter:

[REPEAT] [2] , SHIFTED [+ ;] , [3] [RETURN]

and the result, 5, will display. Even though the shifted [?/] key could also have been used instead of [REPEAT] , the latter is more convenient to use since it operates identically in either shifted or unshifted Keyboard modes.

Finally, when the lines of print are scrolling on the screen, the [←] key is used to slow the rate of scroll to about one line per second. The graphics keyboard uses the [RVS] key for this function.

GRAPHIC SYMBOL REFERENCE TABLE*

GRAPHIC SYMBOL	KEYBOARD ENTRY	GRAPHIC SYMBOL	KEYBOARD ENTRY	GRAPHIC SYMBOL	KEYBOARD ENTRY
♠	A	◧	CHR$(161)	◾	CHR$(187)
◫	B	▬	" 162	◔	" 188
⊟	C	❒	" 163	◰	" 189
⊟	D	❑	" 164	◩	" 190
⊟	E	☐	" 165	◪	" 191
⊟	F	■	" 166	⊟	" 192
◧	G	❒	" 167	⊞	" 219
◫	H	▬	" 168	◧	" 220
◹	I	◩	" 169	◫	" 221
◹	J	◨	" 170	◥	" 223
◪	K	⊞	" 171	π	" 255
☐	L	◾	" 172		
◹	M	◳	" 173		
◪	N	◱	" 174		
☐	O	▬	" 175		
☐	P	◱	" 176		
■	Q	⊞	" 177		
⊟	R	⊞	" 178		
♥	S	⊞	" 179		
◧	T	◧	" 180		
◪	U	◧	" 181		
⊠	V	◨	" 182		
◯	W	❒	" 183		
♣	X	▬	" 184		
◫	Y	▬	" 185		
♦	Z	❑	" 186		

*THESE SYMBOLS ARE AVAILABLE FROM THE KEYBOARD ONCE POKE 59468,12 HAS BEEN EXECUTED.

SHIPMENT OF NEW ROMS

As promised shipment of our new ROMs has begun! Our current shipments to dealers include the ROMs which belong to the PCB Board Numbers 32008 or 320132. Below is a chart depicting what new ROMs should be purchased to replace your old ROMs for said specified boards. ROMs which are not currently available will be shipped to your local COMMODORE Dealer upon their arrival.

Due to an increase in the ROM count, the price per ROM Set has been increased to $89.95.

ROMS for Board #'s 320081 and 320137 will be shipped upon arrival within 4-6 wks.

A	H7	H6	H5	H4	H3	H2	H1
OLD ROMs	901447-08	901447-04	901447-02	901447-06	901447-05	901447-03	901447-09
NEW	901447-26	901447-23	901447-21	901447-25	901447-24	901447-22	901447-20

These ROMS are currently being shipped for board #'s 32008 and 320132

B							
OLD ROMs	6540-018	6540-014	6540-012	6540-016	6540-015	6540-013	6540-019
NEW	6540-026	6540-023	6540-021	6540-025	6540-024	6540-022	6540-020

~~~~~~
~~~~~~

Software

FEATURE PROGRAM

BREAK-EVEN ANALYSIS Part 4 by J. Parsons & C. Westfall

In the last 3 issues of our NEWSLETTER, our feature program has dealt with the BREAK-EVEN ANALYSIS Program, which was listed on page 10 of Volume 1, Issue 2. This month's issue will conclude our discussion and detailed explanation of this program.

Below is a list of the variables and a description of how they are used in the lines to be discussed.

VARIABLE NAME	VALUE OF DESCRIPTION OF ITS FUNCTION
D	Position of cursor within a field
E	Field length
FC=K(0)	Fixed Cost
VC=K(1)	Variable Cost
SP=K(2)	Sale Price
UN=K(3)	Number of Units
GP	Gross Profit
T	Counter
TE	Total number of entries
A$	Home plus 25 cursor down's
C$	40 Spaces
E$(E)	Error messages
S$	The value of a variable (UN,FC,VC,SP) with spaces added to the right to form a string the length of the field.
I%(J,I)	Matrix in which field positions and lengths are stored.

LINE NUMBER	FUNCTION
100	E is set equal to 6 E$(6) is set equal to 38 spaces GOSUB 900 in this case erases any previously displayed error message. Line 900-910 will be discussed later.

LINE NUMBER	FUNCTION
110	This line checks to see if an entry has been made in all four fields. If this condition exists,too much data has been entered. The program is designed to calculate one of the four variables; fixed cost, varible cost, sale price or number of units. If all four are entered no calculation is made. GOTO900 prints the error message and then returns control to line 90 which begins the input process again. In order to check for too much data, the program reused the variable T. T was used as a FOR NEXT counter in lines 2010, 2040 and 2060, and therefore will contain any value from 1 to 50. Each of the possible four entered values, K(0) through K(3), are taken times T. If any one of the values is zero, then T will become zero. So if T is any value other than zero - it means that values have been entered in all four data fields.
120	This line checks for no data. In other words, it checks to see if all four variable fields were left blank. If so, GOTO900 prints the appropriate error message and the RETURN in line 910 transfers control back to line 90 where the input process begins again. In order to check for no data, the variable T is again reused. Line 120 is only executed if T had the value of zero in line 110. It therefore uses T to add together all four variable values. If their sum is zero, then no data has been entered.
130 & 140	These two lines check for too little data; i.e. it makes sure that only one variable was left unentered. I is set equal to zero. It then checks each of the four variables. For each one that is equal to zero, I is increased by one. So if I is greater than 1, then more than one variable had the value of zero. If this condition exists, E is set to Ø and GOTO 900 prints the appropriate error message. The RETURN encountered in

in line 910 transfers control back to line 90 which begins the input routine again.

160

Sets FC = (0) Fixed Cost
Sets VC = (1) Variable Cost
Sets SP = (2) Sale Price.
Sets UN = (3) Number of Units

200

Checks to see if the number of units was equal to zero. If so, it is calculated using the formula:

$$UN = \frac{FC}{SP-VC}$$

210

If Fixed Cost is equal to zero, its value is calculated using:

$$FC = UN\ (SP-VC)$$

220

If the variable cost equals zero, the formula below calculates it:

$$VC = \frac{FC - (UN\ \ SP)}{-UN}$$

230

If the sale price was not entered, it is calculated using:

$$SP = \frac{FC + (UN\ \ VC)}{UN}$$

250

The only value that is calculated in lines 200-230, is the value you left unentered. Therefore, only one variable (i.e. UN, FC VC or SP) are assigned a new value. Line 250 stores this new value in the array K(I) in which the values were originally stored.

260-270

The FOR NEXT loop contained in these lines prints the caluclated values on the screen.

S$ becomes a string equal to the left most 10 characters of a string formed by concatenating your entry plus C$ (40 spaces).

Printing LEFT$(A$, I%(J,Ø) places the cursor on the correct line. SPC(I%(J,1) places the cursor in the right column. It then prints S$ (the entered or calculated value).

280	This line prints the gross profit in reverse field. It uses the same technique used in lines 260-270 to position the cursor.
290	RETURN transfers control to line 90.

900	LEFT$(A$,23) - places the cursor in the correct row SPC(1) - places the cursor in the correct column LEFT$(CR,38) - prints 38 spaces, erasing any previous error message.
910	The cursor is placed in the correct position as in line 900. E$(E) is the appropriate error message.

3000-3010	The FOR NEXT loop in these two lines clears all the fields LEFT$(A$,I%(J,0)) places the cursor in the correct row SPC(I%(J,1) places the cursor in the correct column LEFT$(C$,I%(J,2)) prints X number of spaces- the length of which is specified by I%(J,2) K(J)=Ø resets each variable (UN,FC,VC or SP) to zero
3020	Prints 10 reverse field spaces to 'erase' the gross profit display.
4000	When an insert is entered, line 400 is accessed. This line clears the field from the position of the cursor and all the characters to its right.

Now that you are familiar with the "form" technique of data entry, you have the means to create some very easy to use screen formats for such user oriented data base management systems such as inventory control, payroll, cash flow management and any other business or personal record keeping systems you may wish to develop.

The COMMODORE WORD PROCESSOR is a suprisingly low cost package. It will initially be available in two versions, a standard 8K version which will sell for $24.95, and the expanded 16K disk version which will sell for $99.95. The 8K version supports the cassette for file storage. Both processors are written completely in machine code, affording at least a ten-fold increase in speed over those written in BASIC.

The features common to both processors include:

1. Full cursor control with auto repeat.
2. Any number of tabs.
3. Search for a block of text.
4. The capability to fill in a line (such as a name or address) at the touch of a button.
5. The ability to concatenate text files in memory.
6. Printer format control.
7. Auto indent and right justification margin release.
8. Auto centering of a block of text.
9. The ability to insert or delete a single character or an entire passage.

THE 16K VERSION ALSO INCORPORATES THESE OUTSTANDING FEATURES:

1. The ability to re-locate text at will.
2. Semi-automatic hyphenation: only hyphenates where necessary.
3. Automatic form letter handling, set up the skeleton and list of data,...and watch it go!
4. The ability to append "canned" paragraphs anywhere and everywhere.
5. Alternate text area - used for storing things like form data, canned paragraphs, or simply to have two pages available at once.
6. Complete control of the disk.

These are but some of the marvelous features to be had with the COMMODORE WORD PROCESSING PACKAGE.

The WORD PROCESSOR is easy to use, with some of the keys of the keyboard being re-defined for special functions. There is a built-in auto repeat, which works exactly the way an electric typewriter does.

The print formatter will let you know if there is a format error, and is capable of full margin, and spacing control. If you get into the output or file access modes, and don't wish to, it is easy to return to the editor, without any problem. This is handy, because you can set up the print format in the middle of a paragraph, and pick up where you left off.

Combined with either the 16/32K PET (Graphic Keyboard) or CBM (Typewriter Keyboard), a 2022/2023 Printer, and a 2040 Floppy Disk, comprises a highly powerful word processing system which should maximize any secretary's output.

These two programs are in the final stages of production, and will be available in approximatley 30-60 days. The CBM IS JUST AROUND THE CORNER, YOU DON'T WANT TO RUSH OUT NOW AND GET LESS FOR MORE WITH ANOTHER SYSTEM!!!

~~~~~~

CURSOR: A REVIEW

CURSOR magazine, unlike other computer magazines, is not printed. It comes, once a month, on a cassette tape for the PET computer. Each issue consists of five programs and a couple of printed pages that give some of the background on the programs.

The first program on each tape is called, COVER. The "Cover" is an animated design of some type, which varies from month to month. In last October's issue, the cover displayed an animated keyboard, which played three songs through the CB2 output of the PET. In addition to the "cover", the COVER program also contains the table of contents for that issue.

The programs in each issue consist of some application, and of some game programs. In the past, they have published, among other things, a text editor, a flash card educational program, and a space-war type game of arcade quality (with sound).

A subscription to CURSOR is $33.00 for 12 issues, and $20.00 for 6 issues, with the option of making your subscription retroactive to issue #1. (Note: at last notice they still offered this option, but they may have discontinued it. If you are interested, write them.)

CURSOR MAGAZINE
P.O. BOX 550
GOLETA, CALIF. 93017

The subject of this review is issue #7, Feb. 1979. It contains these programs, each of which will be dealt with **separately.**

COVER	A kaleidoscope diagram
PRICER	A job cost estimation program
SOUND!	A sound effects library
MIND	The popular game of Mastermind (also known as bagels)
~~~~~~

FBALL Football on the PET, Man vs Man, or Man vs PET

PAPER A wallpaper designing program

The COVER program is a nifty display of a kaleidoscope pattern, but it could conceivably have been a little faster in motion. When the contents are displayed, there is evidence of POKEING the screen to display the text. How about using PRINT statements? It would de-hash the screen somewhat.

The PRICER program is designed to aid in estimating job costs. It takes into account numerous costs, such as labor, overhead, profit, and additional expenses. It does toe the line on memory usage. On an 8K PET, with 6 employees, the program takes 5846 bytes plus variable space.

The SOUND! program is one of the "press the button and listen" programs. It contains sounds like birds, flying saucers, police sirens, funeral dirges, and the like.

For those of us who are suckers for punishment, there is a MIND Program which is the CURSOR version of the game of Mastermind. It allows you to set a limit to the number of allowable guesses, and plays a pretty vicious game. The guy that wrote this is truly a programmer with a bizarre twist for great programming. Excellent!

And last, but not lost....for those tried and true Cosell fans, there is the game of FOOTBALL. My only gripe about this program is the numeric codes for plays. I would much rather see a numeric code for each play, with a menu that appears when an invalid command is entered.

In our opinion CURSOR is well worth your subscription consideration.

~~~~~~

Applications

SOFTWARE CONTEST -- WE HAVE A WINNER!!!

In the third issue of the NEWSLETTER we asked interested users to submit programs that would aid the HOUSEHOLD. We are happy to announce the winning home application comes to us from CLIFF COSTA of Highland, New York.

His program, called AUTOMOBILE MILEAGE PROGRAM, is a well documented tape based file management system for automobile fuel cost accounting. It has the ability to perform the following functions:
~~~~~~

1. Enter data from cassette.
2. Enter data from keyboard.
3. Display data on screen.
4. Display graph on screen.
5. Edit data.
6. Store data on tape.
7. Print out mileage data.

The only major fault is that it does not sort the data. This means that if you enter data out of sequence, the out of sequence data will cause all further computations to be meaningless.

To follow is a copy of the print out as generated by the program and the documentation, as written by Mr. Costa.

An explanation for those of you who did not subscribe in time to receive the previous issue of the NEWSLETTER follows. Each month we will be soliciting specific categories of Software. After they have been received they will be evaluated and the "winner" will receive $100.00 worth of free Master Library Software. Or you may wish to compete internally with your local USER GROUP, or compete with another group in your city.

This by no means should stop you from submitting your Application Program just because it doesn't fall within this month's category -send it in, and it could be published on its own merit.

We are now awaiting entries that concern the creation of a "DESK TOP CALENDAR" program as announced in our last NEWSLETTER. The winner of this application shall be announced in our next issue.

The next application for competition should deal with "UTILITIES". The entry deadline date on this is August 6, 1979.

Please send all competing programs to:

THE EDITOR
COMMODORE BUSINESS MACHINES
3330 SCOTT BLVD.
SANTA CLARA, CA 95051

CLIFF

ONCE AGAIN, CONGRATULATIONS RUSSELL, YOU HAVE WON $100.00 WORTH OF FREE SOFTWARE!!!!!

AUTOMOBILE MILEAGE PROGRAM by Cliff Costa

This program will keep track of the dates on which you bought gas, the current mileage on your car odometer, and gallons of gas purchased and its cost. From this information it will calculate your mileage per gallon and total your gas costs. It will display this information on your CRT screen or print it on your printer in the form of a table. It will also display the miles per gallon as a bar graph gasoline costs and operating efficiency.

DIRECTIONS

1. Turn PET on and load program.

2. After the program is loaded the PET screen should show the program title and instruction menu. If it does not, type RUN and hit RETURN. If this fails reload the program.

3. To start the program type 2 and hit RETURN. The screen will display the instruction title INPUT DATA FROM KEYBOARD and will prompt you for the date.
   a. Enter the date of your gasoline purchase in the form 00/00/00 or 00-00-00. For example, January 2, 1979 would be 01-02-79. Hit RETURN after typing in the date.
   b. The screen will prompt you for the odometer reading. Enter up to six digits plus one decimal place. For example 50550.2. Hit the RETURN key.
   c. PET will prompt you for number of gallons of gas. Enter up to two digits plus one decimal place. For example 11.3. Hit RETURN Key.
   d. PET will prompt you for cost of gas. Enter up to two digits plus two decimal places. Fro example 17.43. Hit RETURN key.
   e. Continue in this manner until all of your data has been entered. If you make a mistake continue on, you can correct it later.
   f. After the last entry, type the number symbol(#) for all four entries. This tells the program that you are finished entering data. The program will now calculate all your MPG figures plus total the cost entries. This operation could take up to one minute. When finished the program will return to the instruction menu.

NOTE: If your PET has 8K of memory you may enter up to 30 entries at one time. If you have added memory to your PET you may enter up to 100 entries.

NOTE: If there are no other entries in memory you must enter at least two at this time or you will get a divide by zero error and fall out of the program.

4. Enter instruction number 3 and hit RETURN key to see your data displayed on the screen. If there are any entry errors note the entry number (EN last column on screen). Data will continually be scrolled off the screen. If you wish to stop the data hit any key. The screen will remain fixed until another key is pushed.

5. Enter instruction number 4 to see a graph of your miles per gallon. Note that on bars longer than 13 MPG the date and the actual miles per gallon are printed on the bar. The bar represents the MPG to the nearest integer. If you wish to stop the data hit any key. Hit any key when you are ready to continue.

6. To correct data, enter instruction number 5 and hit RETURN. PET prompts you for the entry number of the incorrect section. Enter the number and hit RETURN. PET will display that entry and ask you if it is the correct entry. If it is correct, type Y for yes. Enter the number of the line for the incorrect entry.

1 for date	3 for gallons
2 for odometer	4 cost of gas

For example: Hit 1 and RETURN. Enter correction 12-03-78, hit RETURN.

PET will respond with - Are there more corrections? Type Y for yes, or N for no. After a N, the program may take up to a minute to return to the instruction Menu.

7. Enter instruction number 6 and hit RETURN to store data on tape. The PET will display the data as it is being outputted to tape. Note that the program computes the MPG to six decimal places even though only two are displayed in the table. The program will return to the instruction menu when all data has been outputted to tape.

8. Type instruction number 8 and hit RETURN to exit program.

9. Type RUN and hit RETURN.

10. Type instruction number 1 and hit RETURN to read data from tape. The program will display the data as it is being inputted from the tape.

11. If you have a printer hooked up to your PET instruction number 7 will print the data in the form of a table. the program assumes that your printer is connected to the PET IEEE 488 BUSS and is wired for address #3. The COMMODORE PET printers meet this criteria.

```
100 DIM A$(5,100):E=1:OPEN1,0,0:A$(5,1)="NONE"
110 PRINT CHR$(147):PRINT TAB(5);"AUTOMOBILE MILEAGE PROGRAM"
120 PRINT TAB(5);"──────────────────────────────":PRINT
130 PRINT TAB(11);"BY CLIFF COSTA":PRINT:PRINT
140 PRINT TAB(5);"1-READ DATA FROM TAPE"
150 PRINT TAB(5);"2-INPUT DATA FROM KEYBOARD"
160 PRINT TAB(5);"3-DISPLAY DATA TABLE ON CRT"
170 PRINT TAB(5);"4-DISPLAY DATA GRAPH ON CRT"
180 PRINT TAB(5);"5-CORRECT DATA"
190 PRINT TAB(5);"6-STORE DATA ON TAPE"
200 PRINT TAB(5);"7-PRINT DATA TABLE ON PRINTER"
210 PRINT TAB(5);"8-END PROGRAM":PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
220 T$(0)="INSTRUCTION NUMBER "
230 PRINT T$(0);"";:INPUT#1,K
240 ON K GOTO 250,390,570,720,840,990,1130,1330
250 PRINT CHR$(147):PRINT TAB(12);"INPUT FROM TAPE"
260 PRINT TAB(12);"──────────────────":PRINT
270 PRINT:PRINT "INSERT DATA TAPE IN RECORDER.":PRINT:PRINT "REWIND TAPE"
280 PRINT:PRINT "HIT ANY KEY WHEN READY!"
290 GET Z$:IF Z$="" THEN 290
300 OPEN 5,1,0,"FILE":INPUT#5,NE:INPUT#5,TGC$:PRINT NE:PRINT TGC$
310 FOR X=1TO(NE-1):FOR Y=1TO 5
320 INPUT#5,A$(Y,X):NEXT Y:FOR Z=1TO5:PRINT X,A$(Z,X):NEXT Z
330 IF ST>0 THEN 350
340 PRINT :NEXT X
350 PRINT "ST=";:PRINT ST:CLOSE 5:PRINT:PRINT "END OF DATA"
360 PRINT "HIT ANY KEY TO CONTINUE"
370 GETZ$:IF Z$="" THEN 370
380 GOTO 110
390 PRINT CHR$(147):PRINT TAB(8);"INPUT DATA FROM KEYBOARD"
400 PRINT TAB(8);"──────────────────────────":PRINT
410 T$(1)="DATE(00/00/00)  ":T$(2)="ODOMETER READING  "
420 T$(3)="GALLONS OF GAS ":T$(4)="COST OF GAS  ":NE=E
430 FOR X=1TO4:PRINT T$(X);"";:INPUT#1,A$(X,E):PRINT:NEXT X:PRINT
440 NE=E
450 IF A$(1,E)="#"THEN 470
460 E=E+1:GOTO 430
470 REM "CALCULATE MPG"
480 FOR X=1TO(NE-2)
490 A=VAL(A$(2,X+1)):B=VAL(A$(2,X)):C=VAL(A$(3,X+1))
500 MD=A-B
510 MPG=MD/C
520 MPG$=STR$(MPG)
530 A$(5,X+1)=MPG$
540 NEXT X
550 C=0:FOR X=1TO NE:B=VAL(A$(4,X)):C=C+B:NEXT X:TGC$=STR$(C)
560 GOTO 110
570 REM "DISPLAY DATA TABLE"
580 PRINT CHR$(147)
590 PRINT TAB(1);"DATE";:PRINTTAB(9);"ODOMETER";:PRINT TAB(19);"GAL.";
600 PRINT TAB(25);"COST";:PRINT TAB(32);"MPG";:PRINT TAB(37);"EN"
610 PRINT TAB(1);"────";:PRINT TAB(9);"─────────";:PRINT TAB(19);"────";
620 PRINT TAB(25);"────";:PRINT TAB(32);"───";:PRINT TAB(37);"──":PRINT
630 GOSUB 1300
640 PRINT A$(1,E);:PRINT TAB(17-LEN(A$(2,E)));A$(2,E);
```

```
650 PRINT TAB(23-LEN(A$(3,E)));A$(3,E);
660 PRINT TAB(30-LEN(A$(4,E)));A$(4,E);
670 G$=A$(5,E):PRINT TAB(31);LEFT$(G$,5);:PRINT TAB(36);E
680 FOR X=1TO200:NEXT X:GOSUB 1270:RETURN
690 PRINT "TOTAL COST OF GAS - ";:PRINT TGC$:PRINT
700 PRINT:PRINT "TOTAL COST OF GAS - ";:PRINT TGC$:PRINT
710 PRINT:PRINT"END OF DATA, HIT ANY KEY TO CONTINUE":GOSUB 1280:GOTO
720 REM "M.P.G. GRAPH"
730 SPACE$="                                                    ":PRINT CHR$(147)
740 PRINT  SPC(12);"MILES PER GALLON"
750 PRINT  SPC(12);"──────────────────"
760 FOR I=2 TO (NE-1)
770 A=VAL(A$(5,I)):A=INT(A)
780 PRINT"■"LEFT$(SPACE$,A)
790 IF A<14 THEN 810
800 PRINT "■"A$(1,I)SPC(1)LEFT$(A$(5,I),5)
810 PRINT:FOR X=1TO200:NEXT X:GOSUB 1270
820 NEXT I
830 PRINT "END OF DATA, HIT ANY KEY TO CONTINUE":GOSUB 1280:GOTO 110
840 PRINT CHR$(147):PRINT TAB(14);"CORRECT DATA"
850 PRINT TAB(14);"────────────":PRINT:PRINT
860 T$(1)="1-DATE (00-00-00)":T$(2)="2-ODOMETER ":T$(3)="3-GALLONS "
870 T$(4)="4-COST OF GAS "
880 T$(5)="ENTRY NUMBER ":PRINT T$(5);"";:INPUT#1,X:PRINT
890 FOR Y=1TO4:PRINT T$(Y),:PRINT A$(Y,X):NEXT Y:PRINT
900 T$(5)="CORRECT ENTRY ? ":PRINT T$(5);"";:INPUT#1,B$:PRINT
910 IF B$ <> "Y" GOTO 880
920 T$(5)="ENTER NUMBER OF INCORRECT ENTRY "
930 PRINT T$(5);"";:INPUT#1,Y:PRINT
940 T$(5)="ENTER CORRECTION ":PRINT T$(5);"";:INPUT#1,A$(Y,X):PRINT
950 T$(5)="ARE THERE MORE CORRECTIONS ?"
960 PRINT T$(5);"";:INPUT#1,B$:PRINT
970 IF B$="Y" GOTO 840
980 IF B$="N" GOTO 470
990 PRINT CHR$(147):PRINT TAB(11)"STORE DATA ON TAPE"
1000 PRINT TAB(11);"──────────────────"
1010 PRINT:PRINT "INSERT REWOUND BLANK TAPE IN RECORDER.":
1020 PRINT:PRINT "HIT ANY KEY WHEN READY!"
1030 GET Z$:IF Z$="" THEN 1030
1040 OPEN5,1,1,"FILE":Z=0
1050 PRINT#5,NE:F$=STR$(NE):GOSUB 1090:PRINT#5,TGC$:F$=TGC$:GOSUB 1090
1060 FOR X=1TO(NE-1):FOR Y=1TO5
1070 PRINT#5,A$(Y,X):F$=A$(Y,X):GOSUB 1090:NEXT Y:NEXT X:CLOSE 5
1080 GOTO 110
1090 PRINT Y,X,F$:Z=Z+LEN(F$)+1:IF Z<192 THEN RETURN
1100 POKE 59411,53:R=TI
1110 IF(TI-R)<16 GOTO 1110
1120 POKE 59411,61:Z=Z-191:RETURN
1130 PRINT CHR$(147):PRINT TAB(6);"PRINT DATA TABLE ON PRINTER"
1140 PRINT TAB(6);"──────────────────────────────"
1150 OPEN 3,4
1160 PRINT#3,"  DATE";
```

```
1170 T=5:GOSUB 1310:PRINT#3,"ODOMETER";
1180 T=5:GOSUB 1310:PRINT#3,"GALLONS";
1190 T=5:GOSUB 1310:PRINT#3,"COST";
1200 T=6:GOSUB 1310:PRINT#3,"M.P.G.":PRINT#3
1210 FOR E=1TO(NE-1)
1220 PRINT#3,A$(1,E);:T=(11-LEN(A$(2,E))):GOSUB 1310:PRINT#3,A$(2,E);
1230 T=(10-LEN(A$(3,E))):GOSUB 1310:PRINT#3,A$(3,E);
1240 T=(12-LEN(A$(4,E))):GOSUB 1310:PRINT#3,A$(4,E);:G$=A$(5,E)
1250 T=(5):GOSUB 1310:PRINT#3,LEFT$(G$,5):NEXT E
1260 CLOSE3:GOTO 110
1270 GET Z$:IF Z$="" THEN RETURN
1280 GET Z$:IF Z$="" THEN 1280
1290 RETURN
1300 FOR E=1TO(NE-1):GOSUB 640:NEXT E:GOTO 700
1310 REM "SUBROUTINE TO TAB TELETYPE"
1320 FOR X=1TOT:PRINT#3,CHR$(160);:NEXT X:T=0:RETURN
1330 END
```

~~~~~~

WHAT ARE YOU DOING WITH YOUR PET?

The following letter is from Mr. J.F. Sudduth of Galt, California. Mr. Sudduth visited us recently and has some great applications for his PET that he uses everyday!

Dear Sir:

This letter is in response to your NEWSLETTER request for new applications of the PET. I have a small business application in the operation of DRY CREEK GOLF COURSE. We are an 18 hole championship golf course with about 50,000 rounds of play per year. The course is about 20 miles south of Sacramento on Interstate 5.

We use an electronic cash register and each day we take the output report from the cash register and feed it into a daily report program on the PET. The resulting daily report is printed out. That report shows daily, month to date, and year to date information on 16 different departments. It provides excellent visibility in our day to day operations.

We write all our checks on the system and maintain a cassette file which is then used in a disbursement program. We also prepare our nightly bank deposit on the PET and feed that information into the disbursement report which in turn gives us a printout on all disbursement transactions and a daily readout on the status of our various bank accounts.

We have separate payrolls for salaried and hourly employees. I have written programs for managing tournaments of various kinds. These programs are used for assignment of pairing, flights and starting times. It is a very efficient means of dealing with a rather tedious job. The programs are also used for scoring after the tournaments have been played.
~~~~~~

Additionally, we use the PET for handling all our business correspondence. We have about 200 tournaments a year and the letters we need for this lend themselves quite readily to automation. As a side comment, this letter is being typed on the system. I had absolutely zero experience in programming when I received my first PET in January 1978. The ensuing hours of study have been fascinating as well as rewarding. I now feel that I can do just about anything I want with the system. Of course, a professional would do these things more efficiently, but I am pleased with my progress.

Sincerely Yours,

J.F. Sudduth
Dry Creek Ranch Golf Club, Inc.

If you'd like to share with other Users some interesting application you've been working on,just drop us a letter and let us know!

~~~~~~

Programming

PET DOS SUPPORT PROGRAM — By R. J. Fairbairn

Now that the COMMODORE 2040 Floppy Disk System is reaching PET owners more support programs are needed. The PET DOS SUPPORT Program is an aid to the 2040 User which humanizes the PET to 2040 interface better than direct mode BASIC statements.

This program consists of two routines; a BASIC driver routine and a machine language routine. The BASIC program calls the machine language which moves the working portion of itself up into high memory. The subroutine then links itself into the CHRGET subroutine in page zero and before returning moves the top of memory pointer down so BASIC will not destroy the working portion. The BASIC program then clears the PET screen and displays an abbreviated set of instructions before executing a NEW command.

Figure A and Figure B are the BASIC and ASSEMBLY Listings of the DOS SUPPORT Program. The programs are entered into the PET as follows. First reset the PET so the memory is initialized, this makes entry of machine code simpler. After the PET has been reset type in the BASIC program exactly as listed in figure A. Then using the machine language monitor enter the object code for the machine language subroutine at $0700 hex. After entry save both routines from the monitor (SA = $0400,EA = $08B8). Finally, using the instructions included in this article test the program to insure correct operation. Good luck and happy computing.
~~~~~~

WARNING: It is advisable to use diskettes that are new or that contain no valuable data during the test phase. This will avoid loss of important data and your time.

The purpose of this program is to aid the CBM or PET 2001 User in operating the 2040 Dual Floppy Disk System. This instruction sheet has been written with the assumption that the reader has a working knowledge of the 2001 series and the 2040.

NOTE: This program has been placed in the public domain but if you would like us to produce a copy for you, send us a blank disk and we'll duplicate the DOS SUPPORT Program on it at no charge. Though, we do ask that you include a self-addressed ,stamped envelope. If you have any comments or suggestions on the following, please refer them to the editor.

The normal method with which the PET communicates with an IEEE Buss device is by the BASIC commands OPEN, PRINT, GET, INPUT and CLOSE. These statements are somewhat verbose in nature and therefore more prone to operator error. There is also the limitation that INPUT and GET cannot be used in direct mode due to shared buffer areas. These conditions are easily handled with the DOS SUPPORT PROGRAM.

DOS SUPPORT PROGRAM may be loaded (saved) as if it were a normal BASIC program. Note should be made of the fact that the 2040 has a special load file name '*' which if used immediatly after power up (reset) executes the following:

1. Initalizes Drive 0
2. Loads the first file on that drive

Thus if the command LOAD"*",8 is executed and the DOS SUPPORT Program is the first directory entry it will be loaded. When the DOS SUPPORT Program is executed it relocates itself up into the highest available RAM memory locations, links into the CHRGET routine and adjusts BASIC's top of memory pointer down. This technique uses about 350 bytes of the Users memory but normal machine operations may proceed without having to reload the DOS SUPPORT Program until such time that a system reset is performed.

The DOS SUPPORT Program functions by capturing the data that the PET operating system passes to BASIC, before the interpreter has a chance to parse it. Thus we can look for Key (escape) characters and process the disk command which follows without the use or knowledge of the BASIC interpreter.

There are four key characters that are recognized by the DOS SUPPORT Program. They will be processed only when they are found in column one of an input line, otherwise a SYNTAX ERROR will occur.

DOS SUPPORT KEY CHARACTERS

@ or > - Passes commands to the Disk.
/ - LOAD's a program.
↑ - LOAD's and RUN's a program.

The greater than symbol when used preceeding a 2040 Disk command passes that command directly to the Floppy Disk System. See the following examples.

```
Thus:
>IØ
is the same as:
PRINT#15,"IØ"
and:
>SØ:FILE1
is equal to:
PRINT#15,"SØ:FILE1"
```

As you can see the > symbol is a substitute for the PRINT#15 statement. Remember that an OPEN statement is required before a PRINT may be executed but no OPEN is required for the DOS SUPPORT Program.

The second function of the > command is the directory list command. As you know the directory of a minidisk can be loaded with a LOAD"$Ø",8. This LOAD will destroy any program you might have in memory. To avoid the destruction of the current program the DOS SUPPORT program prints the directory on the screen.

To avoid possible directory scrolling, you may depress the SPACE key to stop the listing of a directory. Depress any key to continue the listing - or you may depress the RUN/STOP key to stop the directory listing and return to BASIC.

```
>$Ø
```

Means - Display the entire directory of Drive Ø

```
>$1:Q*
```

Means - Display the directory entries of all files on Drive 1 that have names starting with the letter Q.

The third function of the > command is the error channel interrogation feature. The error channel is read by typing a > followed immediately by a RETURN. This is equivilent to the following program segment.

```
10 OPEN 15,8,15
20 INPUT#15,ER,MSG$,DRV,SEC
30?ER",MSG$","DRV","SEC
```

For Users that have the CBM Model Business Keyboard the "@" key may be used in place of the > for key entry convience. This eliminates shifting for this command.

The LOAD / and LOAD-RUN ↑ command characters operate the same as their BASIC counterparts only with a simplified syntax as follows: /WUMPUS

- This command will load the program file WUMPUS. Both drives will be searched if required.

↑1:COPY DISK FILES

-This command will load the program COPY DISK FILES from Drive 1 (if it is there) and execute it.

The following requirements and limitations are placed on the DOS SUPPORT Program User.

1. The DOS SUPPORT commands may only be used in the direct mode.

2. The commands must start in Column 1.

The user may print the directory by using the following commands:

OPEN 4,4: CMD4	: Opens device 4 and changes the primary output device to 4
>$Ø	: Print the directory
PRINT#4 : CLOSE 4	: Return the default output device to the screen and close the file

```
5 SYS2222
10 PRINT"J"TAB(11)"____________________"
20 PRINTTAB(11)"  PET DOS SUPPORT "
30 PRINTTAB(14)"NOW  LOADED
40 PRINTTAB(9)"  COMMANDS FOLLOWING"
50 PRINTTAB(7)"A > OR @  IN COLUMN 1 WILL"
60 PRINTTAB(9)"BE PASSED TO THE DISK."
90 PRINTTAB(7)"CMD        DESCRIPTION"
140 PRINTTAB(7)"$      DIRECTORY BOTH DRIVES
150 PRINTTAB(7)"$0     DIRECTORY DRIVE 0
160 PRINTTAB(7)"$1     DIRECTORY DRIVE 1"
180 PRINTTAB(7)"  ALL 2040 COMMANDS MAY BE
190 PRINTTAB(7)"ENTERED AS IF THEY WERE IN
200 PRINTTAB(7)"A PRINT# STATEMENT.
220 PRINTTAB(11)"  SPECIAL COMMANDS
230 PRINTTAB(7)"/      LOAD A PROGRAM
240 PRINTTAB(7)"↑      RUN  A PROGRAM
250 PRINT"  SPECIAL COMMANDS START IN COL 1 AND
260 PRINT"ARE FOLLOWED BY A 2040 FILENAME.
270 NEW
```

```
PETDOS4.0.

LINE# LOC    CODE          LINE


0001  0000                ;*********************************
0002  0000                ;*
0003  0000                ;*  PET DOS SUPPORT
0004  0000                ;*
0005  0000                ;*     04-27-79
0006  0000                ;*
0007  0000                ;*  BOB FAIRBAIRN
0008  0000                ;*
0009  0000                ;*********************************
0010  0000                ;*
0011  0000                ;* VERSION 3.1 6/14/79
0012  0000                ;*     ADD @ PROMPT FOR BUSINESS
0013  0000                ;*     KEYBOARD. ADD STOP KEY CHECK
0014  0000                ;*     IN DIRECTORY PRINT. ADD
0015  0000                ;*     HALT IN DIRECTORY PRINT
0016  0000                ;*
0017  0000                ;* VERSION 3.2 7/2/79
0018  0000                ;*     FOR (-04) ROM
0019  0000                ;*     WITH LOAD ADDRESS ONE OFF
0020  0000                ;*     BYTE LOW.
0021  0000                ;*
0022  0000                ;* VERSION 3.3 7/2/79
0023  0000                ;*     ADD STACK LOOKUP FOR
0024  0000                ;*     ACTIVATION.
0025  0000                ;*
0026  0000                ;* VERSION 4.0 7/5/79
0027  0000                ;*     ADD CONTROL FOR CMD DURING
0028  0000                ;*     A DIRECTORY LISTING.
0029  0000                ;*


0031  0000                ;
0032  0000                ;BASIC VARIABLES USED
0033  0000                ;
0034  0000                VERCK  =$9D           ;VERIFY FLAG
0035  0000                SAL    =$C7           ;INDIRECT POINTER LO
0036  0000                SAH    =$C8           ;HI
0037  0000                WSW    =$B3           ;UNUSED FLAG (BASIC)
0038  0000                CNTDN  =$BA           ;SAVE AREA
0039  0000                GRBTOP =$5C           ;INDIRECT POINTER
0040  0000                MEMSIZ =$34           ;POINTER TO TOP MEM
0041  0000                TXTPTR =$77           ;POINTER TO BUF
0042  0000                SPERR  =$10           ;EOI ERROR BIT
0043  0000                BUF    =$0200         ;BASIC INPUT BUFFER
0044  0000                SATUS  =$96           ;STATUS BYTE
0045  0000                SA     =$D3           ;SECONDARY ADDRESS
0046  0000                FA     =$D4           ;PRIMARY ADDRESS
0047  0000                LA     =$D2           ;LOGICAL DEVICE #
0048  0000                FNLEN  =$D1           ;FILE NAME LENGTH
0049  0000                FNADR  =$DA           ;FILE NAME ADDRESS
0050  0000                EAL    =$C9           ;END ADDR LO
0051  0000                EAH    =$CA           ;HI
```

```
0052  0000                DFLIO  =$B0             ;DEFAULT OUTPUT DEV.
0053  0000                VARTAB =$2A             ;END OF BASIC PGM.
0054  0000                TMP2   =$FD             ;TEMP VARIABLE
0055  0000                ;
0056  0000                ;PROGRAM VARIABLES
0057  0000                ;
0058  0000                CR     =$0D             ;SYMBOLIC CARRIAGE RETURN
0059  0000                FLAG   =WSW             ;BYTE USED AS A FLAG
0060  0000                PIAK   =$E812           ;KEYBOARD I/O PORT
3061  0000                CMDLN  =CMDEND-CMD      ;LENGTH OF RELCOATE

0063  0000                ;
0064  0000                ;PET ROUTINES USED
0065  0000                ;
0066  0000                LINPRT =$DCD9           ;PRINT LINE #
0067  0000                SPMSG  =$F315           ;SEND A MESSAGE
0068  0000                LD15   =$F322           ;LOAD ROUTINE
0069  0000                TWAIT  =$F8E6           ;WAIT FOR STOP KEY
0070  0000                CHRGET =$70             ;INPUTS CHARACTERS
0071  0000                CHRGOT =$76             ;GET LAST CHAR
0072  0000                NEWSTT =$C6C4           ;NEW STATEMENT EXEC
0073  0000                PRT    =$E3D8           ;PRINT A CHARACTER
0074  0000                LISTN  =$F0BA           ;SEND LISTEN
0075  0000                SECND  =$F128           ;SEND SA
0076  0000                CIOUT  =$F16F           ;SEND CHARACTER
0077  0000                UNLSN  =$F183           ;UN LISTEN
0078  0000                ACPTR  =$F18C           ;GET A CHARCATER
0079  0000                TALK   =$F0B6           ;SEND TALK
0080  0000                OPENI  =$F466           ;OPEN FILE
0081  0000                FCLOSE =$F2AE           ;CLOSE FILE
0082  0000                READY  =$C389           ;REENTER BASIC
0083  0000                RUNC   =$C572           ;CLEAR VARIABLES
0084  0000                LNKPRG =$C442           ;LINK BASIC LINES
0085  0000                UNTLK  =$F17F           ;UN TALK
0086  0000                STXTPT =$C5A7           ;SET START TEXT POINTER
0087  0000                CHKIN  =$F770           ;CHECK IN
0088  0000                CHKOUT =$F7BC           ;CHECK OUT
0089  0000                CLRCHN =$FFCC           ;CLEAR CHANNEL
0090  0000                BASIN  =$FFCF           ;BASIC IN
0091  0000                STOP1  =$F301           ;CHECK FOR STOP KEY
0092  0000                BSOUT  =$FFD2           ;BASIC OUT
0093  0000                FOPEN  =$F524           ;FILE OPEN
0094  0000                LD209  =$F3E6           ;LOAD ERROR

0096  0000                ;
0097  0000                ;WEDGE IN ROUTINE WITH THE
0098  0000                ;COMMAND PARSER AND EXECUTITION
0099  0000                ;
0100  0000                      *=$0700
0101  0700                ;
0102  0700  EA            CMD   NOP               ;THROWN AWAY
0103  0701  E6 77                INC TXTPTR        ;BUMP POINTER
0104  0703  D0 02                BNE WG100
0105  0705  E6 78                INC TXTPTR+1
0106  0707  86 B3        WG100 STX WSW           ;SAVE X IN WSW
```

```
0107  0709  BA                TSX             ;GET STACK POINTER
0108  070A  BD 01 01          LDA $0101,X
0109  070D  C9 9B            CMP #$9B        ;WERE WE CALLED BY MAIN
0110  070F  D0 3A            BNE NOMAIN      ;NO...
0111  0711  BD 02 01          LDA $0102,X     ;MAYBE?
0112  0714  C9 C3            CMP #$C3
0113  0716  D0 33            BNE NOMAIN      ;NOT THERE...
0114  0718  A5 77            LDA TXTPTR      ;FIRST COLUMN
0115  071A  D0 2C            BNE WG997       ;GET OUT NOT FIRST CHR
0116  071C  A5 78            LDA TXTPTR+1
0117  071E  C9 02            CMP #>BUF       ;IN BUFFER?
0118  0720  D0 26            BNE WG997
0119  0722                  ;
0120  0722  A0 00          WG110 LDY #0        ;.Y IS BUF INDEX
0121  0724  84 B3            STY FLAG        ;FLAG SET FOR DIR
0122  0726  B1 77            LDA (TXTPTR),Y
0123  0728  C9 3E            CMP #'>         ;COMMAND PROMPT?
0124  072A  F0 11            BEQ WG115       ;YES...
0125  072C  C9 40            CMP #'@         ;BUSINESS KEYBOARD PRCMPT
0126  072E  F0 0D            BEQ WG115       ;YES...
0127  0730  C8              INY
0128  0731  85 B3            STA FLAG        ;SET FLAG FOR LOAD
0129  0733  C9 2F            CMP #'/         ;LOAD PROMPT
0130  0735  F0 63            BEQ DODIR
0131  0737  C9 5E            CMP #94         ;CHECK FOR ARROW
0132  0739  F0 5F            BEQ DODIR
0133  073B  D0 0B            BNE WG997
0134  073D  C8              WG115 INY
0135  073E  B1 77            LDA (TXTPTR),Y
0136  0740  F0 32            BEQ RDERR       ;READ ERROR CHANNEL
0137  0742  C9 24            CMP #'$         ;DIRECTORY?
0138  0744  F0 54            BEQ DODIR       ;YES
0139  0746  D0 08            BNE NOTDIR
0140  0748  4C 76 00       WG997 JMP CHRGOT
0141  074B  A6 B3          NOMAIN LDX WSW      ;RESTORE .X AND
0142  074D  4C 76 00            JMP CHRGOT     ;RETURN TO CHRGOT

0144  0750                  ;
0145  0750                  ; SEND COMMAND TO DISK
0146  0750                  ;
0147  0750  A9 08          NOTDIR LDA #8       ;GET DEVICE ADDRESS
0148  0752  85 D4            STA FA
0149  0754  A9 6F            LDA #$6F        ;SECONDARY ADDRESS 15
0150  0756  85 D3            STA SA
0151  0758  20 BA F0          JSR LISTN
0152  075B  A5 D3            LDA SA
0153  075D  20 28 F1          JSR SECND       ;SEND SECONDARY ADDR
0154  0760  E6 77          BUMP  INC TXTPTR
0155  0762  A0 00            LDY #0          ;INDEX=0
0156  0764  B1 77            LDA (TXTPTR),Y  ;GET THE FIRST CHARACTER
0157  0766  F0 06            BEQ WG120       ;ZERO IS LAST CHAR
0158  0768  20 6F F1          JSR CIOUT       ;SEND THE CHAR
0159  076B  B8                CLV
0160  076C  50 F2            BVC BUMP        ;MORE
0161  076E                  ;
0162  076E  20 83 F1       WG120 JSR UNLSN     ;UN LISTEN
```

```
0163  0771  B8             CLV
0164  0772  50 23          BVC WG998
0165  0774                 ;
0166  0774                 ; READ THE ERROR CHANNEL
0167  0774                 ;
0168  0774  84 77    RDERR  STY TXTPTR      ;FIX POINTER
0169  0776  A9 08          LDA #8           ;SET FA
0170  0778  85 D4          STA FA
0171  077A  20 B6 F0       JSR TALK
0172  077D  A9 6F          LDA #$6F         ;COMMAND CHANNEL SA
0173  077F  85 D3          STA SA
0174  0781  20 28 F1       JSR SECND        ;SEND SA
0175  0784  20 8C F1  WG140 JSR ACPTR       ;GET BYTE FROM DISK
0176  0787  C9 0D          CMP #CR
0177  0789  F0 06          BEQ WG130
0178  078B  20 D8 E3       JSR PRT          ;PRINT BYTE TO SCREEN
0179  078E  B8             CLV
0180  078F  50 F3          BVC WG140        ;LOOP FOR MORE
0181  0791  20 D8 E3  WG130 JSR PRT          ;PRINT CR
0182  0794  20 7F F1       JSR UNTLK        ;UN TALK
0183  0797  4C 76 00  WG998 JMP CHRGOT       ;DONE WITH CMD

0185  079A                 ;
0186  079A                 ;PRINT THE DIRECTORY
0187  079A                 ;
0188  079A  C8      DODIR  INY              ;GET LENGTH OF CMD
0189  079B  B1 77          LDA (TXTPTR),Y
0190  079D  D0 FB          BNE DODIR
0191  079F  88             DEY
0192  07A0  84 D1          STY FNLEN        ;SET LENGTH (-1)
0193  07A2  A9 01          LDA #<BUF+1      ;FILE NAME ADDRESS
0194  07A4  85 DA          STA FNADR
0195  07A6  A9 02          LDA #>BUF
0196  07A8  85 DB          STA FNADR+1
0197  07AA  A9 08          LDA #8           ;DEVICE ADDRESS
0198  07AC  85 D4          STA FA
0199  07AE  A5 B3          LDA FLAG         ; 0 MEANS DIR
0200  07B0  D0 53          BNE LOADB        ;DO A LOAD
0201  07B2  A5 D2          LDA LA           ;SAVE LA
0202  07B4  85 B3          STA WSW
0203  07B6  A5 B0          LDA DFLTO        ;SAVE DFLTO
0204  07B8  85 BA          STA CNTDN
0205  07BA  A9 60          LDA #$60         ;SECONDARY ADDR
0206  07BC  85 D3          STA SA
0207  07BE  A9 0E          LDA #14          ;OPEN THE FILE
0208  07C0  85 D2          STA LA
0209  07C2  20 83 F1       JSR UNLSN        ;DON'T LISTEN TO FLOPPY
0210  07C5  20 24 F5       JSR FOPEN
0211  07C8  A9 00          LDA #0
0212  07CA  85 96          STA SATUS        ;SET STATUS TO 0
0213  07CC  A0 03          LDY #$03         ;LOOP THREE TIMES
```

```
0215  07CE  84 D1        WG220  STY FNLEN        ;SAVE NEW COUNT
0216  07D0  A2 0E               LDX #14         ;DISK CHANNEL
0217  07D2  20 70 F7            JSR CHKIN
0218  07D5  20 CF FF            JSR BASIN
0219  07D8  85 FD               STA TMP2
0220  07DA  A4 96               LDY SATUS       ;CHECK STATUS
0221  07DC  D0 29               BNE WG235B      ;BAD STATUS
0222  07DE  20 CF FF            JSR BASIN
0223  07E1  85 FE               STA TMP2+1
0224  07E3  A4 96               LDY SATUS       ;CHECK STATUS
0225  07E5  D0 20               BNE WG235B
0226  07E7  A4 D1               LDY FNLEN       ;MORE TO DO?
0227  07E9  88                  DEY
0228  07EA  D0 E2               BNE WG220       ;NOT DONE YET
0229  07EC  20 CC FF            JSR CLRCHN      ;CLEAR CHANNEL
0230  07EF  A6 BA               LDX CNTDN       ;CHECK DFLTO FOR SCREEN
0231  07F1  E0 03               CPX #3
0232  07F3  F0 05               BEQ *+7
0233  07F5  A6 B3               LDX WSW         ;OPEN THE PRINT CHANNEL
0234  07F7  20 BC F7            JSR CHKOUT
0235  07FA  A6 FD               LDX TMP2
0236  07FC  A5 FE               LDA TMP2+1
0237  07FE  20 D9 DC            JSR LINPRT      ;PRINT LINE NUMBER
0238  0801  A9 20               LDA #'          ;PRINT A SPACE
0239  0803  D0 06               BNE SKIPB       ;SKIP OVER BRANCHES
0240  0805  D0 6C        LOADB  BNE LOAD        ;(JMP)
0241  0807  D0 5D        WG235B BNE WG230       ;(JMP)
0242  0809  D0 C3        WG220B BNE WG220       ;(JMP)
0243  080B  20 D2 FF     SKIPB  JSR BSOUT
0244  080E  20 CC FF            JSR CLRCHN
0245  0811  A2 0E        WG250  LDX #14         ;DISK CHANNEL
0246  0813  20 70 F7            JSR CHKIN
0247  0816  20 CF FF            JSR BASIN
0248  0819  48                  PHA
0249  081A  20 CC FF            JSR CLRCHN
0250  081D  68                  PLA
0251  081E  A6 96               LDX SATUS
0252  0820  D0 44               BNE WG230       ;BAD
0253  0822  C9 00               CMP #0          ;EOL
0254  0824  F0 26               BEQ WG240
0255  0826  A6 BA               LDX CNTDN       ;CHECK DFLTO FOR SCREEN
0256  0828  E0 03               CPX #3
0257  082A  F0 05               BEQ *+7
0258  082C  A6 B3               LDX WSW
0259  082E  20 BC F7            JSR CHKOUT
0260  0831  20 D2 FF            JSR BSOUT
0261  0834  20 CC FF            JSR CLRCHN
0262  0837                     ;
0263  0837                     ;CHECK FOR STOP KEY AND PAUSE
3264  0837                     ;
0265  0837  20 01 F3            JSR STOP1       ;STOP KEY
0266  083A  F0 2A               BEQ WG230       ;YES...
0267  083C  20 E4 FF            JSR $FFE4       ;GET A CHAR FROM KEYBOARD
0268  083F  F0 D0               BEQ WG250       ;NOTHING...
0269  0841  C9 20               CMP #$20        ;SPACE BAR?
```

```
0270  0843  D0 CC           BNE WG250       ;NO...
0271  0845  20 E4 FF  WG255  JSR $FFE4       ;ANY KEY STARTS
0272  0848  F0 FB           BEQ WG255
0273  084A  D0 C5           BNE WG250       ;(JMP)
0274  084C                 ;
0275  084C  A9 0D     WG240  LDA #CR
0276  084E  A6 BA           LDX CNTDN       ;CHECK DFLTO FOR SCREEN
0277  0850  E0 03           CPX #3
0278  0852  F0 05           BEQ *+7
0279  0854  A6 B3           LDX WSW
0280  0856  20 BC F7        JSR CHKOUT
0281  0859  20 D2 FF        JSR BSOUT
0282  085C  20 CC FF        JSR CLRCHN
0283  085F  20 83 F1        JSR UNLSN
0284  0862  A0 02           LDY #$02        ; DO TWICE
0285  0864  D0 A3           BNE WG220B
0286  0866                 ;
0287  0866                 ;CLOSE FLOPPY AND RETURN
0288  0866                 ;
0289  0866  20 CC FF  WG230  JSR CLRCHN
0290  0869  A9 0E           LDA #14         ;CLOSE FLOPPY
0291  086B  20 AE F2        JSR FCLOSE
0292  086E  68              PLA             ;CLEAN UP THE STACK
0293  086F  68              PLA
0294  0870  4C 89 C3        JMP READY       ;RETURN "READY"

0296  0873                 ;
0297  0873                 ; LOAD A FILE
0298  0873                 ;
0299  0873  A9 00     LOAD   LDA #0
0300  0875  85 96           STA SATUS       ;CLEAR STATUS
0301  0877  85 9D           STA VERCK       ;LOAD NOT VERIFY
0302  0879  20 22 F3        JSR LD15        ;LOAD A PROGRAM
0303  087C  A5 96           LDA SATUS
0304  087E  29 10           AND #SPERR      ;CHECK STATUS (EOI OK)
0305  0880  D0 28           BNE LDERR
0306  0882  AD 84 F3        LDA $F384       ;CHECK FOR (-04) ROM
0307  0885  30 06           BMI LOAD1       ;NOT (-04)....
0308  0887  E6 C9           INC EAL         ;FIX THE LOAD (-04) ROM
0309  0889  D0 02           BNE LOAD1
0310  088B  E6 CA           INC EAH
0311  088D  A5 CA     LOAD1 LDA EAH         ;SET BASIC'S POINTERS
0312  088F  85 2B           STA VARTAB+1
0313  0891  A5 C9           LDA EAL
0314  0893  85 2A           STA VARTAB
0315  0895  20 72 C5        JSR RUNC        ;FIX POINTERS
0316  0898  20 42 C4        JSR LNKPRG      ;FIX LINKS
0317  089B  A5 B3           LDA FLAG        ;CHECK FOR LOAD OR RUN
0318  089D  C9 2F           CMP #'/         ;LOAD ?
0319  089F  D0 03           BNE WG300       ;NO...
0320  08A1  4C 89 C3        JMP READY       ;LOAD RETURN TO BASIC
0321  08A4  20 A7 C5  WG300  JSR STXTPT      ;SET TXTPTR FOR RUN
0322  08A7  4C C4 C6        JMP NEWSTT      ;RUN PROGRAM
0323  08AA  4C E6 F3  LDERR  JMP LD209       ;PRINT "LOAD ERROR"
0324  08AD                 CMDEND
```

```
0326  08AD                    ;
0327  08AD                    ;THIS ROUTINE POKES TOP OF MEMORY
0328  08AD                    ;DOWN RELOCATES THE PARSER AND
0329  08AD                    ;SETS THE WEDGE
0330  08AD                    ;
0331  08AD  A5 34             POKE    LDA MEMSIZ      ;POKE TOP DOWN
0332  08AF  18                        CLC             ;MINUS ONE
0333  08B0  E9 AD                     SBC #<CMDLN
0334  08B2  85 34                     STA MEMSIZ
0335  08B4  A5 35                     LDA MEMSIZ+1
0336  08B6  E9 01                     SBC #>CMDLN
0337  08B8  85 35                     STA MEMSIZ+1
0338  08BA                    ;
0339  08BA                    ;MOVE THE CODE
0340  08BA                    ;
0341  08BA  A0 01             MOVE    LDY #$01        ;SET UP FROM ADDR
0342  08BC  A9 00                     LDA #<CMD
0343  08BE  85 C7                     STA SAL
0344  08C0  A9 07                     LDA #>CMD
0345  08C2  85 C8                     STA SAH
0346  08C4  A5 34                     LDA MEMSIZ      ;SET UP TO ADDR
0347  08C6  85 5C                     STA GRBTOP
0348  08C8  A5 35                     LDA MEMSIZ+1
0349  08CA  85 5D                     STA GRBTOP+1
0350  08CC  B1 C7             MOV1    LDA (SAL),Y     ;RELOCATE
0351  08CE  91 5C                     STA (GRBTOP),Y
0352  08D0  C8                        INY
0353  08D1  D0 F9                     BNE MOV1
0354  08D3  E6 5D                     INC GRBTOP+1
0355  08D5  E6 C8                     INC SAH
0356  08D7  A5 C8                     LDA SAH
0357  08D9  C9 08                     CMP #>CMDEND
0358  08DB  F0 02                     BEQ MOV2
0359  08DD  B0 04                     BCS WEDGE
0360  08DF  A0 00             MOV2    LDY #0
0361  08E1  F0 E9                     BEQ MOV1
0362  08E3                    ;
0363  08E3                    ;WEDGE INTO BASIC
0364  08E3                    ;
0365  08E3  A9 4C             WEDGE   LDA #$4C        ;JUMP INSTRUCTION
0366  08E5  85 70                     STA CHRGET
0367  08E7  A4 34                     LDY MEMSIZ
0368  08E9  A6 35                     LDX MEMSIZ+1
0369  08EB  C8                        INY
0370  08EC  D0 01                     BNE WEDGE1
0371  08EE  E8                        INX
0372  08EF  84 71             WEDGE1  STY CHRGET+1
0373  08F1  86 72                     STX CHRGET+2
0374  08F3  60                        RTS
0375  08F4                          .END

ERRORS = 0000
```

SYMBOL TABLE

SYMBOL	VALUE						
ACPTR	F18C	BASIN	FFCF	BSOUT	FFD2	BUF	0200
BUMP	0760	CHKIN	F770	CHKOUT	F7BC	CHRGET	0070
CHRGOT	0076	CIOUT	F16F	CLRCHN	FFCC	CMD	0700
CMDEND	08AD	CMDLN	01AD	CNTDN	00BA	CR	000D
DFLTO	00B0	DODIR	079A	EAH	00CA	EAL	00C9
FA	00D4	FCLOSE	F2AE	FLAG	00B3	FNADR	00DA
FNLEN	00D1	FOPEN	F524	GRBTOP	005C	LA	00D2
LD15	F322	LD209	F3E6	LDERR	08AA	LINPRT	DCD9
LISTN	F0BA	LNKPRG	C442	LOAD	0873	LOAD1	088D
LOADB	0805	MEMSIZ	0034	MOV1	08CC	MOV2	08DF
MOVE	08BA	NEWSTT	C6C4	NOMAIN	074B	NOTDIR	0750
OPENI	F466	PIAK	E812	POKE	08AD	PRT	E3D8
RDERR	0774	READY	C389	RUNC	C572	SA	00D3
SAH	00C8	SAL	00C7	SATUS	0096	SECND	F128
SKIPB	080B	SPERR	0010	SPMSG	F315	STOP1	F301
STXTPT	C5A7	TALK	F0B6	TMP2	00FD	TWAIT	F8E6
TXTPTR	0077	UNLSN	F183	UNTLK	F17F	VARTAB	002A
VERCK	009D	WEDGE	08E3	WEDGE1	08EF	WG100	0707
WG110	0722	WG115	073D	WG120	076E	WG130	0791
WG140	0784	WG220	07CE	WG220B	0809	WG230	0866
WG235B	0807	WG240	084C	WG250	0811	WG255	0845
WG300	08A4	WG997	0748	WG998	0797	WSW	00B3

END OF ASSEMBLY

~~~~~~

RANDOM (DIRECT) ACCESS by S. Patterson

This month we will describe how to implement Direct Access on your 2040 Disk. The advantage of Direct Access is that to get to the middle of a data file, you can go right there instead of wading through from the beginning depending on the file size.

The following random file example is built upon a relative record scheme. It is meant to be used in demonstrating the block access commands. Notice that the U1 and U2 commands are used rather that the B-R and B-W commands. Since more than one record is stored in a block, it is necessary to manage end of record pointers in BASIC. A smaller application may take advantage of the latter commands.

This example provides single record access through BASIC programming. It is possible to add data sorts and searches as well as additional key files.

All programming below line 2000 is mostly relative record access. The field accessing routines left-justify binary and alpha fields and right-justify numeric fields. A real situation program might want to generate an error message to the operator or take corrective action (such as rounding numbers to fit the field).
~~~~~~

Record size, including field markers, must be less than 254 characters. Field sizes are restricted to 80, because of the BASIC INPUT# statement used.

Two sequential files are used to support the random access file. Each bear the name of the file name given in the CREATE file code (1100-1180) plus a six character extension. The primary file name must be ten or less characters. All file names less than ten are padded with spaces. The two files are:

FILENAME .DESCR
FILENAME .KEY01

The descriptor file contains information on the record structure and locations. The primary key file is made up of the data from the first field in the record and the relative record number. The example allows the random records to reside on a separate diskette from the sequential support files. This provides added room for random data. The OPEN code (1200-1275) requires the disk ID of the random disk for comparision.

NOTE: If you would like COMMODORE to reproduce the RANDOM Program for you, just send us a blank disk along with a self-addressed stamped envelope.

DIRECT ACCESS EXAMPLE

TO CREATE A FILE:

1. LOAD and RUN RAND 1.0
   a. Insert disk in drive 0.
   b. Open command channel, and initialize disk by typing the following:

      OPEN 15,8,15,"10" RETURN

   c. LOAD RAND 1.0 by typing:

      LOAD"0:RAND1.0",8 RETURN

   d. Insert a blank diskette into drive 1, and type:

      PRINT#15,"N1:COMMODORE,CS" RETURN

   e. Type RUN RETURN

2. The PET will ask "DO YOU WISH TO CREATE A FILE"? Type Y RETURN
3. The PET will display "RANDOM FILE NAME"? Enter the file name, i.e. PHONE LIST RETURN

4. The PET will display "KEY FILE DRIVE NUMBER"?
   Enter 1 [RETURN]
5. The PET displays "RANDOM FILE NUMBER"?
   Enter 1 [RETURN]
6. The PET will ask "ENTER ID OF RANDOM DISK"?
   Type CS [RETURN]
7. The PET displays "NUMBER OF RECORDS"?
   This is the MAXIMUM number of records the file can contain. For this example, enter 10 [RETURN]
8. The PET asks "NUMBER OF FIELDS PER RECORD"?
   This is the number of 'items' each field contains, for this example, enter 4 [RETURN]
9. The PET displays "INPUT FIELD NAME, FIELD SIZE, FIELD TYPE

   TYPES: 0=BINARY, 1=NUMERIC, 2=ALPHA
   FIELD 1?" enter NAME,202 [RETURN]
   FIELD 2?" enter PHONE,15,2 [RETURN]
   FIELD 3?" enter ADDRESS,40,2 [RETURN]
   FIELD 4?" enter COMMENTS,40,2 [RETURN]

TO ADD A RECORD

10. The PET will display "WHOSE RECORD DO YOU WISH TO SEE"?
    Press [RETURN]
11. The PET will display "**** ADD RECORD****"NAME"? enter the desired name, for example, COMMODORE [RETURN]

    PHONE"? enter the phone number, 408-727-1130 [RETURN]

    ADDRESS"? enter the address, for example 3330 SCOTT BLVD SANTA CLARA CA. 95051 [RETURN]

    COMMENTS"? enter a comment, for example DK COMPUTERS [RETURN]

TO SEE A RECORD

12. The PET will display "WHOSE RECORD DO YOU WISH TO SEE"?
    Enter COMMODORE [RETURN]

TO CHANGE A RECORD

13. After the PET has displayed the record, it asks "ANY MODS"?
    Enter YES [RETURN]
14. The PET asks "WHICH FIELD"?
    Enter the number of the field you wish to change, for example, 4 [RETURN]
15. The PET displays that field, for example "MAKES PETS" ? enter the correct contents of the field, for example US HEADQUARTERS [RETURN]

16. The PET will again ask if there are "ANY MODS". If the record is correct, type NO [RETURN]

GETTING THE DIRECTORY OF LISTINGS

17. The PET displays "WHOSE RECORD DO YOU WISH TO SEE"?
    Type /DIR [RETURN]
    The PET displays the directory, in this example, "COMMODORE"

ENDING THE PROGRAM

18. The PET displays "WHOSE RECORD DO YOU WISH TO SEE"?.
    Type // [RETURN], and the program ends.

RANDOM 1.00

```
1 REM RANDOM 1.0
2 REM SUBROUTINES TO MANAGE RANDOM ACCESS FILES
3 REM  VARIABLES ARE SET FROM DATA OF DESCRIPTOR FILE & KEY LIST FILES...
4 REM ...DEFINED BY USER PROGRAM
5 REM VARIABLES SHOULD REFLECT DESIRED FILE STRUCTURE
6 REM ALL FUNCTIONS ACT UPON THE VARIABLES DEFINED BELOW
10 REM
11 REM ************************
12 REM
13 POKE1022,128:REM TURN DOS SUPPORT 3.1 OFF
15 M$=CHR$(13):REM FIELD MARKER
16 SP$="                                                  "+"":REM SPACE FOR PADDING
20 C0=2:        REM DIRECT CHANNEL
21 C1=3:        REM SEQUENTIAL CHANNEL
25 CC=15:       REM COMMAND CHANNEL
30 D=0:         REM CURRENT DRIVE #
31 T=0:         REM CURRENT TRACK #
32 S=0:         REM CURRENT SECTOR #
35 DD=0:        REM DESCRIPTOR DRIVE #
36 RD=0:        REM RANDOM DRIVE #
40 ID$="":      REM RANDOM DISK ID
45 NR=0:        REM # RECORDS IN R-FILE
46 CR=0:        REM CURRENT RECORD #
47 FR=0:        REM 1ST FREE RECORD UNUSED
50 NF=0:        REM # FIELDS IN RECORD
51 CF=0:        REM CURRENT FIELD #
55 RB=0:        REM # RECORDS PER BLOCK
56 RS=0:        REM RECORD SIZE IN BYTES
60 NB=0:        REM # BLOCKS IN R-FILE
65 E=0:         REM ERROR FLAG, OK =0
66 REM EN$,EM$,ET$,ES$,ET,ES  ERROR CHANNEL VARIABLES
70 EP=.5/256:   REM INTEGER CORRECTION
75 AS=0:        REM INDEX ARRAY ADDRESSING STRATEGY
76 REM AS=0: USE ARRAY INDEX; AS=1: T&S ARE SET, CR= RECORD OFFSET IN BLOCK
90 REM "A" VARIABLES ARE TEMPORARY
95 DN=8:OPENCC,DN,CC:  REM DN= DEVICE NUMBER
98 GOTO2000:    REM START OF USER PROGRAM
99 REM
100 REM ************************
```

```
101 REM RANDOM FILE DIMENSION ROUTINE
102 REM  1ST SET NR, NF & NB
103 REM
105 GOSUB150
110 IFFP%=-1THENRETURN
111 FP%=-1
115 DIM FS%(NF) :REM FIELD SIZE
120 DIM FP%(NF) :REM FIELD POSITION
125 REM          FP%(I)= SUM [FS%(I-1)]
130 DIM FT%(NF) :REM FIELD TYPE: 0:BINARY, 1:NUMERIC, 2:ALPHA
135 DIM FH$(NF) :REM FIELD HEADING
140 DIM F$(NF)  :REM FIELD ARGS-ALPHA,BINARY
145 DIM F(NF)   :REM FIELD ARGS-NUMERIC
146 RETURN
150 IFIT%=-1THENRETURN
151 IT%=-1
155 DIM IT%(NB) :REM TRACK INDEX ARRAY
160 DIM IS%(NB) :REM SECTOR INDEX ARRAY
165 DIM K1$(NR) :REM PRIMARY KEY VALUE
170 DIM RR%(NR) :REM RELATIVE RECORD LIST PER KEY
175 RETURN
200 REM ***************************
201 REM UPDATE RECORD, CR
202 REM
205 GOSUB900
210 PRINT#CC,"U1:"C0;D;T;S
215 PRINT#CC,"B-P:"C0;RP
220 FORCF=1TONF
225 GOSUB500
230 NEXTCF
235 PRINT#CC,"U2:"C0;D;T;S
240 GOSUB1000:IFETHEN1900
245 RETURN
300 REM ***************************
301 REM READ RECORD, CR
302 REM
305 GOSUB900
310 PRINT#CC,"U1:"C0;D;T;S
315 PRINT#CC,"B-P:"C0;RP
320 GOSUB1000:IFETHEN1900
325 FORCF=1TONF
330 GOSUB600
335 NEXTCF
340 RETURN
400 REM ***************************
401 REM UPDATE FIELD(CF) OF RECORD CR, SINGLE FIELD UPDATE
402 REM
405 GOSUB900
410 PRINT#CC,"U1:"C0;D;T;S
415 GOSUB1000:IFETHEN1900
420 PRINT#CC,"B-P:"C0;FP%(CF)+RP
425 GOSUB500 :REM UPDATE FIELD
430 PRINT#CC,"U2:"C0;D;T;S
435 GOSUB1000:IFETHEN1900
440 RETURN
450 REM ***************************
```

```
451 REM READ FIELD(CF) OF RECORD CR, SINGLE FIELD READ
452 REM
455 GOSUB900
460 PRINT#CC,"U1:"C0;D;T;S
465 GOSUB1000:IFETHEN1900
470 PRINT#CC,"B-P:"C0;FP%(CF)+RP
475 GOSUB600 :REM READ FIELD
480 RETURN
500 REM ***************************
501 REM UPDATE FIELD(CF), B-P IS SET
502 REM
510 IFFT%(CF)<>1THEN520
515 A$=RIGHT$(SP$+STR$(F(CF)),FS%(CF)):GOTO530
520 A$=LEFT$(F$(CF)+SP$,FS%(CF))
530 PRINT#C0,A$;M$;
535 RETURN
600 REM ***************************
601 REM READ FIELD(CF), B-P IS SET
602 REM
610 IF FT%(CF) THEN645
615 A1$=""
620 FORJ=1TOFS%(CF)
625 GET#C0,A$:IFA$=""THENA$=CHR$(0)
630 A1$=A1$+A$
635 NEXT:F$(CF)=A1$
640 GET#C0,A$:RETURN
645 INPUT#C0,F$(CF)
650 IFFT%(CF)<>1THEN RETURN
655 F(CF)=VAL(F$(CF)):RETURN
700 REM ***************************
701 REM ALLOCATE ONE BLOCK, T & S =REQUESTED TRACK & SECTOR
702 REM  RETURNED T & S ARE ALLOCATED VALUES  (T=18 IS SKIPPED)
703 REM
710 GOSUB800:IFETHEN1900: REM CHECK T & S
715 PRINT#CC,"B-A:"D;T;S
720 INPUT#CC,EN,EM$,ET,ES
725 IFEN=0THENRETURN
730 IFEN<>65THEN1900
735 IFET=18THENT=19:S=0:GOTO715
736 T=ET:S=ES
740 GOTO715
750 REM ***************************
751 REM FREE ONE BLOCK, T & S = TRACK & SECTOR
752 REM
760 GOSUB800:IFETHEN1900: REM CHECK T & S
770 PRINT#CC,"B-F:"D;T;S
780 INPUT#CC,EN,EM$,ET,ES
785 IFEN=0THENRETURN
790 GOTO1900
800 REM ***************************
801 REM CHECK MAX SECTOR
802 REM
810 IFT>35THEN1900
820 E=0:IFT=0THEN=40:GOTO1900
840 A3=16:IFT>30THEN880
850 A3=17:IFT>24THEN880
860 A3=19:IFT>17THEN880
870 A3=20
```

```
880 IFS>A3THEN1900
890 RETURN
900 REM ************************
901 REM SET RECORD'S TRACK, SECTOR & RECORD POINTER FROM INDEX ARRAYS
902 REM
905 D=RD
910 E=0
915 IFAS=-1THENRP=CR*RS+1:GOTO950
920 RP=INT((CR-1)/RB+EP):IFRP>NB OR RP<0THENEN=41:GOTO1900
930 T=IT%(RP):S=IS%(RP)
940 RP=INT(((CR-1)/RB-RP+EP)*RS*RB)+1
950 IFRP>254THEN EN=41:GOTO1900
960 RETURN
1000 REM ************************
1001 REM INPUT 2040 ERROR STATUS
1002 REM
1005 INPUT#CC,EN$,EM$,ET,ES
1010 EN=VAL(EN$):E=0
1015 IF EN$="00" THEN RETURN
1017 ET$=STR$(ET):ES$=STR$(ES)
1020 IFEN$<>RIGHT$("0"+MID$(STR$(EN),2),2)THEN1070
1030 IF EN=1 THEN EM$= ET$+" "+EM$: RETURN
1035 E=E+1
1040 EM$="■"+EN$+"■ "+EM$
1050 IF EN<30 OR EN=65 THEN EM$=EM$+" ON "+ET$+", "+ES$
1060 RETURN
1070 EM$="■SYSTEM NOT RESPONDING PROPERLY"
1080 EM$=EM$+EN$+EM$+ET$+ES$
1085 E=E+1
1090 RETURN
1100 REM ************************
1101 REM CREATE DESCRIPTOR FILE
1102 REM INPUT: F$= FILENAME
1103 REM        ID$,NR,NF,FS%(),FT%(),FH$()
1104 REM        DD= DESCRIPTOR FILE DRIVE #
1105 REM        RD= RANDOM DISK DRIVE #
1106 REM DRIVES MUST BE INITIALIZED
1109 REM
1110 RS=1:D=RD
1115 FORA0=1TONF:FP%(A0)=RS:RS=FS%(A0)+RS+1:NEXT:RS=RS-1
1116 RB=INT(254/RS+EP)
1120 OPENC0,DN,C0,"#":GOSUB1000:IFETHEN1900
1121 GOSUB1280
1122 PRINT#CC,"B-P:"C0;1
1123 FORA0=1TORB:FORA1=1TONF
1124 PRINT#C0,LEFT$(SP$,FS%(A1));M$;
1126 NEXTA1,A0
1130 NB=INT(NR/RB+EP):IF(NR/RB-NB)*RB>=1THENNB=NB+1
1135 T=1:S=0:GOSUB150
1140 FORA0=0TONB-1:GOSUB710:IFETHEN1900
1145 IT%(A0)=T:IS%(A0)=S:GOSUB430:NEXT
1150 GOSUB710
1152 PRINT#CC,"B-P:"C0;1
1155 PRINT#C0,NR;M$;1;M$;NB;M$;RS;M$;RB;M$;NF;M$;
1160 PRINT#CC,"B-W:"C0;D;T;S
1165 A$=STR$(DD)+":"+LEFT$(F$+SP$,10)+",DESCR,U,W"
1166 OPENC1,DN,C1,A$
1167 GOSUB1000:IFETHEN1900
```

```
1168 PRINT#C1,ID$;M$;T;M$;S;M$;
1170 FORA0=1TONF:PRINT#C1,CHR$(FS%(A0));CHR$(FT%(A0));FH$(A0);M$;:NEXT
1175 FORA0=0TONB-1:PRINT#C1,CHR$(IT%(A0));CHR$(IS%(A0));:NEXT
1180 CLOSEC1:CLOSEC0:RETURN
1200 REM ************************
1201 REM OPEN RELATIVE FILE
1202 REM INPUT: F$= FILENAME
1203 REM        DD= DESCRIPTOR FILE DRIVE #
1204 REM        RD= RANDOM DISK DRIVE #
1205 REM DRIVES MUST BE INITIALIZED
1209 REM
1210 A$=STR$(DD)+":"+LEFT$(F$+SP$,10)+".DESCR,U,R"
1215 OPENC1,DN,C1,A$:GOSUB1000:IFETHEN1900
1220 INPUT#C1,ID$,T,S
1225 OPENC0,DN,C0,"#":GOSUB1000:IFETHEN1900
1226 GOSUB1280
1227 PRINT#CC,"B-R:";C0;RD;T;S:GOSUB1000:IFETHEN1900
1230 INPUT#C0,NR,FR,NB,RS,RB,NF
1235 GOSUB100:FT%(0)=T:FS%(0)=S
1240 FORA0=1TONF:GOSUB1298:FS%(A0)=ASC(A$)
1245 GOSUB1298:FT%(A0)=ASC(A$)
1250 INPUT#C1,FH$(A0):NEXT
1255 FORA0=0TONB-1:GOSUB1298:IT%(A0)=ASC(A$)
1260 GOSUB1298:IS%(A0)=ASC(A$):NEXT
1265 GOSUB1000:IFETHEN1900
1270 CLOSEC1
1275 RETURN
1280 PRINT#CC,"U1:";C0;RD;",18,0":GOSUB1000:IFETHEN1900
1285 PRINT#CC,"B-P:";C0;162
1286 GET#C0,A$,A1$:A$=A$+A1$:IFID$<>A$THENEN=43:EM$="WRONG RAND DISK":GOTO1900
1290 RETURN
1298 GET#C1,A$:IFA$=""THENA$=CHR$(0)
1299 RETURN
1400 REM ************************
1401 REM CLOSE RELATIVE FILE
1402 REM INPUT: VARIABLES FROM OPEN SHOULD BE VALID
1409 REM
1410 PRINT#CC,"B-P:"C0;1
1420 PRINT#C0,NR;M$;FR;M$;NB;M$;RS;M$;RB;M$;NF;M$;
1430 PRINT#CC,"B-W:"C0;D;FT%(0);FS%(0)
1440 CLOSEC0
1490 RETURN
1900 E=E+1:RETURN
2000 INPUT"▯▯▯DO YOU WISH TO CREATE A FILE  N▮▮▮▮";A$:IFLEFT$(A$,1)<>"Y"THEN2100
2001 INPUT"▯RANDOM FILE NAME";F$
2002 INPUT"KEY FILE DRIVE NUMBER";DD
2003 INPUT"RANDOM FILE DRIVE NUMBER";RD
2005 INPUT"ENTER ID OF RANDOM DISK   ▁▁▮▮▮▮▮";ID$:ID$=LEFT$(ID$,2)
2006 INPUT"NUMBER OF RECORDS";NR
2007 INPUT"NUMBER OF FIELDS PER RECORD";NF
2010 GOSUB110
2015 PRINT"▯ INPUT FIELD NAME,FIELD SIZE,FIELD TYPE"
2016 PRINT"    TYPES: 0=BINARY, 1=NUMERIC, 2=ALPHA▯"
2019 RS=0
2020 FORI=1TONF:PRINT"FIELD";I,:INPUTFH$(I),FS%(I),FT%(I):RS=FS%(I)+RS+1:NEXT
2025 A$="I":IFDD=RDTHENA$="I"+STR$(DD)
2030 PRINT#CC,A$
2040 GOSUB1100:IFETHEN3900
```

```
2050 OPEN4,8,4,STR$(DD)+":"+LEFT$(F$+SP$,10)+".KEY01,U,W"
2055 PRINT#4,0;M$;:CLOSE4
2090 GOTO2120
2100 REM OPEN RANDOM FILE FOR ACCESS
2103 INPUT"■RANDOM FILE NAME";F$
2105 INPUT"KEY FILE DRIVE NUMBER";DD
2110 INPUT"RANDOM FILE DRIVE NUMBER";RD
2120 GOSUB1200:IFETHEN3900
2140 OPEN4,8,4,STR$(DD)+":"+LEFT$(F$+SP$,10)+".KEY01,U"
2142 INPUT#4,RR:IFRR=0THEN2147
2145 FORI=1TORR:INPUT#4,K1$(I),RR%(I):NEXT
2147 CLOSE4
2150 PRINT"■■■■■■■■■■■■■■■■■■■■SAMPLE RANDOM ACCESS■"
2155 PRINT"TYPE // TO QUIT■"
2156 PRINT"(HIT RETURN TO ADD RECORD)"
2160 PRINT"■■WHOSE RECORD DO YOU"
2161 INPUT"WISH TO SEE   ■■■";RR$
2165 IFRR$=" "THEN2310
2167 IFRR$="//"THEN2400
2168 IFRR$="/DIR"THENGOSUB4000:GOTO2160
2170 FORII=1TORR:IFK1$(II)<>RR$THENNEXT:GOTO2300
2175 CR=RR%(II):GOSUB300
2180 FORI=1TONF:PRINTI;")"FH$(I)":",F$(I):NEXT:PRINT
2185 FF=0
2190 INPUT"ANY MODS   ■■■";A$:IFLEFT$(A$,1)<>"Y"THEN2220
2195 INPUT"WHICH FIELD";A
2200 PRINT"  "F$(A):PRINT"■";:INPUTF$(A):F(A)=VAL(F$(A))
2210 FF=1:GOTO2190
2220 IFFF=0THEN2160
2222 IFA=1THENK1$(II)=F$(A)
2225 GOSUB200
2230 GOTO2160
2300 PRINT"■RECORD NOT PRESENT"
2305 INPUT"DO YOU WISH TO ADD";A$:IFLEFT$(A$,1)<>"Y"THEN2160
2310 PRINT"■■**** ADD RECORD ****■■"
2312 IFFR>NRTHEN2500
2315 CR=FR:FR=FR+1:RR=RR+1
2320 FORI=1TONF:PRINTFH$(I);:INPUTF$(I):F(I)=VAL(F$(I)):NEXT
2330 GOSUB200
2340 K1$(RR)=F$(1):RR%(RR)=CR
2350 GOTO2160
2400 REM CLOSE RAND FILE
2405 GOSUB1400
2410 OPEN4,8,4,"@"+STR$(DD)+":"+LEFT$(F$+SP$,10)+".KEY01,U,W"
2420 GOSUB1000:IFETHEN3900
2430 PRINT#4,RR;M$;
2440 FORI=1TORR:PRINT#4,K1$(I);M$;RR%(I);M$;:NEXT
2445 GOSUB1000:IFETHEN3900
2450 CLOSE4
2455 GOSUB1000:IFETHEN3900
2490 POKE1022,8:END:REM TURN DOS SUPPORT 3.1 ON
2500 PRINT"THE FILE IS FULL, NO ADDITIONAL RECORDS MAY BE ADDED"
2510 GOTO2160
3900 PRINTE,EM$:STOP
4000 FORDI=0TONR:PRINTK1$(DI):NEXT:RETURN
```

For those of you who need to output a listing on an ASCII-only printer, this program will convert and list the graphic characters between [and], and the cursor control between < and >.

The first listing is the converted program (the actual convert program starts at line 5000, the rest is a demo.). The second listing is the normal program (as printed on a PET printer).

```
10 REM III CHARS 2.1
20 GOTO8000
50 A$="     "+CHR$(34)
60 PRINT
70 PRINT
80 PRINT"■    CRSR CHARACTERS    "
90 PRINT
100 PRINTA$"■  HOME CRSR
110 PRINTA$"□  CLEAR CRSR
120 PRINTA$"■  CRSR DOWN
130 PRINTA$"□  CRSR UP
140 PRINTA$"■  CRSR RIGHT
150 PRINTA$"■  CRSR LEFT
160 PRINTA$"■  RVS ON
170 PRINTA$"■  RVS OFF
180 PRINTA$"■  DELETE
190 PRINTA$"■  INSERT
200 PRINT
1000 PRINT"■   GRAPHIC CHARACTERS   "
1010 PRINT
1020 PRINTA$"   SHIFT SPACE
1030 PRINTA$"▌  LEFT 4 VERT
1040 PRINTA$"▄  BOTTOM 4 HOR
1050 PRINTA$"▔  HORZ LINE 1
1060 PRINTA$"_  HORZ LINE 8
1070 PRINTA$"▏  VERT LINE 1
1080 PRINTA$"▒  FULL SHADE
1090 PRINTA$" ▕  VERT LINE 8
1100 PRINTA$"▁  BOTTOM SHADE
1110 PRINTA$"◤  LEFT TRIANGLE
1120 PRINTA$" ▐  RIGHT 2 VERT
1130 PRINTA$"├  VERT RIGHT JUNCTION
1140 PRINTA$"▗  BOTTOM RIGHT SQUARE
1150 PRINTA$"└  TOP RIGHT CORNER
1160 PRINTA$"┐  BOTTOM LEFT CORNER
1170 PRINTA$"▂  BOTTOM 2 HORZ
1180 PRINTA$"┌  BOTTOM RIGHT CORNER
1190 PRINTA$"┴  HORZ TOP JUNCTION
1200 PRINTA$"┬  HORZ BOTTOM JUNCTION
1210 PRINTA$"┤  VERT LEFT JUNCTION
1220 PRINTA$"▎  LEFT 2 VERT
1230 PRINTA$"▍  LEFT 3 VERT
1240 PRINTA$" ▊  RIGHT 3 VERT
```

```
1250 PRINTA$"‾  TOP 2 HORZ
1260 PRINTA$"▔  TOP 3 HORZ
1270 PRINTA$"▂  BOTTOM 3 HORZ
1280 PRINTA$"┘  BIG BOTTOM RT CORNER
1290 PRINTA$"▖  BOTTOM LEFT SQUARE
1300 PRINTA$" ▪  TOP RIGHT SQUARE
1310 PRINTA$"┐  TOP LEFT CORNER
1320 PRINTA$"▪  TOP LEFT SQUARE
1330 PRINTA$"▚  DOUBLE SQUARES
1340 PRINTA$"─  HORZ LINE 5
1350 PRINTA$"♠  SPADE
1360 PRINTA$"|  VERT LINE 4
1370 PRINTA$"─  HORZ LINE 4
1380 PRINTA$"─  HORZ LINE 3
1390 PRINTA$"‾  HORZ LINE 2
1400 PRINTA$"_  HORZ LINE 6
1410 PRINTA$"|  VERT LINE 3
1420 PRINTA$" |  VERT LINE 6
1430 PRINTA$"╲  BOTTOM LEFT CURVE
1440 PRINTA$" ╲  TOP RIGHT CURVE
1450 PRINTA$"╱  TOP LEFT CURVE
1460 PRINTA$"L  BIG BOTTOM LF CORNER
1470 PRINTA$"╲  LEFT DIAG.
1480 PRINTA$"╱  RIGHT DIAG.
1490 PRINTA$"┌  BIG TOP LF CORNER
1500 PRINTA$"┐  BIG TOP RT CORNER
1510 PRINTA$"●  WHITE CIRCLE
1520 PRINTA$"_  HORZ LINE 7
1530 PRINTA$"♥  HEART
1540 PRINTA$"|  VERT LINE 2
1550 PRINTA$" ╱  BOTTOM RIGHT CURVE
1560 PRINTA$"╳  CROSS DIAG.
1570 PRINTA$"o  BLACK CIRCLE
1580 PRINTA$"♣  CLUB
1590 PRINTA$" |  VERT LINE 7
1600 PRINTA$"♦  DIAMOND
1610 PRINTA$"+  PLUS JUNCTION
1620 PRINTA$"▒  LEFT SHADE
1630 PRINTA$"|  VERT LINE 5
1640 PRINTA$"π  PI
1650 PRINTA$"◥  RIGHT TRIANGLE
3000 END
5000 DATA32,112,0,144,7,240,5,201
5010 DATA171,240,1,96,32,115,200,32
5020 DATA44,197,32,118,0,240,12,201
5030 DATA171,208,240,32,112,0,32,115
5040 DATA200,208,232,104,104,165,17,5
5050 DATA18,208,6,169,255,133,17,133
5060 DATA18,160,1,132,9,177,92,240
5070 DATA67,32,225,255,32,226,201,200
5080 DATA177,92,170,200,177,92,197,18
5090 DATA208,4,228,17,240,2,176,44
5100 DATA132,70,32,217,220,169,32,164
5110 DATA70,41,127,32,69,202,201,34
5120 DATA208,6,165,9,73,255,133,9
5130 DATA200,240,17,177,92,208,16,168
```

```
 5140 DATA177,92,170,200,177,92,134,92
 5150 DATA133,93,208,181,76,137,195,16
 5160 DATA58,201,255,240,80,36,9,48
 5170 DATA30,56,233,127,170,132,70,160
 5180 DATA255,202,240,8,200,185,146,192
 5190 DATA16,250,48,245,200,185,146,192
 5200 DATA48,181,32,69,202,208,245,201
 5210 DATA160,144,38,72,169,91,32,69
 5220 DATA202,104,41,127,32,69,202,169
 5230 DATA93,208,160,201,32,176,156,72
 5240 DATA169,60,32,69,202,104,41,127
 5250 DATA9,64,32,69,202,169,62,208
 5260 DATA138,9,96,208,234,169,94,208
 5270 DATA210
 8000 T=PEEK(53):T=T-1
 8020 POKE53,T:T=T*256
 9000 FORI=T TO T+216
 9010 READX
 9020 POKEI,X
 9030 NEXT
 9200 PRINT"■■   ASCII LIST SYNTAX   ■"
 9220 PRINT"SYS"T":80-3000"
 9240 PRINT" ↑  ↑      ↑ ┌──────────────────────┐
 9250 PRINT" |  |      └─| RANGE OF LINES TO LIST |
 9260 PRINT" |  |        └──────────────────────┘
 9270 PRINT" |  | ┌──────────────────────────┐
 9280 PRINT" |  └─| START ADDRESS OF ASCII LIST |
 9290 PRINT" |    └──────────────────────────┘
 9300 PRINT" | ┌──────────────────────────┐
 9310 PRINT" └─| CALL MACHINE LANGUAGE PRGM |
 9320 PRINT"   └──────────────────────────┘
 9500 CLR
READY.

 10 REM III CHARS 2.1
 20 GOTO8000
 50 A$="    "+CHR$(34)
 60 PRINT
 70 PRINT
 80 PRINT"<R>    CRSR CHARACTERS    "
 90 PRINT
 100 PRINTA$"<S>  HOME CRSR
 110 PRINTA$"<3>  CLEAR CRSR
 120 PRINTA$"<Q>  CRSR DOWN
 130 PRINTA$"<1>  CRSR UP
 140 PRINTA$"<]>  CRSR RIGHT
 150 PRINTA$"<=>  CRSR LEFT
 160 PRINTA$"<R>  RVS ON
 170 PRINTA$"<2>  RVS OFF
 180 PRINTA$"<T>  DELETE
 190 PRINTA$"<4>  INSERT
 200 PRINT
 1000 PRINT"<R>   GRAPHIC CHARACTERS   "
 1010 PRINT
 1020 PRINTA$"[ ]  SHIFT SPACE
 1030 PRINTA$"[!]  LEFT 4 VERT
 1040 PRINTA$"["]  BOTTOM 4 HOR
 1050 PRINTA$"[#]  HORZ LINE 1
 1060 PRINTA$"[$]  HORZ LINE 8
```

```
1070 PRINTA$"[%]  VERT LINE 1
1080 PRINTA$"[&]  FULL SHADE
1090 PRINTA$"[']  VERT LINE 8
1100 PRINTA$"[(]  BOTTOM SHADE
1110 PRINTA$"[)]  LEFT TRIANGLE
1120 PRINTA$"[*]  RIGHT 2 VERT
1130 PRINTA$"[+]  VERT RIGHT JUNCTION
1140 PRINTA$"[,]  BOTTOM RIGHT SQUARE
1150 PRINTA$"[-]  TOP RIGHT CORNER
1160 PRINTA$"[.]  BOTTOM LEFT CORNER
1170 PRINTA$"[/]  BOTTOM 2 HORZ
1180 PRINTA$"[0]  BOTTOM RIGHT CORNER
1190 PRINTA$"[1]  HORZ TOP JUNCTION
1200 PRINTA$"[2]  HORZ BOTTOM JUNCTION
1210 PRINTA$"[3]  VERT LEFT JUNCTION
1220 PRINTA$"[4]  LEFT 2 VERT
1230 PRINTA$"[5]  LEFT 3 VERT
1240 PRINTA$"[6]  RIGHT 3 VERT
1250 PRINTA$"[7]  TOP 2 HORZ
1260 PRINTA$"[8]  TOP 3 HORZ
1270 PRINTA$"[9]  BOTTOM 3 HORZ
1280 PRINTA$"[:]  BIG BOTTOM RT CORNER
1290 PRINTA$"[;]  BOTTOM LEFT SQUARE
1300 PRINTA$"[<]  TOP RIGHT SQUARE
1310 PRINTA$"[=]  TOP LEFT CORNER
1320 PRINTA$"[>]  TOP LEFT SQUARE
1330 PRINTA$"[?]  DOUBLE SQUARES
1340 PRINTA$"[@]  HORZ LINE 5
1350 PRINTA$"[A]  SPADE
1360 PRINTA$"[B]  VERT LINE 4
1370 PRINTA$"[C]  HORZ LINE 4
1380 PRINTA$"[D]  HORZ LINE 3
1390 PRINTA$"[E]  HORZ LINE 2
1400 PRINTA$"[F]  HORZ LINE 6
1410 PRINTA$"[G]  VERT LINE 3
1420 PRINTA$"[H]  VERT LINE 6
1430 PRINTA$"[I]  BOTTOM LEFT CURVE
1440 PRINTA$"[J]  TOP RIGHT CURVE
1450 PRINTA$"[K]  TOP LEFT CURVE
1460 PRINTA$"[L]  BIG BOTTOM LF CORNER
1470 PRINTA$"[M]  LEFT DIAG.
1480 PRINTA$"[N]  RIGHT DIAG.
1490 PRINTA$"[O]  BIG TOP LF CORNER
1500 PRINTA$"[P]  BIG TOP RT CORNER
1510 PRINTA$"[Q]  WHITE CIRCLE
1520 PRINTA$"[R]  HORZ LINE 7
1530 PRINTA$"[S]  HEART
1540 PRINTA$"[T]  VERT LINE 2
1550 PRINTA$"[U]  BOTTOM RIGHT CURVE
1560 PRINTA$"[V]  CROSS DIAG.
1570 PRINTA$"[W]  BLACK CIRCLE
1580 PRINTA$"[X]  CLUB
1590 PRINTA$"[Y]  VERT LINE 7
1600 PRINTA$"[Z]  DIAMOND
1610 PRINTA$"[[]  PLUS JUNCTION
```

```
 1620 PRINTA$"[\]  LEFT SHADE
 1630 PRINTA$"[]]  VERT LINE 5
 1640 PRINTA$"[↑]  PI
 1650 PRINTA$"[←]  RIGHT TRIANGLE
 3000 END
 5000 DATA32,112,0,144,7,240,5,201
 5010 DATA171,240,1,96,32,115,200,32
 5020 DATA44,197,32,118,0,240,12,201
 5030 DATA171,208,240,32,112,0,32,115
 5040 DATA200,208,232,104,104,165,17,5
 5050 DATA18,208,6,169,255,133,17,133
 5060 DATA18,160,1,132,9,177,92,240
 5070 DATA67,32,225,255,32,226,201,200
 5080 DATA177,92,170,200,177,92,197,18
 5090 DATA208,4,228,17,240,2,176,44
 5100 DATA132,70,32,217,220,169,32,164
 5110 DATA70,41,127,32,69,202,201,34
 5120 DATA208,6,165,9,73,255,133,9
 5130 DATA200,240,17,177,92,208,16,168
 5140 DATA177,92,170,200,177,92,134,92
 5150 DATA133,93,208,181,76,137,195,16
 5160 DATA58,201,255,240,80,36,9,48
 5170 DATA30,56,233,127,170,132,70,160
 5180 DATA255,202,240,8,200,185,146,192
 5190 DATA16,250,48,245,200,185,146,192
 5200 DATA48,181,32,69,202,208,245,201
 5210 DATA160,144,38,72,169,91,32,69
 5220 DATA202,104,41,127,32,69,202,169
 5230 DATA93,208,160,201,32,176,156,72
 5240 DATA169,60,32,69,202,104,41,127
 5250 DATA9,64,32,69,202,169,62,208
 5260 DATA138,9,96,208,234,169,94,208
 5270 DATA210
 8000 T=PEEK(53):T=T-1
 8020 POKE53,T:T=T*256
 9000 FORI=T TO T+216
 9010 READX
 9020 POKEI,X
 9030 NEXT
 9200 PRINT"<3><R>   ASCII LIST SYNTAX   <Q>"
 9220 PRINT"SYS"T":80-3000"
 9240 PRINT" ↑  ↑       ↑ [0][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@]
@][@][@][@][@][@][@][@][.]
 9250 PRINT" []]  []]       [-][@][3] RANGE OF LINES TO LIST []]
 9260 PRINT" []]  []]         [-][@][@][@][@][@][@][@][@][@][@][@][@][@][@
[@][@][@][@][@][@][@][@][=]
 9270 PRINT" []]  []] [0][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@]
@][@][@][@][@][@][@][@][@][@][@][.]
 9280 PRINT" []]  [-][@][3] START ADDRESS OF ASCII LIST []]
 9290 PRINT" []]    [-][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@
[@][@][@][@][@][@][@][@][@][@][@][=]
 9300 PRINT" []] [0][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@
[@][@][@][@][@][@][@][@][@][@][.]
 9310 PRINT" [-][@][3] CALL MACHINE LANGUAGE PRGM []]
 9320 PRINT"   [-][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@][@
][@][@][@][@][@][@][@][=]
 9500*CLR
READY.
```

Peripherals & Attachments

USING THE PET AS A FREQUENCY COUNTER by R. Lynn and G. Stark

In the course of designing digital logic circuits it is often necessary to measure the number of pulses that occur in a given time. If the number of pulses to be counted is large or if the time interval is long the measurement can be quite difficult or tedious.

One way to such a measurement is to use a $3000 plus delayed sweep, dual trace oscilloscope. A signal representing the time interval is displayed on one channel of the scope and is used to trigger the main time base. The pulses are displayed on the other channel. The main sweep is set slow enough to display the entire time interval and the delayed sweep is set fast enough to discriminate the pulses. The delayed sweep vernier is used to walk the pulses across the screen. The pulses are then counted, usually this has to be done several times to be sure one has not missed a count. One never has a warm feeling about the results, since the measurement is done at only one period in time and one is never quite sure that when the count was wrong the cause might have really been the circuit under test.

Another way to make this measurement is with a $2000 plus gated frequency counter. This technique has several advantages over the scope method. Human error is not a factor since the counter does the counting, but the notorious plus minus one count error of digital counters can be unsettling. Secondly the counter can be left counting for a long period of time and if it is dilligently monitored a more reliable measurement can be made.

With the appearance of the PET computer in the instrumentation world a third way is available for making this type of measurement. A user port connector, this program and the $795 8K PET computer can do a better job at making this measurement. The PET can compile historical data over hours, weeks, days, or even longer. This capability can be very useful if one suspects that the circuit may malfunction only occasionally.

The signals should be TTL levels but since the PET User Port to which they will be connected is a direct connection to an MOS input higher or lower voltages may work (PROCEED AT YOUR OWN RISK). In any case very light loading is presented to the circuit under test.

The gate signal is applied to the PET User Port pin L (PA-7) and the pulses are fed to pin K (PA-6).

The following listings are an assembly program to do the counting and a BASIC program to POKE the machine code into the second cassette buffer as well as display the results of the counts on the PET video screen.

FREQUENCY COUNTER

```
 4 BB=11
 5 DIMSM(BB),CT(BB)
 7 FORI=0TOBB:SM(I)=-1:CT(I)=0:NEXT
 8 GOSUB200:REM LOADER PROG
 9 GOSUB9000:REM SET UP THE SCREEN
10 SYS(832):REM DO THE COUNTING AT $340
20 PRINT"■■■■■■■■■■■■■■■■THE COUNT IS■ ";:A=256*PEEK(1)+PEEK(0):PRINTA
40 FOR I=0TOBB
50 IFSM(I)=ATHENCT(I)=CT(I)+1:GOTO100
60 IFSM(I)=-1THENSM(I)=A:CT(I)=1:GOTO100
70 NEXT
100 PRINT"■■■■■■■■■■BINS ","COUNTS■"
105 FORI=0TOBB:PRINT"■■■■■■■■■";SM(I),CT(I):NEXT
120 GOTO10
200 REM LOADER ROUTINE
210 READC$
220 GOSUB300:REM CONVERT HEX TO DECIMAL
230 ADDRESS=NUM
240 READC$
250 IFC$="END"THEN RETURN
260 GOSUB300:REM CONVERT HEX TO DECIMAL
270 POKEADDRESS,NUM:ADDRESS=ADDRESS+1
280 GOTO240
300 REM CONVERT HEX TO DECIMAL
310 NUM=0:LS=LEN(C$)
320 FORIS=0TOLS-1
330 AS=ASC(MID$(C$,LS-IS,1))-48
340 NUM=NUM+(16↑IS)*(AS+7*(AS>9))
350 NEXTIS
360 RETURN
1000 DATA0340,78,A9,00,85,00,85,01,2C
1001 DATA41,E8,30,FB,2C,41,E8,10
1002 DATAFB,2C,41,E8,70,03,30,F9
1003 DATA60,E6,00,D0,02,E6,01,2C
1004 DATA41,E8,50,ED,10,F2,70,F7
1006 DATAEND
9000 FORI=32768TO33767:POKEI, 86:NEXT
9010 RETURN
9999 END
```

```
                0090            .LS
                0100            .BA $340
                0105 USRPORT    .DE $E841
                0106 USR        .DE 0
                0107 ;SUB TO COUNT PULSES ON
                0108 ;THE PET USER PORT PA6 GATED BY PA7
0340- 78        0110 BEGIN      SEI  ;MASK INTERUPTS
0341- A9 00     0120            LDA #0
0343- 85 00     0130            STA *USR
0345- 85 01     0140            STA *USR+1    ;ZERO COUNTER
0347- 2C 41 E8  0150 TIME       BIT USRPORT   ;TEST GATE
034A- 30 FB     0160            BMI TIME      ;WAIT FOR AN EDGE
034C- 2C 41 E8  0170 T2         BIT USRPORT   ;TEST FOR A PULSE
034F- 10 FB     0180            BPL T2        ;WAIT FOR AN EDGE
0351- 2C 41 E8  0190 START      BIT USRPORT   ;NEXT EDGE?
```

```
0354- 70 04      0200              BVS CNT1      ;COUNT ONE
0356- 30 F9      0210              BMI START     ;IF TIME NOT UP CONTINUE
0358- 58         0220              CLI  ;UNMASK INTERUPTS
0359- 60         0230              RTS  ;DONE
035A- E6 00      0240 CNT1         INC *USR      ;COUNT
035C- D0 02      0250              BNE CNTR2     ;OVERFLOW?
035E- E6 01      0260              INC *USR+1    ;YES, COUNT 256
0360- 2C 41 E8   0270 CNTR2        BIT USRPORT   ;TEST FOR TRAILING EDGE OF GATE
0363- 50 EC      0280              BVC START     ;EXIT
0365- 10 F3      0290              BPL CNT1      ;COUNT AGAIN
0367- 70 F7      0300              BVS CNTR2     ;WAIT TILL PULSE IS DONE
                 0310              .EN

LABEL FILE:  [ / = EXTERNAL ]

/USRPORT=E841          /USR=0000          BEGIN=0340
TIME=0347              T2=034C            START=0351
CNT1=035A              CNTR2=0360
//0000,0369,0369
```

THE 2022/2023 PRINTERS, AN OVERVIEW

The 2022 and 2023 are Tractor and Friction Feed printer mechanisms, respectively. Both feature full upper and lower case, graphic, and reverse-field capabilities. Through the use of secondary addresses in the OPEN statement, the user can invoke many different features. These are listed by address:

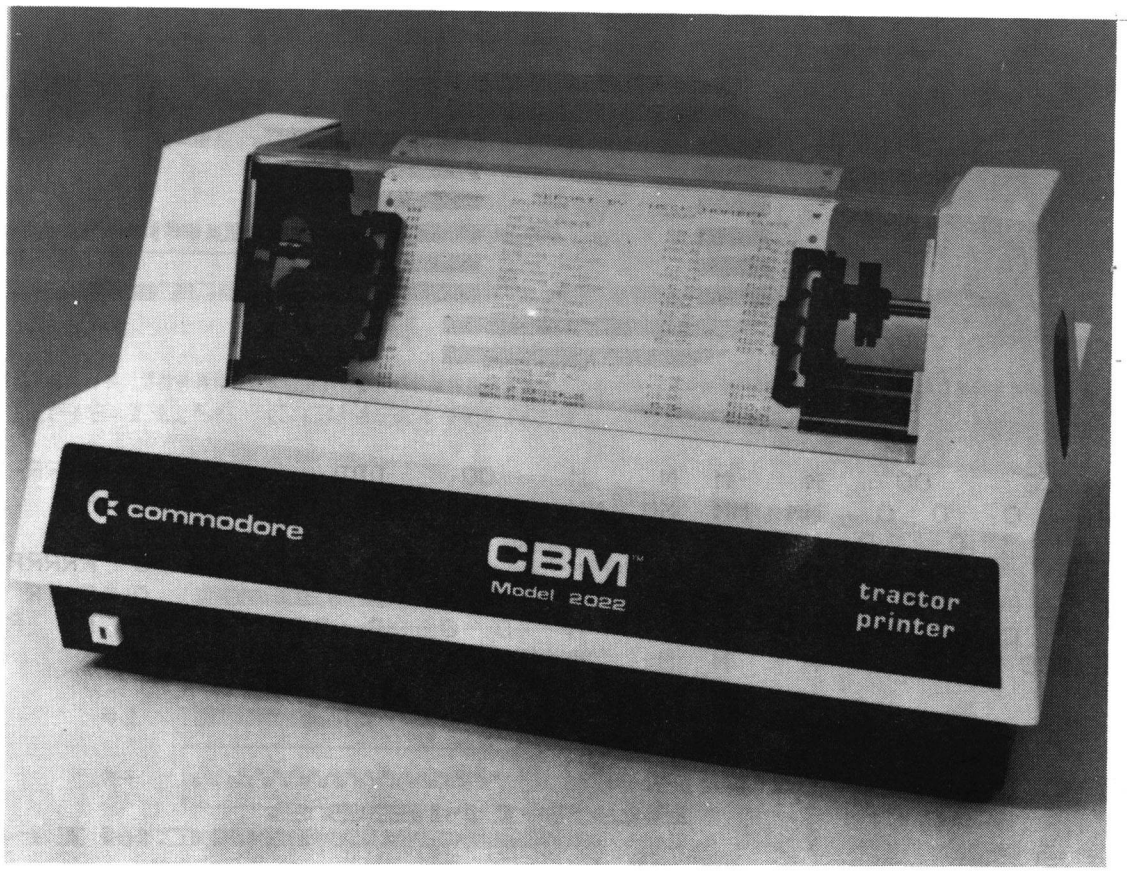
0. Print data exactly as received.
1. Print data under format control.
2. Define format.
3. Set the number of lines per page.
4. Enable the printer diagnostic messages.
5. Define the programmable character - this character is printed using a CHR$(254).
6. Set number of lines per inch - this is only available on the 2022 tractor-feed printer.

The printers use any standard roll of fan-fold paper, and print up to 80 characters per line. The 2022 tractor feed printer handles any standard pin-feed forms, such as invoices, labels, or multi-part forms. The 2022 also features an adjustable paper width, so any size form, up to the 80 character limit, can be used with ease.

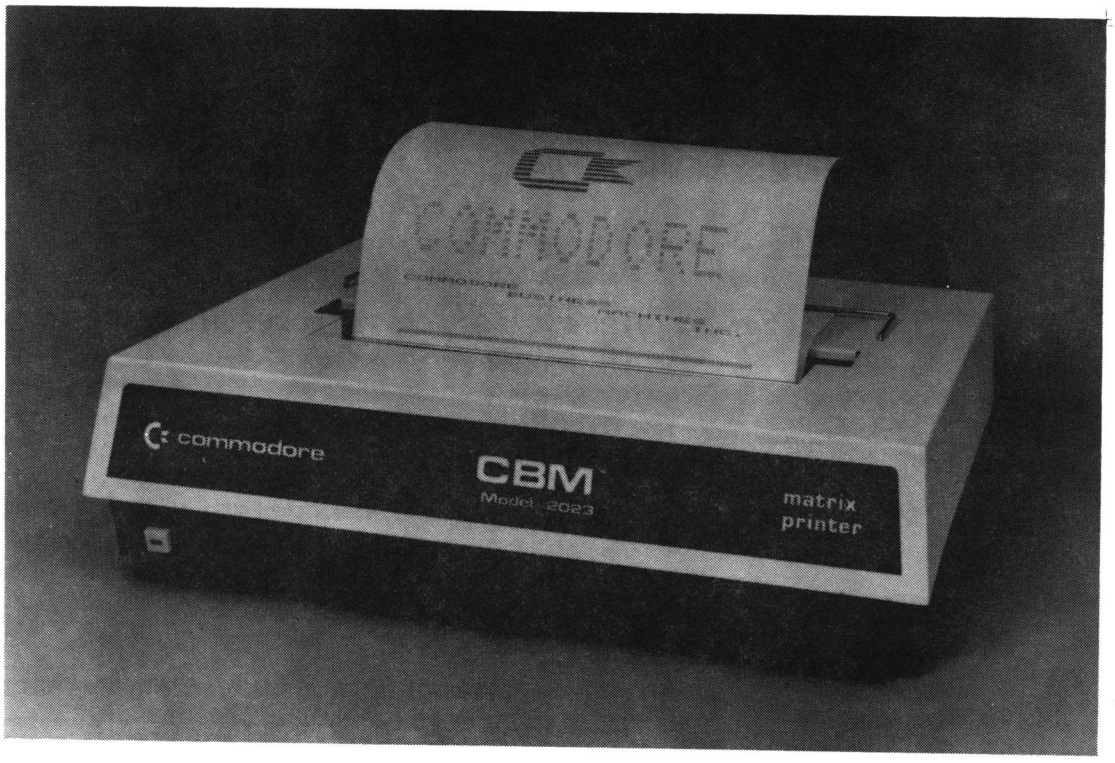
The printers are economically priced at only $995.00 for the 2022, and $849.00 for the 2023.

These printers are IDEAL for use with the COMMODORE WORD PROCESSING SYSTEM, as well as all your other printing needs.

To follow is a copy of the output from a Printer Demo Program, which only begins to illustrate the benefits of these new printers.



2022 TRACTOR PRINTER



2023 MATRIX PRINTER

```
  CCC      OO     M     M  M     M     OO     DDD        OO     RRRRR    EEEEEE
 C   C    O  O    MM   MM  MM   MM    O  O    D  D      O  O    R    R   E
C        O    O   M MM M   M MM M   O    O    D   D    O    O   R    R   E
C        O    O   M MM M   M MM M   O    O    D   D    O    O   RRRRR    EEEE
C        O    O   M     M  M     M  O    O    D   D    O    O   R  R     E
 C   C    O  O    M     M  M     M    O  O    D  D      O  O    R   R    E
  CCC      OO     M     M  M     M     OO     DDD        OO     R    R   EEEEEE
```

COMMODORE
BUSINESS
MACHINES
INC.

PRESENTS

THE

MODEL
2022 & 2023
PRINTERS

OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO

EACH PRINTER HAS FULL GRAPHIC CAPACITY

UPPER CASE

!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ

lower case

!"#$%&'()*+,-./0123456789:;<=>?@abcdefghijklmnopqrstuvwxyz

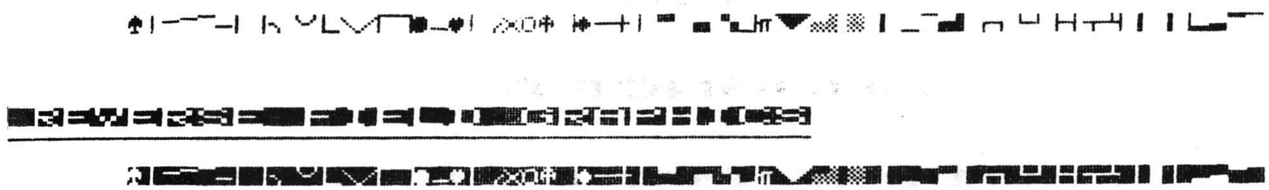
REVERSE FIELD UPPER CASE

!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ

reverse field lower case

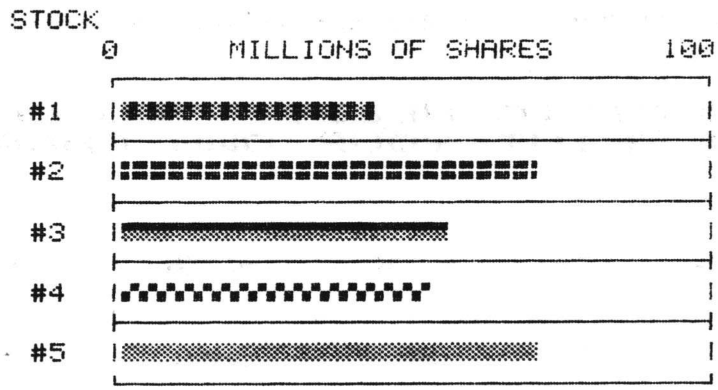
!"#$%&'()*+,-./0123456789:;<=>?@abcdefghijklmnopqrstuvwxyz

GRAPHICS



##

YOU CAN EASILY CONSTRUCT GRAPHS



HORIZONTAL BAR GRAPHS

OR FORMAT DATA IN CONCISE CHARTS

MORTGAGE AMORTIZATION TABLE

PRINCIPAL $ 2100 AT 6 % FOR 1.0 YEARS
REGULAR PAYMENT = $ 75

No	Interest	Amoritzation	Balance	Accum Interest
1	$ 10.50	$ 64.50	$2035.50	$ 10.50
2	10.18	64.82	1970.68	20.68
3	9.85	65.15	1905.53	30.53
4	9.53	65.47	1840.06	40.06
5	9.20	65.80	1774.26	49.26
6	8.87	66.13	1708.13	59.13
7	8.54	66.46	1641.67	66.67
8	8.21	66.79	1574.88	74.88
9	7.87	67.13	1507.75	82.75
10	7.54	67.46	1440.29	90.29
11	7.20	67.80	1372.49	97.49
12	6.86	68.14	1304.35	104.35

YOU

CAN

PRINT

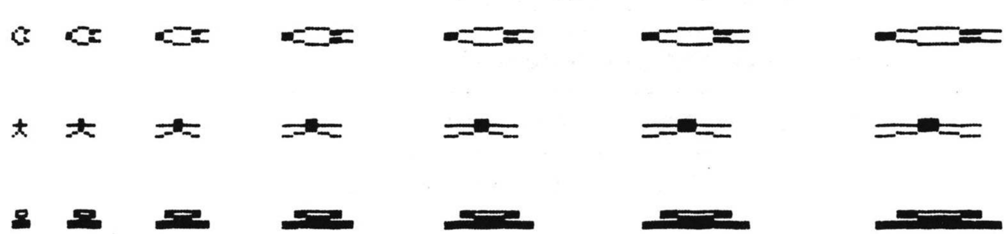
ENHANCED

CHARACTERS

Ideal for your report headings.

&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

AND FOR SPECIFIC NEEDS, YOU CAN EVEN DESIGN AND PRINT YOUR OWN CHARACTERS



VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV

OTHER BENEFITS INCLUDE:

MODEL 2023:

1) Fast-Only Moments Before Your Data Is Permanatley Recorded
2) Easy to Use-Eliminates Complex Interfacing
3) An Intelligent Peripherial- Does Not Use The computers Memory!
4) Self-Diagnostic Test Prefomed At Power-up.
5) Uses Standard Paper And Ribbon
6) Serviced By Over 400 Dealers
7) An Outstanding Value At Only $849.00

MODEL 2022:

1) Has All Of The Above Features Plus-
2) Offers Tractor Feed Mechanism For Precise Document Feed Control
3) Handles Any Standard Pin Feed Form-For All Your Business Needs
4) Features Programmable Line Spacing For Custom needs
5) Economically Priced At Only $995.00

++

SEE THE OTHER FINE COMMODORE PRODUCTS INCLUDING:

THE CBM BUSINESS KEYBOARD COMPUTER
THE PET GRAPHICS KEYBOARD COMPUTER
& THE CBM 2040 DUAL DRIVE FLOPPY DISK
AT YOUR LOCAL COMMODORE DEALER.

THE CURSOR (tm) STANDARD INPUT SUBROUTINE

by Glen Fisher
Software Editor
CURSOR Magazine

As many people have discovered, Basic's INPUT statement has some 'features' which are not all that one would like them to be. For example, if you hit RETURN in response to an INPUT, you are dropped out of the program. It is possible to circumvent that problem by using a GET statement. However, an unadorned GET lets you have only one character of input, and has no cursor to tell you that the program is waiting for something. Also, you can't change your mind and take back that key you just pressed.

To solve those problems, we have developed a subroutine that reads a line of input and returns it to whoever wanted it. The subroutine supports editing of the answer, although it doesn't allow the use of the cursor control keys or the RVS key. Having used it in some 30 to 40 programs since it was written, we have found it to be quite useful, and the restrictions have been found to be not at all restrictive. Indeed, it has been so useful that we no longer publish programs using INPUT statements. Either they use the input routine, or they use GET statements. We prefer the input routine, however, and resort to GET's only when we are pressed for space and can't make room for the subroutine.

The special features of the routine are these:

Only visible characters are accepted; anything else is simply ignored.

The DEL and SHIFT-RETURN keys work as they do normally; DEL deletes the last character typed, and SHIFT-RETURN deletes everything typed on the line so far, albeit the input routine does it more spectacularly.

When the user hits RETURN, the input routine prints the variable CR$ (for Carriage Return). Normally, CR$ contains a CHR$(13). It may contain anything at all, however. Try it with various strings, especially the null string ("").

The input routine is capable of reversing the case of typed letters. If the variable FL (for FLip) is 1, the case of any letter typed is flipped, so that upper case becomes lower case, and vice versa. If FL is 0, everything is left as it was typed. (This feature is of use mostly on old-style PETs, for which lower case was shifted, and upper case was unshifted.)

The cursor which blinks to indicate that input is wanted is the grey square to distiguish it from the normal white square of INPUT.

The input routine is completely ROM-independent. It will run correctly on any PET in existence.

Finally, the input routine is in the public domain, so anyone may use it with no strings attached. (Naturally, we would like credit for it).

To use the input routine, enter it into your program at some convenient place. (We always put it at line 60000. We also keep it on a tape of its own so that we can easily add it to any program we're working on.) Then, whenever you need an input, GOSUB to the routine (GOSUB 60000,

in our case). When it returns, the typed string will be in the variable IN$. It does use some variables other than IN$ CR$, and FL, but they all begin with Z, so it is easy to avoid conflicts with your variables.

```
 0 REM THE '&' IN LINE 60020 IS SHIFTED.
 1 REM [LEFT] IS THE CURSOR-LEFT KEY.
 2 REM WE SUGGEST YOU REMOVE ALL THE BLANKS
 3 REM WHEN YOU USE THIS ROUTINE.
 4 :
 60000 IN$=" ":ZT=TI:ZC=2:ZD$=CHR$(20)
 60010 GET Z$:IF Z$<>"" THEN 60070
 60020 IF ZT<=TI THEN PRINT MID$(" &",ZC,1);"[LEFT]";: ZC=3-ZC:ZT=TI+15
 60030 GOTO 60010
 60070 Z=ASC(Z$):ZL=LEN(IN$):IF (Z AND 127)<32 THEN PRINT " [LEFT]";:GOTO 60110
 60080 IF FL AND (Z AND 127)>64 AND (Z AND 127)<91 THEN Z$=CHR$((Z+128) AND 255)
 60090 IF ZL>254 THEN 60010
 60100 IN$=IN$+Z$:PRINT Z$;ZD$;Z$;
 60110 IF Z=13 THEN IN$=MID$(IN$,2):PRINT CR$;:RETURN
 60120 IF Z=20 AND ZL>1 THEN IN$=LEFT$(IN$,ZL-1):PRINT "[LEFT]";:GOTO 60010
 60130 IF Z=141 THEN Z$=CHR$(-20*(ZL>1)):FOR Z=2 TO ZL:PRINT Z$;:NEXT Z:GOTO 60000
 60140 GOTO 60010
READY.
```

BITS AND PIECES

INDENTING PROGRAM TEXT — by B. Seiler

Many programmers like to indent their FOR-NEXT loops, to enhance readability. Up until now, this has only been possible by putting a colon (:) at the start of each line to be indented or spaced. For example:

```
10 FOR I=1 TO 10
20 : FOR J=1 TO 10
30 :   FOR K=1 TO 10
40 :
50 :     PRINT I,J,K
60 :
70 :   NEXT K
80 : NEXT J
90 NEXT I
```

This helps readability greatly, but you can go even further! By substituting SHIFTED(graphic) characters instead of colons, and using ▒ ▒ (graphic space graphic) to blank a line, the listing would be typed in like this (note: any shifted character can be substituted for the ▒):

```
10 FOR I=1 TO 10
20 ▒ FOR J=1 TO 10
30 ▒   FOR K=1 TO 10
40 ▒ ▒
50 ▒     PRINT I,J,K
60 ▒ ▒
70 ▒   NEXT K
80 ▒ NEXT J
90 NEXT I
```

This would list like this:

```
10 FOR I=1 TO 10
20   FOR J=1 TO 10
30     FOR K=1 TO 10
40
50       PRINT I,J,K
60
70     NEXT K
80   NEXT J
90 NEXT I
```

The same result is achieved, but the listing is cleaner. To use the screen editor, and retain this formatting, list the problem lines, put a ▒ after the line#, and edit as usual.

~~~~~~

Users' Directory &

Announcements

One of the major advantages in being a member of the PET USER'S CLUB is the ability to get hold of PET related Software and ideas. Although our Master Library of programs is now growing, we get frequent Software inquiries for a wide range of applications.

In this issue, we have included the current USER'S DIRECTORY, containing lists of people writing Software, importing literature or starting local PET Groups. If you would like to use your PET for fun and profit, why not offer personal tutoring in PET programming to new PET owners. Alternatively, if you require a program to be written for you, ask for contacts via the USER'S DIRECTORY. The possibilities are endless. Please write to the EDITOR, U.S. PET USER'S CLUB, at our current address below.

To include your name in the USER'S DIRECTORY, please complete the following form:
~~~~~~

TO: THE EDITOR, U.S. PET USER'S CLUB,
COMMODORE BUSINESS MACHINES
3330 SCOTT BLVD.
SANTA CLARA, CA 95051

NAME: _____

ADDRESS: _____

_____

SERVICES OFFERED/SPECIAL AREA OF INTEREST: _____

_____

_____

To include as many contacts as possible, we must restrict each USER to only one line of description.

COMMODORE reserves the right to edit or withdraw any entry.

LISTED BELOW ARE PET USERS WHO HAVE RECENTLY SUBMITTED THEIR SPECIALTY OR AREA OF INTEREST TO FURTHER COMMUNICATION WITH PET OWNERS THROUGHOUT THE UNITED STATES. IF YOU WOULD LIKE TO OFFER YOUR SERVICES TO OTHERS, PLEASE FILL OUT THE "USER DIRECTORY" FORM ON THE PREVIOUS PAGE.

NAME AND ADDRESS	SERVICES OFFERED/SPECIALTIES
Jeff Harvey Box 247/201 South Main St. North Syracuse, NY 13212	Tutoring new PET owners, Individualized Business Programs
Steve Lee 40-11 73 Street Woodside, New York 11377	Small Business Software and Personal games (32K)
David L. Mattis P.O. Box 162 Morton Grove, Ill 60053	Using IEEE-488 Port for industrial control.
Bennett A. Meyer 35 Barker Ave. White Plains, N.Y. 10601	Westchester PET User's Club Oriented toward hobby & Small Business Users.
Donald Ross 10 Elizabeth Place. Armonk, N.Y. 10504	Mathematics, Education
Ken Upcraft 1161 N. Ballenger Flint, Mi 48504	Education, Business Applications

---

THE LIST OF PET USER GROUPS LISTED BELOW IS BY NO MEANS COMPLETE. PLEASE NOTIFY US IF WE OMITTED YOUR GROUP.

Association of Personal Computer Users
5014 Rodman Rd.....Bethesda, MD 20016

Amateur Computer Group of New Jersey
Box 379.....South Bound Brook, N.J. 07076

Bambug
1450 53rd St.....Emeryville, CA 94608

JAPS-Jacksonville Area PET Society
401 Monument Road #177.....Jax, Florida 32211

Lawrence Hall Of Science, UC Berkeley
Computer Project, Room 254.....Berkeley, CA 94720

Las Vegas PET Users
4884 Iron Ave.....Las Vegas, Nev. 89110

Lincoln Computer Club
750 E. Yosemite.....Manteca,CA 95336

Long Island PET Society
Harborfields High School, Taylor Ave....Greenlawn, NY 11740

Madison PET Users
1400 East Washington Ave Rm 187...Madison, WI 53703

Northern New England Computer Society
P.O. Box 69.....Berlin, NH 03570

North Orange County Computer Club
3030 Topaz, Apt. A.....Fullerton, CA 92361

NW PET Users Group
2134 NE 45th Ave.....Portland, ORE 97213

Northwest PET User Group
P.O. Box 482.....Vashon, WA 98070

PACS PET User Group
La Salle College, 20th & Olnes Sts..Phila, Penn. 19144

PET User Club (CAPE)
2054 Eakins C7.....Reston, VA 22091

PET User Group
2235 Lakeshore Dr.....Muskegon, MI 49441

PET User Group C/O Meyer
35 Barker Ave.....White Plains, NY 10610

PET User Group
Texas A & M Microcomputer Club, Tex. A & M Tex.

PET User Group
P.O. Box 371.....Montgomeryville, PA 18936

PET User Group
2323 Washington Blvd.....Ogden, Utah 84401

PET User Group of Hypot
P.O. Box 69.....Berlin, N.H. 03570

PUG
7170 S.W. 11th St.....West Hollywood, Fla. 33023

PUG of Silicon Valley
22355 Rancho Ventura Blvd.....Cupertino, CA 95014

Sacramento PET Workshop
P.O. Box 26314.....Sacramento, CA 95826

SCOPE
1020 Summit Circle.....Carrollton, TX 75006

SPHINX
314 10th Ave.....Oakland, CA 94606

St. Louis Club
40 Westwood Court.....St. Louis, Mo. 63131

The Human Society--United PET Users
1929 Northport Dr. #6.....Madison WI 53704

Valley Computer Club
P.O. Box 6545.....Burbank, CA 91510

DEALER UPDATE

SINCE OUR LAST ISSUE WE HAVE SEVERAL MORE NEW DEALER'S TO ADD. PLEASE NOTE THOUGH, SOME ARE SIMPLY UPDATES AND ALL DEALERS HAVE BEEN ALPHABETIZED BY STATE.

ALABAMA

Rickles Electronics &
Communications Co.
2800 W. Michigan Blvd.
Gadsen, Alabama 35904

CALIFORNIA

International Institute of
Natural Health Sciences
7422 Mountjoy
Huntington Beach, CA 92647

Paradyne Consumer Electronics
404 Second St.
Davis, CA 95616

Service Radio Co.
1250 Crefthaven Dr.
Pasadena, CA 91105

COLORADO

Byte Shop
300 E. Foothills Pkwy.
Ft. Collins, CO 80525

FLORIDA

Continental Exports, Inc.
1401 N.W. 78 Ave. Suite 303
Miami, FLA. 33126

ILLINOIS

Financial Dynamics Computing
530 Park Ave.
River Forest, ILL. 60305

INDIANA

Ft. Wayne Electronics
3606 Maumee Ave.
Ft. Wayne, IN 46803

IOWA

The Computer Store of Davenport
4128 Brady St.
Davenport, Iowa 52806

KANSAS

Central Kansas Office Systems, Inc.
307 N. Main
Hutchinson, Kansas 67501

MINNESOTA

Schaak Electronics
2138 Burnsville Ctr.
Burnsville, MN 55337

MASSACHUSETTS

The Sound Co.
447 Sumner Ave.
Springfield, Mass. 01108

NEW JERSEY

Stonehenge Computer Co.
89 Summit Ave.
Summit, New Jersey 07901

NEW YORK

Digital Design
820 Willis Ave.
Albertson, New York

Logical Playback
10 Grace Ave
Great Neck, N.Y. 11021

Computer Strategies, Inc.
300 N. Main St.
Hillcrest Prof. Bldg.
Spring Valley, New York 10977

PENNSYLVANIA

A.B. Computers
115 E. Stump Rd.
Montgomeryville, PA 18936

TENNESSE

Micro Systems, Inc.
3100 Walnut Grove
Memphis, Tenn.

TEXAS

DALWORTH MICRO COMPUTERS
3103 Woodside Dr.
Arlington, TX. 76016

Kent Electronics
5626 Bonhomme Rd.
Houston, TX 77036

UTAH

Mobilite Computers
337 W. 200 South
Salt Lake City, Utah 84101

VIRGINIA

Computer Systems Store
1984 Chain Bridge Rd.
McLean, VA 22101

COMMODORE INTERNATIONAL OFFICES

COMMODORE BUSINESS MACHINES, INC.
3330 Scott Blvd.
Santa Clara, Calif. 95051

COMMODORE/MOS
Valley Forge Corporate Center
950 Rittenhouse Road
Norristown, PA 19401, USA

COMMODORE BUSINESS MACHINES LIMITED
3370 Pharmacy Avenue
Agincourt, Ontario, Canada M1W2K4

COMMODORE SYSTEMS DIVISION
360 Euston Road.
London NW1 3B1, England

COMMODORE BUROMASCHINEN GmbH
Frankfurter Strasse 171-175
6078 New Isenburg
West Germany

COMMODORE JAPAN LIMITED
Taiei-Denshi Building
8-14 1kue 1-Chomeasahi-Ku, Osaka 535, Japan

COMMODORE ELECTRONICS
(HONG KONG) LTD.
Watsons Estates
Blcok C, 11th Floor
Hong Kong, Hong Kong

Commodore Business Machines, Inc.
3330 Scott Road
Santa Clara, California 95050

Dated Material

Do not forward