SUPERPET GAZETTE

Editorial Offices
Post Office Box 411, Hatteras, N.C. 27943
Telephone: 919 - 986 2443 Business
986 2434 at Home

WHAT'S A STARTER-PAK?

It's an informal, step-by-step manual (with a tutorial disk), chock full of specific, concrete examples of how to use the 6809 side of SuperPET--a matter which the Commodore manuals neglect entirely.

After two years of using the 6809 side of SuperPET, we knew enough to write the Starter-Pak. You'll see how to:

Read and write disk files
Send the screen to printer or disk
Write to disk from program
Read disk files from program
Send disk files to the printer
Write to the printer from program
Determine your printer filename
Structure a filename (in English, with examples...)
Get input from the screen or keyboard
Edit files the easy way

Print any part of a file or program to disk or printer
Use the microEDITOR as a simple, built-in word-processor
Recover otherwise lost programs or languages
Handle the 6809 DOS commands easily
Use search and replace in the microEDITOR
Set them durn switches on the right side of the computer...

Use screen dumps to printer or to disk (including monitor dumps)
Send APL files to or from disk and to printer (if you can print APL)

And there's more we don't list above. The manual is 20 pages long, and covers the essentials. The disk contains a tutorial, which takes you through the fundamentals. In addition, the disk contains programs in every SuperPET language (except COBOL) showing the basic operations, and a few fun programs.

Also on disk is a universal dump from screen to any printer or to disk, in four versions. You choose which to use. You can dump a whole screen or just a few lines at any time--and it doesn't matter what kind of printer you have, or where it is hooked up. And it works in the monitor. How else do you save the stuff?

Attached are four pages from the manual, to give you an idea of what it has in it, as well as an alphabetical index to the disk.

You ask: Why no COBOL? Answer: My God, how many languages do you expect one man to use?

The Starter-Pak won't make you an expert in one day. But it will get you over the hump on basic SuperPET operations. Once you know the fundamentals, the rest is easy. And those #$%&!! handbooks which came with SuperPET will begin to make some sense.

For a copy, send $15.00 to ISPUG, at the address above. State disk format! Make checks out to ISPUG, in U.S. funds. We'll ship a disk and a manual the day your order arrives.

Dick Barnes
Editor

TABLE OF CONTENTS

The SuperPET STARTER-PAK

by Dick Barnes, Editor, SuperPET Gazette
PO Box 411, Hatteras, N.C. 27943

MEMO FROM: THE WHAT THE HELL DO I DO NEXT DEPARTMENT, SuperPET Gazette

TO: Ye Who Are About to Turn on the Power for the First Time

I remember all too well the rage and frustration I felt when I searched--uselessly--for the instructions on how to talk to my disk drives and printer, on how to print directories, on what codes handled the screen (and a thousand other small details) when first I tried to use SuperPET.

The manuals told me more than I ever wanted to know about manipulating matrices, but nothing at all about the fundamentals of how to use SuperPET.

This pamphlet and the accompanying tutorial disk are intended to remedy that deficiency.

So start reading. As you proceed, you'll be told when to turn SuperPET on and how to use the tutorial disk.

If, after using this material, you have any suggestions on what I should have covered (and didn't), or if anything is not clear--drop me a line.

[Note: when you read the disk copy of this material, you'll think me mad, for two marks will appear often in text :~ and ` . Know that they mark underline and boldface for my printer program--not a new system of punctuation.]

TABLE OF CONTENTS

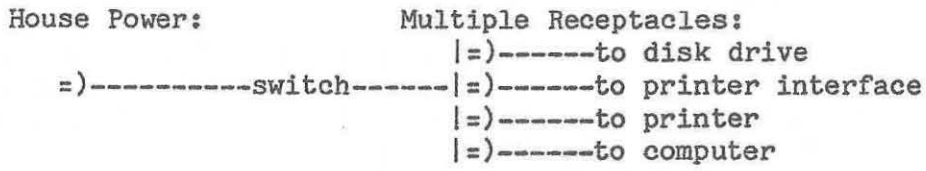
1 : INTRODUCTION

A : Before You Turn on the Power

If you've read the manuals and remain bewildered, be of good cheer. This pamphlet tells you all the things you need to know to start using SuperPET, even if the manuals don't.

(A Secret: the manuals are written in High Abyssinian Greek, an argot spoken only at universities and in the mountains of what was once called Ethiopia; only three living people fully understand it, and two of them are deaf and dumb.)

I assume you've read the Waterloo Systems Overview manual, have peeked at the language manuals, have thumbed through the Commodore manual on the disk drive you're using, and have got drives and computer hooked up. If you haven't connected your printer, computer, disk drives, and the transformer for your printer interface to a single, multi-outlet receptacle, I'd suggest you do so. With a single switch in the power line from 'house' power to the multi-outlet receptacle, you can turn everything off and on with one switch, like this:



Radio Shack sells a good multiple-receptacle outlet. You may want to buy one which filters voltage/power spikes and provides some protection to downstream devices. They're inexpensive.

B : The External Switches

Look on the right side of the lower, metal case. You'll see switches. There may be either two or four. If there are four:

1. Put the switch marked READ R/W PROG in R/W, which means Read or Write to the 64K bytes of bank-switched memory at address $9000.

2. Put the switch marked 6809 6502 PROG in 6809 position. This puts the 6809 microprocessor in control.

3. The remaining two switches control ROMs. Switch three controls socket UD12 on the lower board, which is designed for ROMs such as POWER, and which use the address of $9000-$9FFF. Turn it off in 6809 mode. Leave it off in 6502 if you want to use the upper 64K of memory from 6502. If you want to use the ROM in 6502 mode, turn it on.

4. Switch four controls socket UD11 (address $A000-$AFFF); ROMs for word-processing (such as PAPERCLIP and WordPro) go in this socket. Turn it off in in 6809, on in 6502 mode.

If there are only two switches: You may have a very early machine with three boards internally. You can tell by opening the case and looking. If so,

you need to have your dealer install two more switches, which work as I've described above. If you see only two boards inside, you have a late model, and need only two switches.

You probably can tell if you have a late-model, two-board machine by examining the mount for the two switches. Is it monolithic (one mount for two switches)? If so, you probably have a two-board machine and need only two switches.

On all two-board models with two switches: ROMS are controlled by a POKE in 6502 mode, not by external switches. See paragraphs 1 and 2, above, for proper setting of the two external switches.

[Note for later: On two-board models, in 6502 mode, the ROM $A000-$AFFF socket, for WordPro, etc., is on the upper board at U46, and is always on in 6502. The $9000-$9FFF socket, for POWER and such, is at U45 on the upper board, and is turned ON with POKE 61438,1 or OFF with POKE 61438,0. It boots up OFF in 6502. You cannot use the 64K of bank-switched memory if U45 is poked ON, because, of course, it has the same address as bank-switched memory: $9000.]

C : The REFERENCE SHEETS

Before you proceed, look at the REFERENCE SHEETS attached to this pamphlet. There are four, and they summarize all DOS Commands, all microEDITOR commands, Search, Search/Replace, mED metacharacters for Search and Replace, printer filenames, and the general structure of SuperPET Filenames.

Punch them for a 3-ring binder and put them where you can get to them. You will need them frequently, for reference, as you proceed.

D : On Printers

I strongly recommend that you not put your printer on the serial port. That port is too useful for telecommunications. Moreover, you'll find no word-processing programs that can deal successfully with the 6551 ACIA chip in SuperPET (so you cannot print from them); last, you'll screw around setting baud rate and other parameters for the serial port to match your printer--and, since the port is not buffered, you'll probably drop some characters when you do get set up. Stick with the user/ieee-488 port. It works splendidly.

E : Loading Languages

Turn on power to your disk drives, computer, and printer.

You should see a menu of languages on the screen; if you don't, make sure the 6809/6502 switch is in 6809 . If that doesn't work, consult your dealer.

If the menu is there, put the language disk in drive 1. Commodore doesn't mark the disk as a 'language disk'; it labels it 'COMMODORE. Use w/8050 (or Use w 4040) WCS'. (Great label. Tells you a lot.) If you now press a lower case 'e', and hit the RETURN key, the loader will load the microEDITOR from drive 1 into RAM. (microBASIC is loaded with: b <RETURN>, microPASCAL with: p <RETURN>, etc. COBOL must be loaded by typing COBOL in caps before you hit <RETURN>.)

Should you have only one drive (poor thing), you can trick the loader into loading from drive 0 (drive Only!). Simply preface your command with: disk., as shown below. This trick also works with two drives when you want to load from drive 0:

```
disk.e  <RETURN>      [Loads the microEDITOR)
```

If, on the other hand, you have two separate drives, and the second is device 9, you can handle the languages from either drive of device 9:

```
disk9/1.e   <RETURN>  or   disk9/0.e   <RETURN>
```

Anyway, load the microEDITOR (which we'll call the mED from here on out), because we're going to use the mED to explain how SuperPET works. (If you should get a blank screen after trying to load the mED, take a look at your R/W switch. Is it in READ? If so, you can't load anything into the upper 64!)

F : In the microEDITOR

You should see on the screen a message identifying the mED and asking you to hit <RETURN>. Do so. You'll then see:

```
<beginning of file>
<end of file>
```

and you'll have two cursors, one on the 'file' lines above, and one on line 24 of the screen. The upper cursor is the 'screen cursor'; the lower one is the command cursor, where (suprise) you enter commands to the DOS and to the mED.

The Programmed Function (PF) Keys:

Press SHIFT and touch KEYPAD 8. The command cursor will disappear. You now know what PF 8 means--programmed function key 8--and what it does: takes you out of command mode into editing or screen mode. Touch PF0 (SHIFT KEYPAD 0). Aha! A new, blank line appears between <beginning of file> and <end of file>! Type from the keyboard. Text appears on that line. Hit <RETURN> and then touch the ESC key: a line is erased from cursor to end-of-line (EOL). Here's what the useful PF keys do:

```
PF 9 and PF .   Move up-text/down-text one full screen
PF 6 and 3      Move up-text/down-text one line at a time
PF 8            Puts you in SCREEN mode
PF 5            Puts you in COMMAND mode
PF 2            DELETES the line the screen cursor is on
PF 0            INSERTS a new, blank line just below screen cursor
```

2 : ON MANIPULATING TEXT, DOS COMMANDS, DISK FILES AND SUCH

A : How to Format and Back up a Disk

Now, press PF 5. The command cursor will reappear on line 24. We're going to 'new' a disk, using the DOS commands from the mED, and then put to and get from disk some files. Put an unformatted disk in drive 0, and at command cursor enter:

g ieee8-15.NO:tutorial,tu [Use no quotation marks]

Hit <RETURN> and you'll 'new' a disk with the name 'tutorial' and ID of 'tu'. In the command, we tell the DOS (Disk Operating System) to open a channel to device 8 (our disk drive), and to the command channel (No. 15). Note the 'N' for 'NEW' is entered in capital letters. While the disk is formatting, take a look at both sides of REFERENCE SHEET 1, which covers DOS commands. Any DOS command shown on page 2 of that sheet may be entered exactly as we did it above.

B : The Universal microEDITOR

At this point, you wonder why I start you off in the mED. Easy: it runs, with some changes to adapt it to the language, in every language of SuperPET except APL. In all of those languages except microBASIC, you write and edit your programs in the microEDITOR; in microBASIC, you'll find the mED a superb program editor, far more versatile than the general screen editor of micro-BASIC itself. And, by itself, the mED is a pretty good text-processor. What you see on screen is exactly what you'll print to printer. You need nothing else for word-processing if your needs are limited to occasional letters. And, if you develop a text-processing program in one of the languages, you'll find the mED very useful for integrating programs (taken straight from disk) with text which explains them. I should know: I publish the SuperPET Gazette exactly that way. (For more information, see pp. 95 ff, Vol. I, No. 8, SuperPET Gazette. You can convert WordPro and PAPERCLIP files to ASCII files you can read in 6809.)

C : DOS Commands the Easy Way

By this time, your disk should be formatted. Put the tutorial disk which came with this pamphlet into drive 1 in place of the language disk. Enter the following command at command cursor:

g ieee8-15.CO=1 <RETURN>

This is a COPY command; it copies everything on drive 1 to drive 0. Note that the destination drive always is left of the = sign. You may wonder why I had you format a disk and then copy to it--well, I wanted to show you how. You could have both formatted and copied the disk with a single command, shown below:

g ieee8-15.DO=1 <RETURN>

This is a DUPLICATE command; unfortunately, some disk drives get hiccups when you back up in one step, so it's nice to know how to get around the problem. Format first, then copy. Destination always is left of =.

You now have a backup copy of the tutorial disk in drive 0. Take the original out of drive 1 and put it somewhere safe, away from dust and heat.

D : Working with Disk Files:

Take the backup tutorial disk out of drive 0, and put a write-protect tab over the small notch on the left side of the disk (left as it comes out of the drive). Then put it back in drive 0. We can now read it, but we cannot write to it.

We need a scratch (or learn-how) disk to practice on. So, put a new disk in drive 1, and format it:

g ieee8-15.N1:practice,pr

While you're waiting, review reference sheet 2; it covers general commands in the microEDITOR. We're about to use 'em. Disk 1 formatted? Okay, let's get a file from drive 0, the first half of this tutorial: [Note: the second half is filed as tutorial.1]

get tutorial:e <RETURN>

The first half of this pamphlet should load into mED. After it has loaded, hit PF 5 (to make sure you're in command mode), and enter, at command cursor:

$ <RETURN>

You will find yourself at end-of-text; $ stands for 'end'. Then enter:

-60 <RETURN>

and you'll find the screen cursor 60 lines back of end-of-text. Next, enter:

1 <RETURN> [Use the number '1', not the lower-case letter 'l']

and you'll find you are back to line 1 of this pamphlet. Well, you've lost your place.... So type in: 354 <RETURN> and you'll be right back here. Next time you leave a spot in a program or in text, get the command cursor with PF 5 and ask it: # <RETURN> (Pray tell, O SuperPET, what line is the screen cursor on?) The line number will appear on line 25 of the screen. At this point, I suggest you start reading the screen, not from hard copy--at line 354.

All the commands for the microEDITOR are in Systems Overview (if obscurely); you'll find a summary in REFERENCE SHEET 2. Try them now; if you louse up the screen copy of 'tutorial', you can always get another copy from disk. So, <u>try everything</u> --right now, on this text, on this screen.

E : Let's Handle Some Files

All files created by and read by the microEDITOR are SEQUENTIAL files; the word 'sequential' means that data are written in the sequence received, and any reading of the file returns data in the sequence written. Moreoever, all such files are ASCII representations of what was in RAM. PROGRAM files are not ASCII files, but rather a direct copy of what was in RAM. The mED can neither read nor create such files. (If you are confused by the multiplicity of file types SuperPET can use, welcome to the club. See Refsheet 4, p. 2. Read the article on files in the <u>SuperPET Gazette,</u> pp 117 ff, Vol. I, No. 9.)

<u>Coding of Files by Language/Facility:</u>

You ask, "Hey, what's the ':e' on those files on the disk?" With SEVEN languages/facilities in SuperPET, you must identify the files. So, :e means "Read it in the microEDITOR, Henry," :b means "Load it in microBASIC," _p stands for microPASCAL material (':' is fatal there), :a or :aws stands for APL material,

:c for COBOL, and :f for FORTRAN.) I would strongly (well, I SCREAM on this one) advise you to never do your DOS work on 6809 files in 6502 because you know the old BASIC 4.0 DOS commands. If you'd been driving a horse and buggy, you would not expect a new Mercedes to respond to 'giddup' or 'whoa'; if you attempt to do your DOS work and/or file identification in 6502, you will find all your 6809 filenames in capital letters (and maybe in reverse field, which you cannot handle in 6809). You will not be able to tell 6809 from 6502 files, or where they run, or what language they are in. Learn the 6809 DOS commands and you will never need to drop into 6502--AND you can identify your files in 6809. If you're as lazy as I am, you'll do all your 6809 filenames in lower case, so you can identify them, and because--well, why SHIFT? Need the exercise?

In any event, don't ever send me a disk in 6809 with all the filenames in CAPITAL letters and no identification of the language used. I'm not about to work through the SEVEN languages/facilities (Assembly language is the seventh) to find out where the stuff runs. If it comes in, my hungry wastebasket eats it.

Creating TEXT,SEQ Files:

That out of the way, let's create some TEXT, SEQ files. Enter, at command cursor (be sure your practice disk is in drive 1):

```
put disk/1.tutorial.bak  <RETURN>
```

and a copy of the material in mED's memory goes to drive 0 under the filename of 'tutorial.bak'. While still in command mode, enter:

```
*d <RETURN>
```

and the screen copy of 'tutorial' is deleted from memory. Get it back into mED with:

```
g disk/1.tutorial.bak  <RETURN>
```

Yes, 'get' can be abbreviated to 'g'. When you have 'tutorial.bak' in memory, file the first 80 lines to drive 1 with:

```
1,80 p disk/1.tutorial.start  <RETURN>
```

and you'll put the first 80 lines of text to the indicated file on drive 1. Now, enter:

```
80,$ p disk/1.tutorial.last    <RETURN>
```

and you'll file line 80 through end-of-text to the file indicated.

Now, let's try a few filenames with spaces or blanks in them. Here, we must enclose the commands in quotation marks " or in apostrophes ' (SuperPET doesn't care which). Try this one:

```
20,40 p "disk/1.twenty lines"   <RETURN>
```

You need to use quotes only when there are spaces in filenames, or when you want apostrophes in the filename, as in:

1,5 p "disk/1.Henry's file"

At this point, see Reference Sheet 4, page 2, on printer filenames. Give the last few commands above to your printer.

Scratch the mED's memory again, with: *d <RETURN> and tell drive 1 you want another copy of tutorial.bak. As it starts to come into memory, hit STOP. You've just stopped the load, and only the first part of the file is in memory. When you want to identify a file, that is a quick way to get just the beginning.

I'm sure you get the idea. You have the REFERENCE SHEETS in front of you; try the COPY, RENAME, CONCATENTATE, and VALIDATE commands. Put more fragments of this pamphlet to drive 1, and manipulate them.

Remember that you use the same commands, in exactly the same way, when you handle files in all SuperPET languages where the mED is used. So master the commands now. Even if you wipe the disk in drive 1, the backup tutorial disk has a write-protect tab on it (I hope!), so you can get more copies.

F : The MOUNT Command:

If you have an 8050 or 8250 drive, stop. You don't need MOUNT (which is the same as the INITIALIZE command in 6502). If you have any other drive, MOUNT is necessary, for it lets the drive know what disk is in it. Here's how you give it, at command cursor:

mou disk <RETURN> (For drive 0)

mou disk/1 <RETURN> (For drive 1)

Go ahead: try the MOUNT command on drive 1. It won't hurt a thing. You'll note the red LED comes on as the drive 'initializes.'

G : Inserting Text at Screen Cursor:

You should have some small fragments of the tutorial on drive 1: how about 'twenty lines'? You can insert that file (or any other on disk) wherever you want it in text. Put the screen cursor on this line, hit PF 5 for command mode, and say:

g 'twenty lines' <RETURN>

The disk file will open, the screen will flicker, and file 'twenty lines' will insert in the mED's screen file, starting on the line below the screen cursor. This is a very handy technique both for moving text and for revising programs (or for copying code from your library into a new program).

3 : DIRECTORIES : A FEW GENERAL BUGS

A : Handling Directories on Screen

See REFERENCE SHEET 2. Go into command mode with PF 5, and say:

di <RETURN>

and you'll get a directory of the backup tutorial disk. When the screen fills, the directory pauses so you can read it (some improvement over BASIC 4.0!); you can start it again with RETURN or with STOP. In addition, you can abort any directory call by hitting STOP several times in rapid succession.

B : Copying and Editing Disk Directories

If you want a disk copy of a directory, so you can call it to screen and edit it, or comment it, try: (The following will work ONLY in Version 1.1 Software!)

```
di disk/1 disk/1.index <RETURN>
```

This command will force drive 1 to make a copy of its own directory as the disk file 'index'--on drive 1. Clear the mED's memory with *d <RETURN>, and get the file 'index' on the screen with:

```
g disk/1.index
```

After you edit it (or don't, as you wish), refile it to drive 1 as the file 'index'. No, you needn't scratch the old file. SuperPET will overwrite it. Unlike BASIC 4.0, SuperPET overwrites files without a murmur (or a warning), so be careful when you file. You may overwrite a file that you want.

Now, let's put another directory to disk; this time, we'll file a directory of drive 0 to drive 1:

```
di disk disk/1.tempo
```

After the drive lights have gone out, get the directory of drive 1 with:

```
di disk/1
```

Notice anything unusual? Is the file 'tempo' marked with an asterisk, just in front of *SEQ? It may be. It seems to happen on some drives (yes, a bug). The asterisk indicates an improperly closed file. Don't ever SCRATCH such a file! Either overwrite it (as we did above) or remove it from the disk with a VALIDATE command, like this (assuming it is on drive 1):

```
g ieee8-15.V1
```

C : Sending Directories to Printer

Sorry, but it's pretty hard to tutor on sending directories to printer when I don't know what type of printer you have, or where it is hooked up. So see page 1 of Reference Sheet 2, which summarizes all the commands for sending a directory to a printer. Choose the format which fits your printer, and try a few commands.

If you should happen to have Version 1.0 Software, you won't be able to handle directories as I've outlined above. V1.0 has no directory printout or copy capability. Version 1.1 does--which is a good reason to have V1.1!

D : Another Occasional Bug in DOS Commands

Waterloo did not design the SuperPET operating system to use 'g ieee8-15.' Jim Swift, a Canadian, discovered that all DOS commands could be so entered. Without that trick, we'd have no universal way to enter DOS commands from the mED, wherever used. If you don't use 'g ieee8-15.', you'll find you cannot enter DOS commands in mPASCAL or mCOBOL without a program in that language, which is, in both languages, a large pain in the behinder. In microBASIC, if you did not have 'g ieee8-15', you'd have no way but the following clumsy and slow method to enter DOS commands in immediate mode:

```
open #10, 'disk', output
print #10,'N1:filename,fn'        (To new a disk on drive 1)
close #10
```

So you can understand why all of us love Jim Swift. There is, however, an occasional bug in 'g ieee8-15.' It is harmless, but if you don't know about it, it can really bother you. Once in a while, after you've given a DOS command, the red "drive in operation" light on the drive won't go off. This happens most frquently with VALIDATE or INITIALIZE commands. If this happens to you, hit the STOP key. The red light will go off; you will see a lot of '00, OK,00,00' lines on the screen. Delete them or erase them. Neither your disk nor your screen file will be harmed. And the DOS command will have been executed.

If you know that the problem can occur, and how to solve it, it is no problem at all.

E : The <end of file> Problem in the microEDITOR

I'm going to show you how to create the bug, and how to avoid it. With anything longer than one screen page in the mED (get a file into it, preferably a long one), do the following:

Get down to <end of file> with: $ <RETURN> at command cursor.

Hit PF 8, and put the screen cursor ON repeat ON the line: <end of file>

Hit PF 0, and insert a new, blank line.

See anything? No? Well, cursor up-text with PF 9 until the new, blank line disappears below the bottom of the screen. Now cursor back with PF . (period). Lo! A line from previous text has been copied to the file! You couldn't see it, but it was there; had you put the mED file to your printer, it would have printed in hard copy. Okay, we know about the problem. How do you avoid it?

F : Insert Mode in the microEDITOR

Easy: work in insert mode when you are at end-of-file. You enter insert mode by hitting PF 5 (for command mode), and typing: i <RETURN> at command cursor. Now, each time you hit RETURN, a new, blank line opens up--and that new line will not copy a false line. Simple enough, once you know about the bug. You may also avoid the problem by hitting ESCAPE when you open a new line with PF 0, or by entering any text and hitting RETURN. Easy.

Sorry to trap you in insert mode--you can can get out of it by using UP or DOWN cursor, or by typing a period all by itself on a new, blank line and hitting RETURN.

G : Truncation - A Limit

This isn't really a bug, but when you run into it you'll think it is! Refer to Reference Sheet 1, page 2. When you handle very long filenames, the DOS command may exceed its limit of 40 characters. The reference sheet shows how you handle the problem--by copying, renaming, or concatenating files to a new, short name. When you're through, you rename the file to its original verylongfilename. Try a few renames on the files on drive 1.

H : Recalling Previous Commands : Re-execution

You may recall any command issued at command cursor by typing: ? <RETURN> You may then re-execute it with RETURN. But there is a bug and danger! Any function keys but LEFT/RIGHT cursor and INSERT/DELETE will cause the command to be re-executed! If you just finished printing six pages to printer.... Be careful. Test the bug by recalling a previous command; then touch UP/DOWN cursor. Bingo! The command is re-executed.

You will find another useful 'recall' in SuperPET. Touch: n <RETURN> at command cursor. The last file command you gave will appear on line 25. If you cannot remember a filename, this may help you. ('n' is short form for 'name'.)

4 : INPUT/OUTPUT WITH DISKS, FILENAMES, AND SUCH

A : On SuperPET Files

Before you proceed, I suggest you look at REFERENCE SHEET 4, particularly at the section on the structure of SuperPET filenames. I also suggest you read the definitive article on SuperPET files starting on page 117, Vol. I, No. 9, of the Gazette.

In general, you must distinguish between a FILENAME and a FILE-DESIGNATOR:

In SuperPET, a filename includes the device to which or from which you want to get or put a file, as below:

```
  g disk/1.barbarians                  p printer (what is in RAM)
     /        \                           /              \
 device      file-designator          device      implicit file-designator
```

File-designator, as you may have gathered, is what we wrongly call 'filename'; it is the TITLE, or NAME, of the file. In Waterloo terms, the 'filename' is the entire package: device AND file-designator.

B : Input/Output with Disks

Once you grasp this, it is easy to write to, or to read from, disk files. In all languages, the general method is as follows:

1. Open a Logical File Number (LFN) to the disk drive. We've used numbers up to 1500 to do this, without trouble. Example, in microBASIC:

```
open #200, 'disk/1.example', input  (We're going to read file 'example')

linput #200, line$  (We input a whole line at one gulp)

print line$         (We print it to screen)
```

2. If you want to print to another file, such as a printer or to another disk:

```
open #100, 'printer', output     (Open a communication line to printer)

open #50, 'copyfile', output     (Open another to disk 0, for 'copyfile')

print #100, file$                (We print line$ to printer)

print #50, line$                 (We print line$ to the disk file)
```

3. Close the files, either with a: close #100 (for example), or, if you're using Version 1.1 mBASIC, with a single command which closes ALL files: reset. The 'reset' command is valid in both immediate and program modes.

Though each language will vary in the exact phrasing of the commands, the general method works in all SuperPET languages, including APL.

C : On DOS Channels and Logical File Numbers:

Please do NOT confuse the 'channels' of the DOS (of which there are 15) with the LFN's (Logical File Numbers) which I call 'lines' (for communication lines). When you 'open #10', you open a communication line numbered 10 to a specific device--just as if you'd made a phone call to a specific number. All subsequent messages to that device must be sent to that number. If you had Uncle Harry on 323 4567, you would not give him a message for Aunt Mabel, who is on another line at 435 4444. Think of the "#'s" as telephone numbers and you won't go wrong. You must open and close them, just as you dial and hang up a phone. DOS channels are <u>not</u> LFN'S.

As a matter of fact, the DOS automatically assigns an LFN (opens a comm. line) whenever you use g ieee8-15. It is also done automatically in SuperPET whenever you employ any command to your files. See Reference Sheet 4, page 2.

D : A Sample Input/Output Program

Following is a simple program in microBASIC, which reads a file, revises it, and sends output to both printer and to a disk file. Note that we ignore the end-of-file marker in the input disk file because we surely don't want to print it, and sense EOF with an intrinsic function: io_status, which will be zero until we reach EOF, when it becomes 2 (it can be 1, which means an input/output error--a bad file, a bad disk, an incorrect filename, etc.)

```
100 open #100, 'disk/1.example', input        ! We read this file.
110 open #200, 'printer', output
120 open #300, 'new.example', output          ! New disk file.
130 on eof ignore
```

```
140 loop
150   linput #100, line$
160   if io_status <> 0 then quit  ! This line ALWAYS must follow the input
170                                ! request--or you'll never quit the loop.
180   line$ = line$ + 'test'       ! We modify the input.
190   print #200, line$            ! Send modified output to printer.
200   print #300, line$            ! And send it to new disk file.
210   print line$                  ! And send it to screen as well.
220 endloop
230 reset : stop
```

In general, you will find the 'linput' command a great improvement over either 'input' or 'get'. 'Input' gets data up to the first comma in the file. 'Get' gets one character at a time. Both have their uses, but are slow. Get, in particular, is like molasses in ice.

You can have a number of communication lines open to one disk (I think five is maximum, but have never used that many); you may have lines open to printer and both disks simultaneously, some for input, some for output. In one program, I have two input channels open to the same disk at the same time, two output channels open to the other disk, and one line open to printer.

E : A Note on Input Statements in Program : Differences from BASIC 4.0

There's a substantial difference between microBASIC and BASIC 4.0. First, a RETURN entered as a reply to an input request will not take you out of program. RETURN is a valid input response. I often use it in printfile programs to my printer, like this:

```
130 input "At end page--change paper and hit RETURN", o$
```

You throw o$ away. The method simply pauses the program until my cut-paper is changed and DIABLO is ready to print again. You'll also find RETURN is a good substitute for a 'No' response to a prompt. (Note that BASIC 4.0's semicolon at the end of an input statement has become a comma.) If you're accustomed to using 'get' in program to avoid RETURN problems, forget it. It is not needed.

When you want the operator to tell a program what file to open, using an input statement, here is a simple way:

```
200 input "What file do you want printed? ", file$
210 open #50, file$, input
```

'File$' must include the disk the file is on. You could respond to the input statement (for a file designated as 'example') in several different ways:

disk/1.example	(If the file is on drive 1)	Notice no quotation marks or apostrophes are needed for the input response.
example	(If it is on drive 0)	
disk9/0.example	(If the file is on another set of drives, device 9)	

There is a default, in SuperPET, to drive 0, device 8. You need never type it in. On the middle example, above, you could have said: 'disk/0.example', but it isn't necessary.

Other than the structure in microBASIC, there are some other differences: (1) # need not be crammed tightly against a 'print' statement. In BASIC 4.0, you must say: print#3. In microBASIC, you can cram, but it's better to say: print #3. (2) Programs using integers, especially in loops, run almost twice as fast in microBASIC as do floating-point values. Example:

```
 90 one% = 1 : number_items% = 1000
100 for i% = one% to number_items%
110   item%=i%*value%
120 next i%
```

In SuperPET, integers are stored in two bytes; floating-point values in five bytes. Whenever speed is essential, use integers. Programs will run faster even when 1 and 0 are converted to integers, as in: zero% = 0 : one% = 1 . Unfortunately, you don't have integers in microAPL; you do have them in all other SuperPET languages.

In SuperPET, you must 'close #20', though in old BASIC you 'close 20'.

F : On Cramming Code and Variable Names

Next, don't take spaces out of SuperPET programs, in the vain hope that you will thereby speed up your program (and ruin your eyes!). If you take ALL the spaces out of a microBASIC program, including the spaces for indented structure, you'll decrease run time only 2.6%--at a horrible cost in readability.

Now, let us praise God: You can use variable names through 31 characters in length--and they run just as fast as variable names of one character, believe it or not. DO NOT write programs with short, hard-to-identify variable names. Let them identify themselves clearly. If you're dimensioning a list of 1000 items, call it: number_items, or number_of_items or num_items (if you don't like to type), but for the love of Allah, don't call it 'n'! In a week, you won't know what 'n' means--and neither will anyone else.

If you now mutter: "I can't cram those long varible names onto one line!", cease muttering. You can continue them to the next line, like this:

```
100 interest_charged = (interest_rate * principal_remaining)/12 + &
110 & late_charge + (penalty * days_late)
```

The continuation ampersand '&' takes care of the problem in Version 1.1 if it is the first/last non-blank character on a line. See the microBASIC manual. You can even continue with '&' when there are comments on the lines!

G : Notes on Debugging Code

SuperPET languages run debuggers of varying capability. That in microBASIC is by far the best of the lot. At first, it may confuse you, so here's what to expect. If, for example, you leave off one parenthesis on a statement, the program will PAUSE (not stop), and say:

```
**Expecting )
   330 interest = principal*((interest_rate/100)/12
```

Note that the bad line is printed to screen, and that a carat ^ is found at the point of error. You may insert the necessary paren, hit RETURN, and continue your run with: cont <RETURN>. That is a vast improvement over the frustrating 'SYNTAX ERROR' of BASIC 4.0! SuperPET will even keep disk files open when such errors occur. In short, you debug and keep right on running. You'll find debugging arrangements in all the languages.

Several languages contain PAUSE statements. These will halt the program (but do not STOP it) and print the PAUSE message to the screen. You may insert such statements wherever your code is giving you trouble; halt the program, ask for the value of variables, and then continue with: cont <RETURN>. You may even (shades of Allah!) list a part or all of your program to the screen and revise the code, and then continue! Read the language manuals carefully on debugging. The features are invaluable.

5 : THE DISK TUTORIALS FROM WATERLOO:

Each language manual for SuperPET is prefaced with a number of tutorial examples. The manuals don't tell you this, but all the examples are on the second disk you received, marked:

COMMODORE
use w/8050 Floppy (or w/4040)
Tutorial

MicroBASIC examples are identified as bex.nn (where nn is the example number; mPASCAL examples as pex.nn, COBOL examples as cb.nn, etc. Unfortunately, the disk does not contain any of the tutorials for Assembly language.

You learn to program by programming. Revise the Waterloo examples and watch what happens. The tutorials are well done, and you'll learn a lot by using them.

6 : NOTES ON THE LANGUAGES

A : A Patch for microBASIC Version 1.1

MicroBASIC Version 1.1 has a bug in it, as issued. Version 1.0 is okay. If you have V1.1, read the disk file: patch:e for an explanation of the problem and for instructions on how to run the patch program on this disk--which fixes the bug. The program itself is listed as: mbasic_patch2.

B : APL Version 1.1

Waterloo, in issuing APL Version 1.1, made some major changes to the language. Conversion of workspaces from V1.0 to V1.1 is a headache. I strongly recommend that, if possible, you start out with Version 1.1 APL to avoid the problem.

C : Some APL Screen Dumps & DOS Commands by Menu

APL DOS Commands are a pain to enter; an ISPUG member named Reg Beck wrote a nice APL program, menu-driven, which handles all DOS commands in APL. Load it into Workspace with:)LOAD DOS:AWS. You then get a menu. Follow the instructions on screen. This program makes handling the DOS commands in APL a pleasure.

If your printer can handle the APL character set, you may want to use two APL functions on this disk, filenamed: SDUMPS:AWS. One of them, DUMP, will send the screen to printer from line 1 to the line above the cursor. It's set for an 'ieee4' printer filename, but you can change that if you need to. The second dump sends to disk an image of the screen in external representation (this means that overstruck characters are sent as a character, a backspace, and another character). Both dumps are useful for letter-quality printers (daisy-wheelers).

Instructions are on disk, filenamed: apl_dumps:e. You may find these handy when you're learning APL, since you must write functions to send anything in APL to printer without them.

E : The Waterloo Roman Font on Screen

For those curious about the ASCII code assignments on the keys, see the program: get_keyboard:bd (b for mBASIC, d for DEMO) on disk. It runs in microBASIC, and will send the name and ordinal (read ASCII code number) of each key to your printer; you tell the program what key you will press; the program sends its name and ASCII ordinal to screen and printer.

Also, find a short demonstration of what the ASCII codes from 1 through 12 do to SuperPET's screen: try_codes:bd runs in microBASIC and shows you exactly what happens. (Note: a SEQ file is called off disk into mBASIC with:

old 'filename' <RETURN>

A program file, in distinction, is loaded with: load 'filename' <RETURN> .)

Next, you can view the Waterloo Roman character set (including the graphics characters which can be keyed from the shifted keypad) by running the tiny program on disk as: pokescreen:bd, in microBASIC.

F : Some Programs : The Construction of Guess...Endguess

Last, I've put some simple programs in mBASIC, mFORTRAN, and mPASCAL on disk as demonstrations. They supplement Waterloo's tutorial programs. You may be interested in the disk file: guessdemo:bd, which demonstrates what is not too clear in the manuals: how to use the guess...endguess statement.

7 : DISK DRIVE OWNER MAINTENANCE : AN 8050 BUG AND THE CURE

Sometimes (and usually with Tandon-made 8050 drives, which have a top-hinging door) the drives will refuse to load anything when you first turn on power. This usually happens when the last access to the drive was to a track larger than 55. First, you'll get a 'DRIVE NOT READY' message, though the green light is on, and disks are in both drives. If you again attempt to load something from menu, you'll get an 'IO TIME OUT' message. There are two ways to solve this:

1. Prevent it: Read a directory from both drives before you shut off power. This insures that the R/W head on the drive is in the right place when you next turn on the drives.

2. Cure it Enter the monitor from menu with: m <RETURN> and enter the following code at the > prompt in the monitor:

```
>m 1000 cc 10 07 bd b0 e7 3f 64 69 73 6b 2f 30 00  <RETURN>

>g 1000 <RETURN>
```

The code above generates the MOUNT command (or INITIALIZE) for drive 0. To kick drive 1 back to proper operation, change the ending 30 00 to 31 00, hit RETURN on the code line, and give the: >g 1000 again. Both drives should then work (it fixes my drives nicely).

If the trick should fail, lift the lid to the drives, remove the board above the drives (three screws), and gently move the R/W heads toward the front of the drive. I've done it, but only as a test. The monitor code has always pulled me out of the bug.

Corrosion Bugs:

I live near the sea, and salt-mist corrosion accelerates what will happen anywhere, given enough time: The pin-and-socket connectors in the computer, but more often in the disk drives, will corrode just enough to cause mad things to happen. Disk drive motors will not run; drives won't obey DOS commands, etc. Cure: every once in a while, lift the lid on the drives, gently remove the top board (you'll find it hinges up toward the rear with a WOODEN prop to hold it there), and:

1. Clean the ways (circular, shiny steel) with a cotton swab lightly covered with instrument oil. The R/W head moves on these ways. Move it back and forth, and get rid of the crud. Change swabs frequently, until they come off CLEAN.

2. If your drive is a Tandon, and begins to make HORRID noises, put a very small drop of instrument oil (about 20-weight) on the point of a hatpin, and carefully oil the shaft of the motor which drives the R/W head. The oil must go just where the shaft emerges from the motor housing. After about an hour of subsequent operation, the horrid noises will dwindle to a murmur. BE CAREFUL with the oil! Protect the drive, get rid of ANY excess, and use the utter minimum. You need to put on only a tiny drop.

3. Gently move all pin-connectors back and forth (or up and down) to wipe off all corrosion and to get good contact. Don't miss a one. Do not take the connectors apart, simply move them about ten times.

4. Get rid of the dust in the case with a vacuum cleaner (use ONLY a plastic head on the hose, not a metal one!).

5. Brush the dust off the electronic board with a fine, clean camel-hair brush (and vacuum that up, too). You need a half-inch brush with long bristles.

With this regimen, we've gotten our Tandon drive to operate for over two years without a service call, despite the fact that it quit on us over ten times. The cleaning and wiping of contacts always has put it back in operation. I'd suggest that you have a dust cover for your drive, and that you use it whenever the computer is off.

When the Keyboard Bounces:

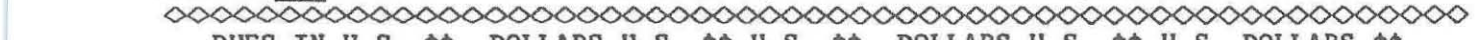
Sooner or later, your keyboard will start to bounce: Press an 'a' and you'll get 'aa' on the screen--or the SHIFT key won't shift. Send me a self-addressed, stamped envelope and I'll send back instructions on how to fix it.

8 : DON'T FORGET US — ISPUG

This pamphlet and disk were designed to overcome the hurdle of manuals which don't tell you the fundamental things you need to know to get started with SuperPET. It could have been a book (and have taken a year to write). I limited it to the gut essentials to get it written. For more information (every two months) from all over the world, on SuperPET, join ISPUG (International SuperPET Users Group). Dues are $15.00 U.S. per year in North America, and $25.00 U.S. elsewhere. Send an application and your check to:

Paul V. Skipski, Secretary ISPUG, P.O. Box 411, Hatteras, N.C. 27943.

You'll receive the SuperPET Gazette, which covers the waterfront on SuperPET, and is the resource for details on SuperPET. All back copies are available.



DUES IN U.S. $$ DOLLARS U.S. $$ U.S. $$ DOLLARS U.S. $$ U.S. DOLLARS $$

APPLICATION FOR MEMBERSHIP, INTERNATIONAL SUPERPET USERS' GROUP
(A non-profit organization of SuperPET Users)

Name:_____ Disk Drive: _____ Printer:_____

Address:_____
Street, PO Box City or Town State/Province/Country Postal ID#

For Canada and the U.S.: Enclose Annual Dues of $15:00 (U.S.) by check payable to ISPUG. DUES ELSEWHERE: $25.00 U.S. Mail to: Paul V. Skipski, Secretary, ISPUG, P.O. Box 411, Hatteras, N.C. 27943, U.S.A.
SCHOOLS: Send check with Purchase Orders. We do not bill or send vouchers.

ALPHABETICAL INDEX TO STARTER-PAK DISK

This is the alphasorted directory of: "tutorial tu" 2C, created by sortdir:bu, on disk. A commented index of 6 pages is on disk as index:e. It explains the purpose and use of each program or file. This listing is on disk as: alphindex.

"alive:fd"	4	SEQ	"alphindex"	9	SEQ
"apl_dumps:e"	11	SEQ	"aplfiles:aws"	31	PRG
"banner:bd"	7	SEQ	"cancode:e"	2	SEQ
"change_device:bu	1	SEQ	"curdemo:fd"	5	SEQ
"datesetsee_pd"	2	SEQ	"dir:men"	2	PRG
"disktoprinter:fd	4	SEQ	"disktoscreen_pd"	2	SEQ
"dos:aws"	48	PRG	"ds:bu"	3	SEQ
"dsktoprintr_pd"	4	SEQ	"dump:bu"	4	SEQ
"factorials_pd"	2	SEQ	"get_keyboard:bd"	5	SEQ
"guessdemo:bd"	6	SEQ	"hexdump:fd"	4	SEQ
(Index file, scratched by sort)			"index:e"	49	SEQ
"iodemo:fd"	3	SEQ	"list:f"	2	SEQ
"loader:au"	19	PRG	"marquee:bd"	6	SEQ
"massive:bd"	11	SEQ	"mbasic_patch2"	4	SEQ
"patch:e"	13	SEQ	"peekhex:bu"	3	SEQ
"pokefont_pd"	3	SEQ	"pokescreen:bd"	2	SEQ
"primes_pd"	5	SEQ	"printfilep:bu"	12	SEQ
"printscreen:bd"	6	SEQ	"printscreen:bu"	6	SEQ
"readfile1:fd"	3	SEQ	"readfile:fd"	3	SEQ
"reset:men"	1	PRG	"retrieve:men"	1	PRG
"reverse_pd"	3	SEQ	"rhyme:fd"	6	SEQ
"screentodisk:bd"	5	SEQ	"screentodisk:fd"	3	SEQ
"screentodisk_pd"	3	SEQ	"scrntoprntr:fd"	3	SEQ
"scrntoprntr_pd"	1	SEQ	"sdumps:aws"	20	PRG
"seeclock_pd"	2	SEQ	"setcurs_pd"	5	SEQ
"setdate:fd"	3	SEQ	"setprinter_pd"	3	SEQ
"sortdir:bu"	21	SEQ	"systimeset_pd"	6	SEQ
"timesetsee_pd"	2	SEQ	"timtest:fd"	3	SEQ
"try_codes:bd"	6	SEQ	"tutorial.1:e"	94	SEQ
"tutorial:e"	102	SEQ	"u:7ef0"	2	PRG
"u:men"	2	PRG	"ulf:7e60"	2	PRG
"ulf:men"	2	PRG	"when:men"	2	PRG
"whentab:bu"	8	SEQ			

Read the commented index:e in the microEDITOR, loaded alone. You may locate the comments on purpose and function of each program there. In addition, each program is commented in the particular language used.