MGT
MILES GORDON
TECHNOLOGY

Pick-POKE-It

For use with MGT's PLUS D Interface

ZX Spectrum 48K
ZX Spectrum +
ZX Spectrum + 128
ZX Spectrum 128K + 2

Pick-POKE-It

from Miles Gordon Technology

For use with MGT's PLUS D Interface

ZX Spectrum 48K
ZX Spectrum +
ZX Spectrum + 128
ZX Spectrum 128K + 2

USER MANUAL

CONTENTS

INTRODUCTION

Pick-POKE-It adds additional facilities to the PLUS D interface which allow the user to explore and modify existing programs as well as providing a very powerful set of tools for machine code and BASIC programmers, available at the touch of a button without affecting the program being worked on.

Normally, the PLUS D offers five snapshot facilities - two methods of printing screen-shots, as well as a screen, a 48K and a 128K save to disc. When the PLUS D DOS is modified with Pick-POKE-It, all these features are retained, but there are six extra facilities. In the first place, the contents of the Spectrum RAM can be displayed, edited, searched and disassembled, making Pick-POKE-It an essential tool both for beginners who wish to modify their games software and for more experienced users writing their own programs. More advanced users will be able to use Pick-POKE-It to inspect and edit the Z80 registers, and to switch from one bank of memory to another, particularly useful with the 128K Spectrum.

Of special note are the following Pick-POKE-It features:

- The software has been professionally designed for ease of operation and clarity of display; it is fully menu-driven and allows 15 lines of code to be displayed on the screen. While experienced programmers will be able to make greatest use of the software, Pick-POKE-It will be an excellent introduction to machine-code for the novice, and is a useful tool even for those who have little interest in programming but who simply wish to breathe new life (or infinite lives!) into their games.

- By making use of the PLUS D's snapshot button, PICK-POKE-It allows you to inspect, edit or print out the contents of RAM while a program is still running. This means that you are able to inspect any changes that you may have made immediately by returning to the original program, and when you are satisfied with your changes, re-commence or save the modified program as required. And of course, the PLUS D will continue to work at its normal speed - loading and saving a full 48K program in less than four seconds.

If machine code is new to you, there will be enough guidance in this manual to get you started and to use Pick-POKE-It properly. But if you'd like to learn more, you'll need to do some extra reading too. There's a suggested reading-list at the end of the manual. You'll also find that the various Spectrum magazines print articles to introduce their readers to machine-code and to suggest how commercial software can be modified with POKEs.

MODIFYING YOUR SYSTEM DISCS

All you need to create your first Pick-POKE-It system disc is the program supplied on cassette and a PLUS D disc containing a working SYSTEM file. This system disc must have at least 48K of free space.

It's a good idea, however, to start by producing a disc version of the program on cassette, and to use this as a master for producing future Pick-POKE-It system discs. To do this, you'll require the disc to have at least 80K of free space.

Start with the PLUS D system booted in the normal way. Then with the disc you wish to use in the drive (Drive 1 if you have more than one), load the program from tape using the command:

```
LOAD ""
```

Then simply follow the instructions on-screen.

If you now catalogue your disc you should see three additional files as below:

```
+DPP1
+DPP2
+DPP3
```

Each of these files is about 4K long. You will also notice that your original system file has been replaced by one called:

```
+SYS PP1
```

If you answered "Yes" when asked if you wished to save the program on tape onto your disc, then two additional files will be present:

```
PICKPOKEIT
POKEITCODE
```

Later you can load the PICKPOKEIT program to create further Pick-POKE-It system discs without having to use the cassette - but keep the tape safe somewhere as a back-up.

From now on, if you boot the system using the Pick-POKE-It system disc, the extra facilities will always be available to you, although for normal operation the PLUS D will behave as before. But when you press the snapshot button and then the Spectrum key "P", there will be a few seconds of disc activity, and then the Pick-POKE-It menu will be displayed. Next we describe how to use the various menu options.

WHAT IF YOU CAN'T LOAD PICK-POKE-IT

A few PLUS D users are still using Version 1 of the ROM, which was used in PLUS D's sold in December 1987-January 1988. Pick-POKE-It won't work with the version 1 ROM. If you find that Pick-POKE-It doesn't work, check the serial number on the bottom of your PLUS D. If it's a 4-figure number commencing with 1, then you have a PLUS D with the Version 1 ROM. Call MGT on 0792-791100 and we'll arrange a replacement ROM for you.

GENERAL FEATURES OF Pick-POKE-It

THE NORMAL PLUS D SNAPSHOT FACILITIES

These operate exactly as before. Press the snapshot button and then select any one of keys 1 to 5 on the Spectrum.

If you press the snapshot button, then key "P", the Pick-POKE-It menu will appear. The menu confirms that if you now press keys 1 to 5, the same original PLUS D snapshot facilities for printing and saving to disc are retained.

USE WITH TWO DISC DRIVES

If, after moving into snapshot mode, you hold down the CAPS SHIFT key while selecting keys 3, 4 or 5, the drive selected will change from one to the other.

If you have 2 disc drives and wish to use the Pick-POKE-It facilities on a program contained on a disc which does not have the Pick-POKE-It system files, then you can force Pick-POKE-It to take its files from the alternate disc by holding down CAPS SHIFT while you press "P".

But if you wish to save a file to disc from the Pick-POKE-It Menu, then the destination disc (i.e. the disc on which you are saving the file) must have the Pick-POKE-It system on it.

CHANGING DISCS

Once you are in the Pick-POKE-It mode, you should not change the Pick-POKE-It system disc unless you are replacing it with a disc also containing the Pick-POKE-It files, or unless you exit from Pick-POKE-It first. If you do, and then try to select another of the options from the Pick-POKE-It menu, your Spectrum will crash (though all is not lost! - see below).

EXITING FROM Pick-POKE-It

When the Pick-POKE-It menu is displayed, press key "X" on the Spectrum to return to the normal PLUS D operating system. This means that you can modify a program in memory using Pick-POKE-It, then exit, and snapshot save the modified program on a disc not containing the Pick-POKE-It system files.

When you are using any of the special Pick-POKE-It facilities, then key "X" will return you to the main Pick-POKE-It menu.

Pick-POKE-It TEMPORARY FILES

When you press the snapshot button and key "P", you'll hear the disc drive spinning into operation for a few seconds. What's happening here is that the software is setting up temporary files on disc so that there can be no loss of RAM data while the Pick-POKE-It facilities are being used.

These files are given the names DraySoftSC and DraySoftPG. Don't give these file-names to any files that you create - though we can't imagine why you should want to!

When you exit from Pick-POKE-It by pressing key "X", these temporary files will be automatically erased.

If the computer crashes while you are using Pick-POKE-It, the temporary file will be retained on the disc so that you can restore it. You'll need to boot up the Pick-POKE-It system again, and then follow the instructions in the section on Bank Switching later in this manual.

In these circumstances, the temporary files will be erased automatically for you later - you don't need to worry about it.

Because Pick-POKE-It writes to your disc, you must make sure that your disc is not write-protected before you start work with Pick-POKE-It.

CRASHES

If your computer crashes while you are using Pick-POKE-It, then you'll need to turn off the electrical power, then turn it on again, and reboot from scratch. The temporary files described above will still remain on disc.

AN INTRODUCTION TO MACHINE-CODE

We're not going to attempt to teach machine-code here, but if you're a newcomer there are certain principles that you need to understand to make effective use of Pick-POKE-It.

You'll probably have done at least some elementary programming in BASIC. BASIC is a high-level computer language - high-level because it uses terms which are familiar to all of us. For example, it's easy to understand what LOAD and SAVE mean because these have meanings close to those in everyday use.

But while we understand the words, they mean nothing to a machine like a computer. What the computer CAN understand is whether or not an electrical signal is present, whether it is on or off. The conventional way to present this information is to use the Binary system: instead of using "on" or "off", we use 1 or 0.

The Spectrum is an 8-bit computer, meaning that eight of these signals can be sent in parallel to the central processing unit (CPU) at any one time. Thus the signals being sent at any one time could be represented as 01101101 or 0101101.

Did you have to look at these numbers twice to see the difference between them? That's the problem that programmers would face if they had to work in Binary code. While the machine understands the signals, it would be very difficult to read pages of numbers written in Binary: our programs would be filled with errors.

That's why most people start programming in BASIC, which is - debatably - the easiest of the programming languages. When you type LOAD, a series of interpretations are made until the CPU receives an electrical signal that it can understand. But here's another problem. The interpretations take time - fractions of a second perhaps, but enough to slow down a program appreciably. If we had to write our computer games in BASIC, they would be very slow-moving.

And so, programmers use an intermediate language which is closer to Binary, but at the same time relatively easy to read - or at least it is if you've had some practice. This language is called machine-code.

Machine-code can be represented in either Decimal numbers - those that are in everyday use - or Hex, which stands for Hexadecimal. The Hexadecimal system is a way of arranging numbers in units of 16 - 0-1-2 3-4-5-6-7-8-9-A-B-C-D-E-F. Conventionally Hex numbers are preceded by the sign #. Thus, #3B equals 59 in Decimal [(3 x 16) plus 11]; #EA equals 234 [(14 x 16) plus 10]. For practice, try converting the following Hex numbers to Decimal - don't worry, you won't have to do this while using Pick-POKE-It and the answers are on the last page of this manual:

#5B;   #F2;   #20 (take care - it's not 20);   #29C;
#BF0;   #112A;   #6E1F;   #AB10;   #FFFF
(Go on - try it before you go on to the next section!)

This will probably seem a strange - and unnecessarily complicated - method of counting. But for the computer engineer or serious programmer, it's a better method than Decimal for the following reasons:

- The computer works in a Binary manner (remember the electrical signals that are either on or off?). There's a direct correlation between Hex and Binary. In simple terms, the Decimal number 16 is represented by #10 or by 10000 in Binary; 256 - an important number in computing - is represented by #100 or by 100000000 in Binary.

- Conversely there's very little relation between Decimal numbers and Binary. 10 in Decimal is 1010 in Binary; 100 is 1100100; and 1000 is 1111101000.

- Hex numbers are relatively easy to remember and read, particularly where large values are concerned. For example, the top location in the Spectrum's memory is 65535 in Decimal but #FFFF in Hex.

When using Pick-POKE-It, you'll see information about addresses in the Spectrum's memory. Consider the address as being the place where information is stored, rather in the same way that you might put messages into somebody's pigeon-hole at work or at school. There are 65535 (= #FFFF) addresses/pigeon-holes in the Spectrum's memory, and the machine regularly checks each of them to see whether any messages have been left there; if it finds a message it acts upon it. Some of the locations are filled with permanent information for standard control of the computer: this is from address 0 to 16384 (= #4000). Then addresses up to 23754 (= #5CCA) are taken up by messages affecting, for example, the screen. The remainder of the memory is reserved for messages which determine how a program runs, and is the area that we are primarily concerned with when we wish to customise software.

Many of the addresses may remain empty; but where messages are left Pick-POKE-It will show these as machine-code instructions. The maximum value of an instruction will be 255 in Decimal or #FF.

If you're a beginner, you don't need to understand what the machine-code instruction means. Indeed, usually sets of instructions act together, so a single instruction may have a different significance at one address when compared with another. However, as long as you understand what we mean by Hex and Decimal, by address and instruction, then you will be able to follow a very simple set of rules to customise your software - to give yourself infinite lives or energy in a game, for example, or perhaps to change the text on the screen. If you want to go on to learn more about machine-code later, all well and good.

ENTERING INFORMATION IN Pick-POKE-It MODE

ENTERING ADDRESSES

When using one of the Pick-POKE-It facilities, you are asked to enter an address, and the value can be entered either as a Decimal number or in Hex. For Hexadecimal values the entry must start with a "#" character; you don't need to mark Decimal numbers - simply enter the number. Pick-POKE-It checks all entries for validity. If you've made an error (for example, if the number is too low or too high) an appropriate error report is given, and you are asked to enter the value again. If you wish to change your entry, use the Spectrum's normal character delete key (or keys in the case of the Spectrum 48K).

DEFAULT ADDRESS VALUES

Since the Pick-POKE-It functions are intended only to be used with RAM (the area of memory that changes when programs are running), all addresses below 16384 (#4000) are invalid and cannot be entered. Additionally, since the area of memory from address 16384 (#4000) to 23754 (#5CCA) contains the screen image, channels and Spectrum system variables, this area changes as the Pick-POKE-It programs run. To overcome this, the original RAM data is saved as a temporary file on the disc and a special facility is provided for looking at this area. This facility is described later in the "Bank Switch" section.

If, when you are asked for a start address in any of the Pick-POKE-It programs, you simply press ENTER, then the default value of 23755 (#5CCB) will be automatically entered, because this is where you are likely to want to start. However, if you have chosen to start with a higher address, then the next time you are asked for a start address, ENTER will take you to the place where you started before - provided you have not used another of the Pick-POKE-It programs in the interim.

NUMBER ENTRY

When you are editing the RAM contents or searching the RAM, you will be asked to enter a value as well as specifying an address. If you are entering a number - which will be a machine-code instruction - it can be either a Decimal number or a Hex value, in which case you must mark it with a "#". All entries are checked for validity, and if necessary an appropriate error message will be displayed and a new value requested. You can use the Spectrum's delete key(s) if you wish to change what you have entered.

ENTERING ASCII VALUES

When you are using Pick-POKE-It's Search and Edit programs you are allowed to look for or replace a string of characters. For example, you might want to find the part of the program where the instructions to print the High Score table are printed on screen, so you'd be searching for the character string "High Score". By convention, the letters, numbers and sign conventions which appear on the computer keyboard are known as ASCII values.

In this example, when Pick-POKE-It asks you for the value you're searching for, you can type $High Score, and the appropriate part of the program can be located.

Whenever you wish to enter an ASCII value, simply type $ and follow it directly with the required text. You don't need to type in inverted commas. The maximum length of an ASCII string which can be entered at one time is 29 characters, not counting the $ at the start. No $ is required at the end, and the string can contain "$" characters if required.

USING A PRINTER

The Disassembler and Display Memory programs can send their output to the printer port, using the set-up from your original PLUS D system file. When you ask the information to be sent to the printer, the program will ask for an end address. If ENTER alone is pressed, then the default value of 65535 (#FFFF) - the top of the Spectrum's RAM will be used.

Printing can be terminated at any time by pressing key "X" - although printing will obviously continue until your printer's buffer is empty.

Now that you've learnt the basic principles of Pick-POKE-It, we'll start using the facilities. Don't be afraid to experiment! If anything goes wrong, your original program will still be on disc. Remember that you must keep a disc with the Pick-POKE-It programs in drive 1 when you are using the Pick-POKE-It programs. And if you do get into trouble you can always escape from Pick-POKE-It and return to the original PLUS D facilities by pressing key "X".

Pick-POKE-It FACILITIES

THE DISASSEMBLER

Pick-POKE-It's Disassembler allows you to display or print out machine code in the Spectrum's RAM, together with the Mnemonic. (For non-programmers, the Mnemomic is a sort of short-hand that programmers use to note the instruction which is actually being given to the CPU. If you want to know more, you'll need to do some reading - any of the books on our recommended reading list will be suitable.)

Typically, you'll display or print the disassembly when you want to see a machine-code listing of the entire program held in RAM.

To use the Disassembler, press the Snapshot Button, then key "P" to enter the Pick-POKE-It routines. Then press key 6 and you'll be asked to specify the start address. If you want to see the entire program, press ENTER and the listing will automatically start from address 23755 (#5CCB). Similarly, if you press ENTER when asked to specify the end address, Pick-POKE-It will default to the top of RAM - location 65535 (#FFFF). Then you'll be asked whether you wish to print out the listing. If you do, press Y; if not simply press ENTER.

For each instruction to the Z80 processor, you'll see the address, the bytes forming the instruction, and the Mnemonic displayed, in a form like this (although the actual values will be different when you try it):

```
5CCB     00        NOP
5CCC     013700    LD     BC,0037
5CCF     E7        RST    20
5CD0     310E00    LD     SP, 000E
5CD3     00        NOP
```

All the values displayed here are in Hexadecimal, although in this display the # character is not used. (As always, when you are entering the start address and the end address, you can do so in either Decimal or Hex, but if you use Hex, you must precede the number with #.)

The first column shows the address in memory; the second column tells you the bytes forming the instruction at this address; while the third and fourth columns are a machine-code Mnemonic, showing the programmer clearly what operation is being performed.

Newcomers to machine-code may be interested to know what the Mnemonics mean in this example:

```
        NOP               - No Operation at this address.
        LD  BC, 0037      - Load the register BC with the value
                              0037
```

(NB: "register" refers to a location in the computer's CPU - it is a location in ROM; whereas "address" refers to a location in memory - a location in RAM.)

```
        RST 20            - Restart at Restart Code number 20
        LD  SP, 00DE      - Load the Stack Pointer with the
                              value 00DE
```

From this, you should be able to see that there's nothing particularly mysterious about Mnemonics - although if you want to understand them fully, you'll need to do some further reading.

Note also that not all the possible addresses are listed. For example the listing jumps from #5CCC to #5CCF (missing out addresses #5CCD and #5CCE). In this case, this is because the instruction at address #5CCC is executed over three address lines.

Experienced programmers will know what they want to do with the Disassembly. Newcomers will probably find the other Pick-POKE-It facilities more useful at first. However, you might like to experiment by loading a game and printing out the disassembly. Or try typing in a short Basic program (for example, the Squares program in the PLUS D manual) to see what a Basic program looks like in machine code.

DISPLAY MEMORY

This facility displays the Spectrum RAM data either to the screen or the printer. A typical display would be arranged like this:

```
23936    #5D80    245    #F5    PRINT
23937    #5D81     34    #22    "
23938    #5D82     83    #53    S
23939    #5D83     84    #54    T
23940    #5D84     79    #4F    O
23941    #5D85     80    #50    P
23942    #5D86     32    #20
23943    #5D87     84    #54    T
23944    #5D88     65    #41    A
23945    #5D89     80    #50    P
23946    #5D8A     69    #45    E
23947    #5D8B     34    #22    "
```

Column 1 displays the address number in Decimal. Column 2 displays the address in Hex. Columns 3 and 4 display the RAM data at that address in Decimal and in Hex respectively. And column 5 displays the data as an ASCII value where appropriate.

Let's look at column 5 more closely. At address 23936, the code 245 Decimal can be (and is, in this instance) represented in BASIC by the keyword PRINT. Similarly the code 34 Decimal at address 23937 has an ASCII value equivalent to "; and the code 83 Decimal at address 23938 has an ASCII value equivalent to S. If you read down the right-hand column, you'll see that the instructions at this range of addresses is responsible for printing the message "STOP TAPE" on the screen. Notice the consistencies: the ASCII value T always has a code representation of 84 Decimal (#54), while O has a code representation of 79 Decimal (#4F). Can you estimate from this information the Decimal and Hex code representations for Q, U and G?

At address 23942 there is no ASCII value marked in our example. In fact, many addresses will have no information in column 5, or there may be a yellow or white square on the screen. If you see a blank yellow square, then the ASCII value is a space. If you see a blank white or an unmarked square in column 5, then the address contains a machine-code instruction or a status value that has nothing to do with an ASCII value.

There's one small problem. In the example above, the code 245 Decimal (#F5) is said to be the equivalent of the BASIC command PRINT. From the ASCII values which follow, we can see that this is an interpretation which is likely to be correct - the BASIC statement PRINT "STOP TAPE" makes sense. However, in a different context, 245 Decimal may have another meaning altogether. Instructions in the immediate vicinity may have changed the meaning slightly or completely. To draw a parallel which may make this easier to understand, the same thing happens in language. For example the word FED has a basic meaning which has something to do with food and eating. However, in the context, I'M FED UP, it departs from its basic meaning completely, although it still looks to be the same word.

For this reason, some of the BASIC command words or ASCII values in column 5 may be an inaccurate interpretation of the code representations in columns 3 or 4. How do you know which are correct and which are not? Experience - and a little common-sense! Patches of column 5 will look like normal lines of BASIC - and they probably are. Others won't - and they probably aren't. However, it will make enough sense for you to be able to search for particular areas of the program - and we'll see how to do this in the section on Searching RAM.

To display the memory, select option 7 from the Pick-POKE-It menu screen. Then follow the instructions on pages 9 and 10 to specify addresses and other values.

EDIT RAM CONTENTS

This is the Pick-POKE-It facility which actually allows you to change your programs. Experienced programmers may want to change entire screens or perhaps sprites, having located them using the Disassembler or the Display Memory facilities. But for beginners, there's plenty of information in the standard Spectrum magazines to allow you to enter infinite lives or energy or ammunition easily. You'll be able to look at all your old games in new ways.

When starting out, try finding a so-called "Multiface Poke" in one of the magazines for a game in your possession. Here are a few examples, culled from recent magazines:

PAPERBOY	
49263,0	- Infinite Papers
50577,190	- Infinite Lives
50495,201	- Immunity from injury
STARGLIDER:	
54647,201	- More fuel
54690,201	- More shields
BMX KIDZ	
52108,0	- Energy
LAST NINJA II	
29966,n	- n = Lives
40777,0	- Lives
PLATOON	
31138,0	- Grenades
31268,0: 31269,0	- Hits
31270,0	- Morale

Each of these codes is an address (in Decimal), then after the comma a machine-code instruction (in Decimal) to enter at this address.

If you haven't already done so, boot up using your Pick-POKE-It disc, and then load the game in the normal way. If the game isn't on the same disc as Pick-POKE-It, it'll make things easier if you save it there. Bring up the normal Pick-POKE-It screen, and then select option 4 or 5 as appropriate to resave the game. Go back to the Pick-POKE-It menu screen and select option 8. You'll be asked to specify the start address.

In the case of PAPERBOY, you'd type in 49263. This would bring up this display:

```
        ENTER NEW VALUE ($ for STRINGS)
       RESTART (R) EXIT (X) CONT (ENTER)

     Start Address >: 49263

        49263     #C06F    61    #3D   =

     Value  >:
```

Columns 1 and 2 display the start address, and columns 3 and 4 the current code representation at this address respectively, while the right-hand column shows the ASCII value.

To have infinite papers in the game, the value you need to type in is 0. Simply type 0, then press ENTER. You'll see the amended line displayed:

```
        49263     #C06F     0    #00
```

and the next address - 49264 - will appear, ready for further amendment. But you don't want to amend this: the next address to change is 50577. So press R, then enter to restart. When asked for the address, type 50577, and do the same as before. Continue until you've made all your changes. Finally type X, then ENTER to return to the main menu, and X again to return to the game. All your changes will be retained in RAM, and you should now have immunity, and infinite lives and papers as you play. If you wish to save this version of the game, take a snapshot save in the normal way.

If you'd wanted to make a change at address 49265 but not at 49264 above, you could have pressed ENTER at 49264 to keep the same line and pass on directly to the next address.

You can also use the Edit RAM Contents facility to change an ASCII string - for example, you might want to change the words HIGH SCORE on screen to BEST SCORE, but we'll deal with that in the next section.

SEARCH RAM

This facility allows you to search the contents of RAM for a number, or a sequence of numbers, or for a string of ASCII values. Once you have found what you were looking for, you can then choose to disassemble, to display memory, or to edit memory from that point - or to continue the search for the next occurrence. Because this combines the features of all the Pick-POKE-It facilities mentioned so far, it is likely to be the one you will use most often.

Suppose you wanted to change the text HIGH SCORE displayed on screen during a game to the text BEST SCORE. To try this, we took the game BMX RACERS, loaded it in the normal way, brought up the Pick-POKE-It main menu, and then selected option 9. We started searching from the default value of 23755 Decimal, but to make life a little more difficult we started searching not for HIGH SCORE but simply for SCORE. To do this entered the value $SCORE. (Remember that when you type in an ASCII value, you need to precede it with $.) The message -

```
        STRING FOUND AT   : - 41674   (#A2CA)
```

appeared, together with a sub-menu:

```
        1)  DISPLAY MEMORY
        2)  DISASSEMBLE
        3)  EDIT MEMORY
        4)  CONTINUE SEARCH
        R)  RESTART SEARCH
        X)  QUIT
```

We chose option 1 - DISPLAY MEMORY - which showed us that the string SCORE did indeed begin at address 41674. But to see whether this was the HIGH SCORE display that we were looking for, we needed to go back a few address lines further. So we entered R to restart, and specified a start address of 41665 Decimal. This showed that the ASCII string at this location was only SCORE, not HIGH SCORE. We entered X, which returned us to the sub-menu; option 4 then continued the search for the next occurrence of the string SCORE. This wasn't the one we were looking for either, and we had to continue several more times until eventually we found HIGH SCORE beginning at address 44878.

We then selected option 3 from the sub-menu - EDIT MEMORY. In the previous section, we described how to change code with the Edit Memory facility, but this time we needed to change a string of ASCII values in a sequential sequence of addresses. First we typed in the required start address - 44878. The address line appeared on the screen with the first letter of the current ASCII string - H - in the right-hand column. When asked to enter a value, we entered $BEST SCORE. Immediately, the nine edited address lines (one address line for the space between the words) appeared with all amendments made.

Having made the change we entered X to return to the sub menu, then X again to return to the main Pick-POKE-It menu, then X to return to the original program.

When we started to edit the program earlier, we were actually looking at a screen which had HIGH SCORE displayed on it. When we returned to it, the words HIGH SCORE were unchanged: this was because Pick-POKE-It has no effect on the current screen in the computer's memory. However, as we continued to play the game, we saw that HIGH SCORE had indeed been replaced by BEST SCORE.

We then took an ordinary snapshot save of the program so that on all future occasions we would play it with the same change made.

If all this is new to you, why not experiment by starting with the High Score table in one of your favourite games and altering the names of the highest scorers to your own name? There are three things you have to remember. First, if the high score table uses special graphic characters instead of the Spectrum's own character set you won't be able to search for an ASCII string as we did in our example; so, to practise, use a game which does use the normal Spectrum characters. Secondly, if you're replacing a sequence of letters, the number of letters you use must be the same as the original. Finally, when searching for an ASCII string, remember that you must use exactly the same letters as the original program: if HIGH Score appears on screen, then you must search for HIGH Score; if it's written HiGh ScOrE, that's what you must type. This is because the ASCII values for capital letters are different from those for lower-case letters.

ADDITIONAL FEATURES FOR PROGRAMMERS

The final two facilities on the main Pick-POKE-It menu - Z80 Registers and Bank Switching - will be best understood by more experienced users.

Z80 REGISTERS

When the snapshot button on the PLUS D is pressed, the contents of all the Z80 registers are stored within the interface so that the program can be correctly restarted. Selection of the Z80 REGISTERS option from the Pick-POKE-It menu allows you to display and edit the stored copy of the registers.

Random changing of the register contents will almost certainly cause the Spectrum to crash, or the program in RAM to give strange results. Changing the SP (Stack Pointer) or PC (program counter) registers will almost certainly have this effect.

When you exit from the Z80 REGISTERS facility you will be asked if you wish to keep the changes you have made. If you have made a mistake or are in doubt, then answer NO and the original register values will be preserved.

This facility is a very powerful aid to machine code debugging, since it allows the cause of computer crashes to be explored; quite often you will be able to recover the program.

In addition to the normal Decimal and Hexadecimal modes of number entry, this facility allows two other special modes:

BINARY NUMBER ENTRY

Numbers can be entered in Binary form by preceding the entry with B. No leading zeros are required.

EDITING HALF OF A REGISTER PAIR

Certain registers in the Z80 can be used in pairs for 16 bit values (numbers from 0 to 65535) or as two individual registers for 8 bit values (numbers from 0 to 255). To allow editing of one half of a register pair without affecting the other, the symbols < and > can be used. The symbol < is used to change the left-hand register, and the symbol > the right-hand register. The symbol must be the first character in the entry and must precede the B when Binary numbers are entered.

BANK SWITCHING

This facility allows you to perform two separate tasks:

(i) BANK SELECT (128K Spectrums only)

The 128K Spectrum RAM consists of 8 separate 16K pages, of which only 3 are selected at any one time. The RAM pages are numbered 0 to 7 and normally pages 2 and 5 are permanently selected, with one of the remaining pages 0,1,3,4,6 or 7 being selected as required.

RAM page 5 is normally located from address 16384 (#4000) to 32767 (7FFF), page 2 is located from address 32768 (#8000) to 49151 (#BFFF), and one of the remaining six pages occupies the address range 49152 (#C000) to 65535 (#FFFF).

To enable all of the RAM to be accessed by the Pick-POKE-It routines, there needs to be a method of selecting the 6 pages of RAM at address #C000. The BANK SWITCHING program provides this facility.

When the BANK SWITCHING routine is first entered, the screen will show the RAM bank which was selected when the PLUS D snapshot button was pressed. To select another page, simply keep pressing the space key until the required page is flashing, then press key X to exit. The required page will then be switched in at address 49152 (#C000). You can then return to the BANK SWITCH program at any time to select another page.

Note that the page 5 selection is not one of the 128K RAM banks, but has a special purpose which is described in the following section.

There is no need to restore the original page selection before exiting from Pick-POKE-It as the exit routine tidies everything for you.

(ii) SCREEN AND VARIABLE AREA (48K and 128K Spectrums)

The memory area from address 16384 (#4000) to 23754 (#5CCA) contains the screen image, channels and Spectrum system variables. This area changes as the Pick-POKE-It programs run. To overcome this, the original RAM data is saved as a temporary file to the disc and a special facility is provided for looking at this area. By using the BANK SWITCH facility and selecting RAM page 5, the temporary file is loaded in from disc and occupies the address range 49152 (#C000) to 56522 (#DCCA), where it can be examined and edited without interference. The original content of these locations is saved as another temporary file on the disc.

There is no need to restore the original page selection before exiting from Pick-POKE-It as the exit routine tidies everything for you. If however, you wish to restore the original contents at address 49152 (#C000) then return to the BANK SWITCH program and simply select the page you require. In the case of the 48K Spectrum, selecting any page other than 5 will restore the original RAM contents.

CALCULATING THE ADDRESS OFFSET

If, for example, you wished to examine or edit the contents of the Spectrum system variable ATTR P (the paper and ink colours) at address 23693 (#5CBD), it would be necessary to calculate its new location after page 5 has been selected. Your calculation would be:

23693 - 16384 + 49152 = 56461

Obviously, this would be tedious, so a special address entry mode is included to enable the program to calculate the address for you. The special mode is invoked by placing a * symbol before the address - i.e:

*23693 or *#5CBD

The * symbol is considered an invalid character when RAM page 5 is not selected.

RECOMMENDED READING LIST

These are books which will help you to learn more about machine code programming. Some of them are Spectrum specific; others give a more general introduction to the Z80 processor.

Understanding Your Spectrum - Ian Logan - (Melbourne House)
Mastering Machine Code on your ZX Spectrum - Toni Baker - (Interface)
The Complete Spectrum ROM Disassembly - Ian Logan & Frank O'Hara - (Melbourne House)
The Working Spectrum - David Lawrence - (Sunshine)
40 Best Machine Code Routines for the ZX Spectrum - John Hardman & Andrew Hewson - (Hewson)
Spectrum +2 Machine Language for the Absolute Beginner - Joe Pritchard (Melbourne House)
Z80 and 8080 Assembly Language Programming - Kathe Spracklen - (Hayden)
Programming the Z80 - Rodney Zaks - (Sybex)

If your interest develops and you wish to write your own machine-code routines and programs, we recommend Hi-Soft's Devpak as a comprehensive Assembler/Debugger with a clear manual. Also commended by reviewers recently is Lerm Software's Z80 Toolkit - particularly good for writing shorter routines for those who have an understanding of the first principles of machine-code programming.

Contact MGT if you need further details.

Answers to questions on page 8

91	(5 x 16) + 11
242	(15 x 16) + 2
32	
668	[2 x (16 X 16)] + (9 x 16) + 12
3056	
4394	
28191	
43792	
65535 (This number is the highest address in the Spectrum's memory)	