ROOM FOR MANOEUVRE

Simple utility programs, like the variable search program we wrote on pages 664 and 700, can be written entirely in BASIC, using only information about how individual lines of BASIC are stored. For more complicated utilities, however, we need greater detail and therefore must resort to machine code.

In order to operate our variable search program, we merged it with the program to be searched. With this method, the only information from the operating system that we had to supply was the address where the BASIC program starts; the end of the program being searched was found by testing for the lowest line number in the utility program.

The utility that we are creating is a variable replace program. This is a very useful program to have on file. If you had used a variable name throughout a program only to find it was illegal, imagine how much time such a program would save. Similarly, you might have written a program you wanted someone else to use in which the variable names were not easy to decipher. Here we explain the necessary theory for the machine code and in the next instalment we will publish the listings.

SPACE IN MEMORY

In this exercise, we need to put the utility program in a separate section of memory from the program it is working on. We must also find a different method of locating the end of the BASIC program, and a means of accommodating two BASIC programs in the computer at the same time.

The three computers that we are looking at — the BBC Micro, Commodore 64 and Sinclair Spectrum — use a set of pointers to tell the operating system and the BASIC interpreter where to locate BASIC programs and variables, etc. (see page 56). Unfortunately, the details are different in the three machines.

On the BBC Micro, there are four important pointers: PAGE and TOP, which hold the beginning and end address of the BASIC program; LOMEM, which holds the start address of the BASIC variables; and HIMEM, which holds the end address of the BASIC area. These four pointers are stored as built-in BASIC variables, and we can read or alter their values by simple BASIC statements. If we have a BASIC program in memory and we wish to add another, we change PAGE to a value higher than TOP — using the command OLD to reset TOP and LOMEM — and can then add the new program without affecting the original program. We change from one program to another by giving new values to both PAGE and HIMEM and using the command OLD.

Once we have the utility program running, the values of the pointers refer to the utility program; to enable the utility to find the start and end of the program it is to work on, we need to copy the original values into an area of memory that will not be altered when we change programs. Another method of finding the end of a program is to use the end marker that the BASIC interpreter puts in. This is simply a byte holding a value of 128 or more, immediately following the carriage return character at the end of the last line of the program. This byte, and the one following, will be interpreted as the HI and LO bytes of the next line number. Since the HI byte of this number is 128 or more, this will give a line number of 32768 (256×128) or more. As the highest valid line number is 32767, we can be sure that we have found the end of program marker and not just another line number.

The Commodore 64 uses seven pointers, stored in zero page memory, to indicate various parts of the BASIC program area. TXTTAB, at addresses 43 and 44, points to the start of the BASIC program; VARTAB, ARYTAB, STREND, FRETOP and FRESPC, at addresses 45 to 54, point to various sections of the variable table; and MEMSIZ, at addresses 55 and 56, points to the end of the BASIC area. It is possible to change these pointers in order to create a separate area in which to run a BASIC program by use of the POKE command. However, a short machine code program is recommended as it is more direct, and reduces the chances of crashing the computer with a typing mistake.

In the Commodore 64, the end of a BASIC program is indicated by two bytes containing zeros immediately following the zero byte marking the end of the last line of the program. Following the chain of pointers at the beginning of each line of the program until we find a pointer of zero will indicate the end of the program.

FOR THE SPECTRUM

Creating this utility is rather more complicated on the Spectrum. Instead of a separate area for the BASIC program, there is a single, continuous block of memory that includes not only the BASIC program and variables, but also all the workspace areas used by the operating system and the BASIC interpreter. With this layout of the memory it is difficult, if not impossible, to have two BASIC programs in the main working area, so we will make a copy of our program above RAMTOP and work on it there. This still leaves the problem of recovering the program and fitting it into the main