THE HUMAN FACTOR

An important aspect of program design is the 'man-machine interface' — the part of the program that deals with the transfer of information from user to program and vice versa. Here, we investigate the factors to be considered when designing this interface.

Computer programming was for many years a mysterious topic that was understood only by professionals who were prepared to devote much time and effort to the subject. Before the advent of the microcomputer with its typewriter-style keyboard, programs were often entered one byte at a time via switches on the computer's front panel, or by punching holes in tapes on a teletype console.

Today's user is, by contrast, a pampered creature. Manufacturers no longer expect the computer owner to struggle with machine code, and the phrase 'user-friendly' was coined to indicate that micros may be used and programmed by anyone, regardless of experience. In 1982, the Alvey Committee, in a report entitled *A Programme for Advanced Information Technology,* identified the man-machine interface (MMI) as one of the four main areas of research and development, together with software engineering, very large scale integrated circuit (VLSI) design and knowledge-based systems.

In any application, the interaction between computer and user, where data or instructions are passed between the two, is of paramount importance. This 'dialogue' is conducted through the computer's input/output (I/O) devices, with the keyboard serving as the main source of input and the display screen providing the output. Joysticks, paddles, mice, touch screens and other devices may also be used for input, while the computer can utilise a printer, sound (or speech) generator or even a robot to express the output.

In addition to any constraints imposed by the I/O devices used, the dialogue between user and machine is influenced by software. For example, the computer's operating system (OS) controls many details of the screen and keyboard operation. The rate at which keys repeat when held down, and the delay between repetitions, is set by the operating system, which also buffers keystrokes to allow the computer to store characters that have been entered faster than they can be displayed. This is very important as it affects the speed at which the user may enter information into the computer. The buffer size is critical and should be known by the user — the CP/M operating system, for example, buffers a single keystroke; many home machines buffer 10 strokes or more.

But keystroke buffers may cause problems. An experienced user who is working with a menu-driven system may know in advance that the menu choices he requires are 2 from the main menu, 5 from the next menu, then 3, 4, 6, etc. Because he is familiar with the system, he types his choices at great speed. With a 10-character buffer, the user will end up where he wanted to go because the keystrokes will all be 'remembered' in the correct sequence. With a one-character buffer, the time taken to display the second menu may be longer than the time taken to type the sequence. Thus, instead of selecting choice number 5 from this menu, then 3 and so on, choice number 6 alone is made (because this is the only character held in the buffer) and the system stops there.

But a large buffer can also lead to problems. A menu program that takes a long time to react to a keypress (this may occur if the choice leads to a file being read) may cause the user to think that nothing is happening. The natural response is to try the last choice entered, then press an assortment of keys until there is a response. This

may lead to the program attempting to process the spurious characters held in the buffer; the results may be surprising!

'Garbage collection', which involves clearing the computer's memory registers to free working space is another source of problems. This can make a program appear to 'hang' for long periods, during which the user may again try to take corrective action. Garbage collection is likely to cause problems in large programs that do a lot of string handling. Some versions of BASIC allow the