## *DEVICES : IMPLICATIONS*

What constraints are imposed on the operating system design by our need to use devices, and by the ways in which we would like to use them ? To answer this question, we have to look in two directions – both towards the programmes which require the devices, and towards the devices themselves. It is the operating system's job to link the two in such a way that people's needs are adequately met.

We identified two different sorts of device – or, perhaps better, two ways of using a device – by inspecting this link. In some cases, the device was quite well-matched to the programmes' needs, and the main requirement was for fairly simple buffering software. In others, and most notably with terminals and discs, the needs of the programmes could be quite different from the facilities provided by the device. In this event, additional software was necessary, typically to map several active streams of communications onto a single hardware interface in such a way as to satisfy the requirements of all parties to the transactions.

We have already done some work at the programme end during our earlier discussion on files, so our main concern here is with the devices. It is useful to think about using the devices on two time scales – in the short term, for actual input and output, and in the longer term, as components of the system which must be managed. The management view will be explored more thoroughly in the next section.

### USING THE DEVICES.

**Device table :** There must be some way for the system to find all the information it needs about devices; a device table is the obvious answer. The table might be organised as an actual array of device descriptors ( in which case the number of permissible devices could be hard to change ), or as a linked list of descriptors. To manage the device table, we also need operating system means – procedures or instructions – for putting devices into the tables, and taking them out. We might also need means for determining the state of devices and for controlling them – so, for example, it might be useful to mark a device as reserved so that after its current activity is completed it will not be allocated to any other programme. If we are using a table of this sort, there must be a way to set it up when the computer is started; this is part of the configuring operation.

**Device drivers :** Software to handle the physical communication with each device at the hardware interaction level is needed. Such software must provide an adequate set of operations. Interrupt handling, or something of the sort, will be needed if we want to continue processing in parallel with the device operation.

**File attributes :** If we want to check that device equivalence is used safely, we need information about the files. This is more than just extending the disc file directories : we need to know ( in the device table ) what attributes a file must or mustn't have to be acceptable to a device, and also ( defined by the programme that makes the file ) the attributes of each logical file in the process's file table.

**Terminals :** Terminals are special because they need to communicate both with the operating system and with processes. We might even require that a terminal should be able to switch between several different processes. The need for some communication channel with the operating system, bypassing the current programme, can be very hard to satisfy if the system architecture isn't designed to permit it – for example, if the terminal is completely driven by the programme which is using it. If we are using a graphical windowed interface in the common "Macintosh" mode, the need for switching implies the existence of a screen manager, which keeps track of the position of a cursor or similar marker and directs communications accordingly. With a more traditional system, we might define some special key sequence as an escape from the current programme to the system, which we can then instruct to switch the terminal to a different process if that is required.

**Discs ( and any other direct-access storage devices ) :** Two complicating features are characteristics of devices of this sort : as with terminals, many processes may use the device concurrently, and – more obvious with discs than with terminals – there are several different ways in which discs are used. The interface software must therefore provide a variety of routes between software and disc, and also coordinate all transactions.