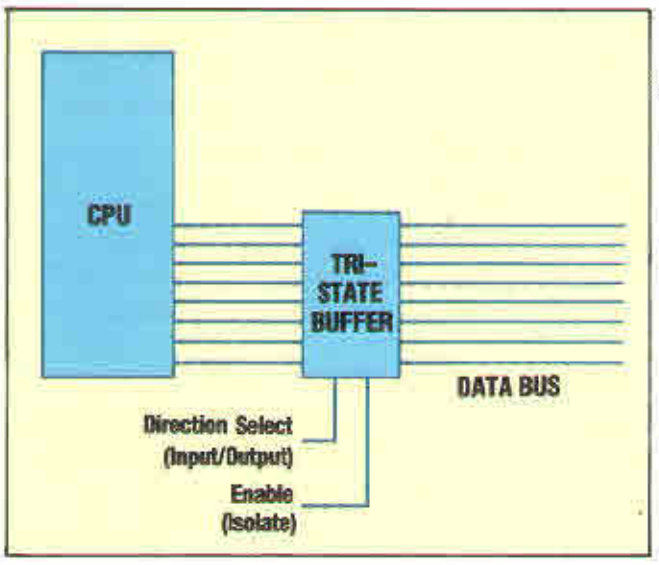
MEMORY JOGGING

The central processing unit or CPU is the nerve centre of your home computer. In this instalment of the Computer Science course, we take a look at the logic of data transfer between the CPU and memory. In this discussion, we introduce the address and data buses, tri-state devices and the memory address register.

Each memory location in a home computer is normally made up of eight bits. Data can be transferred, eight bits at a time, along a series of eight parallel lines to the CPU where the data can then be used according to a program instruction. Data can also be sent in the opposite direction in order to be stored in a memory location. The machine code instruction LDA $1234 causes the number in location $1234 to be sent along the data bus to the CPU. STA $1234 causes a number to be sent from the CPU, again down the data bus, and stored in location $1234.

The data bus must therefore allow transfers in both directions. At certain times it is also important to isolate the CPU from the data bus. Thus each line of the data bus can be in one of three states (INPUT, OUTPUT or ISOLATE). In order to achieve the necessary switching between these states, each data bus line has a small electronic circuit known as a *tri-state device.*



Eight such tri-state devices are combined into a single integrated circuit. The diagram above shows how this integrated circuit links the data bus with the CPU. The diagram also shows the 'enable' and 'direction select' lines, which put the eight tri-states into the required operative state. Such circuits can also be used for connecting other devices, such as input/output peripherals, to the data bus.

Whenever we wish to call up the contents of a particular location we refer to the location required by its address. Each location in ROM and RAM has its own unique number that refers to it. Let us now look at how, at the hardware level, any location in memory can be accessed so that a data transfer can be accomplished.

Most microcomputers have a second highway between the CPU and memory called the *address bus.* Normally, the address bus has 16 lines rather than eight. This means that up to 65,536 separate addresses can be specified ($2^{16} = 65{,}536$). That is, up to 64 Kbytes of memory can be accessed using a 16-bit address bus. The total memory area can be thought of as being broken up into modules, each containing 256 locations. The lower eight bits of the address can then be used to find the particular location within a given module. The module itself may be selected by using some or all of the remaining eight address bits.

If we consider the simple example of a microcomputer with a total memory size of two Kbytes, we can see how the selection of any particular location takes place. As each module of memory contains 256 locations, our two-Kbyte computer will require eight modules. For our simple computer we will assume that the memory is equally divided between ROM and RAM.

The address of the required location is held in a special 16-bit register in the CPU, called the memory address register or MAR. As the lower eight bits of the address select a particular location within any module, the lower eight lines of the address bus can be connected to each of the memory modules. In order to select a particular module we now require only a further three bits ($2^3 = 8$). This three-bit code must be decoded into eight output lines, one for each module.

The diagram shows how memory modules link via these data and address buses to the CPU. Each memory module has a single line to it from the three(bit)-to-eight(line) decoder. Three of the higher address bits are used to determine which module is to be selected. If more RAM modules were to be added, then more of the upper eight bits would be required to select any single module.

MACHINE CODE INSTRUCTIONS

Having seen how a particular location in memory may be selected and data transferred, let us look at how the CPU carries out a machine code instruction. Any machine code program is normally stored in consecutive storage locations. One instruction may take two or three bytes of storage. An instruction such as ADD $13FF means 'add the contents of the location with hexadecimal