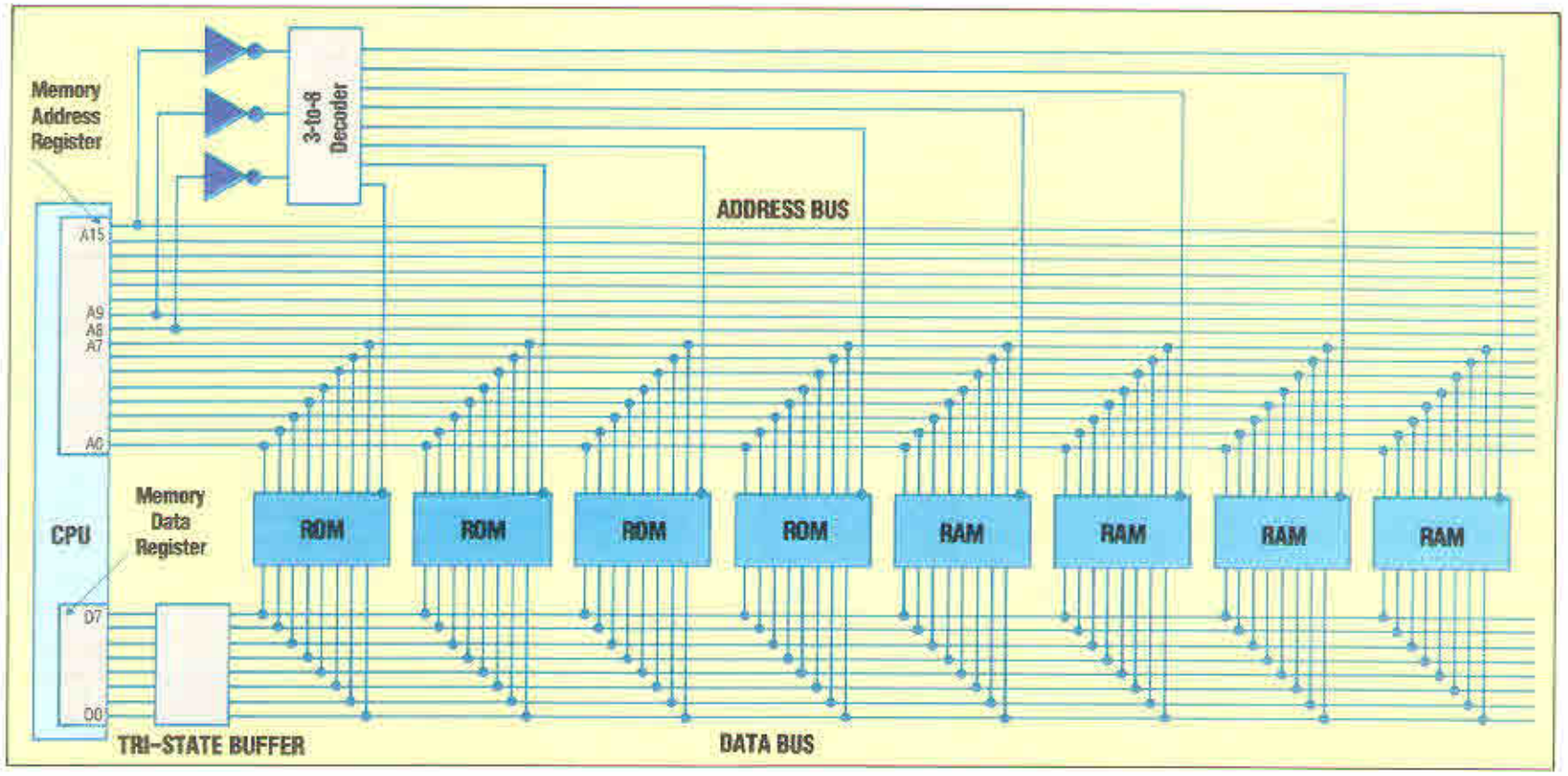


address $13FF to the accumulator'. This instruction would require three bytes: one to hold the binary code for the instruction ADD and two to hold the 16-bit address, $13FF. Let us say that it is stored in locations $1000, $1001 and $1002.

Before the instruction can be processed it must be fetched from memory. This requires three separate accesses to bring the three bytes along the data bus to the CPU. On completion of the fetch cycle, the complete instruction is in a special register within the CPU. All that remains is to decode the instruction and obey it. The instruction in our example requires a further memory access to get the contents of location $13FF so that it can be added to the accumulator.

All computer manufacturers publish the characteristics of their processors in the form of timing diagrams. These show the order of events for a number of different computer operations. We can draw up a timing diagram for the fetch and execute cycles of a machine code instruction. The timing of operations is controlled from the clock pulse (see page 246) and our graph shows that in this imaginary system the address bus is enabled by the sync pulse leading edge, while the data bus is enabled by the sync trailing edge. The sync pulse itself is triggered by the trailing edge of the first clock pulse in any operation phase, or machine cycle. The cycles are of different durations because the processor needs longer to decode the op-code byte of an instruction than to handle the operand bytes: the op-code must be decoded immediately because it specifies the number of operand bytes.

**Fetch And Execute**
A machine code instruction consisting of an op-code byte followed by two operand bytes is handled in an instruction cycle comprising fetch and execute phases. During the fetch phase the address bus accesses the memory locations holding the instruction, and the data bus carries the instruction bytes to the CPU. Here, the data and address buses are still busy during the execute phase because the instruction being executed causes a memory access