software design decisions, based purely on how the JVM behaves, are also outlined where they had a significant performance benefit.

The PC Architecture

The modern PC is a very complicated beast. Its hardware has been optimized and iterated over many times to produce a highly effective and generalized computing platform. However, it also carries with it legacy components and functionality designed to maintain its backward compatibility. In some ways this is a blessing. The basic architecture of an IA-32 machine has not changed since its advent in 1985 with the 386. In fact, in terms of system architecture, the 386 itself was not much of a departure from its x86 predecessors.

Although Figure 9-1 is in some ways grossly simplified, with some text changes and some duplication of boxes, this could easily pass as an architectural diagram for JPC itself.

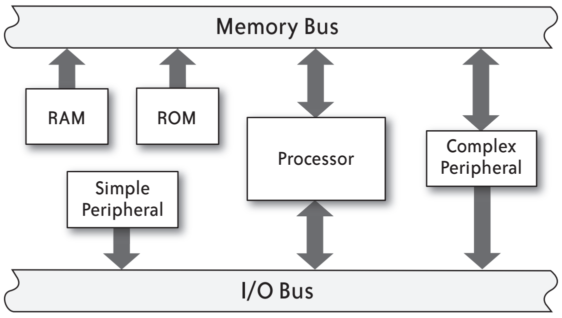

FIGURE 9-1. Basic architecture of a modern PC

Designing the bulk of JPC was a relatively simple matter of systems analysis, and mapping from the original system to JPC for the bulk of the emulation is almost a 1:1 correspondence between the hardware specs and the Java class. For example, a serial port in JPC for example is represented by a single class, `SerialPort`, that implements `HardwareComponent` and `IOPortCapable`. This simplistic approach gives rise to a design that is easy to understand and navigate, and on the whole, objects within the architecture are loosely coupled to each other. This gives JPC the benefit of being very flexible, so just as in a real machine, virtual devices can be “plugged in” to the PCI bus, and components can be interchanged to build virtual machines of wide-ranging specifications.

The only reason to depart from this path is when clarity of design and modularity are in direct competition with performance. This occurs in two key places in JPC: once at the concentration of computation (the processor) and once at the concentration of bandwidth (the memory system). These hot spots have two effects on the project: