12 Architecture and Instruction Sets

The usual meaning of architecture stems from the design of buildings and is concerned with such things as the way that the purpose of the building is related to the use of space, the size and shape of the structure, the materials used, the aesthetic qualities of the building (does it look good?) and so on. An architect, or team of architects, will have designed the building first of all 'in their mind's eye', recording details on paper and using models, long before a stone is set or the earth is disturbed. Similarly, the design of a new computer involves a large team of design engineers creating a computer in imagination long before its creation in silicon as a working system.

As with the building architects, the computer architects are concerned with the purposes of their creation, and such things as operating speeds, instructions sets, memory size, operation codes and operating system and the way the hardware (electronic circuitry, switches, wires, sockets, and so on) is connected together to fulfil these purposes.

*Computer architecture is the way that a computer has been designed, at both hardware and software levels, to carry out its tasks.*

The *instruction set* of a computer is an important part of the architectural system. The instruction set is the total number of separate, unique instructions designed to carry out all the tasks for which the computer has been designed. Depending on the size of the computer, the instruction set will be stored entirely in the computer's ROM or there may be a small program in the computer's ROM that allows the instruction set to be loaded into memory from the backing store (such a program is called a *bootstrap*: have you heard about pulling yourself up by the bootstraps?). In either case, the boundaries between hardware and software become somewhat blurred down at this lowest level of the way the hardware *interfaces* or connects with the rest of the computer software through the instruction set. For this reason, the instruction set (and certain other special programs) can be considered as *firmware*. Figure 12.1 shows these different levels in a computer system.
Many different ways have been used to classify the instructions in a set into smaller groups of related instructions. One way of classifying the instruction is given below.

<table>
<thead>
<tr>
<th>Groups in the instruction set</th>
<th>Example instruction</th>
<th>Binary code for the mnemonic</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movement of data</td>
<td>LDA #163</td>
<td>10000110</td>
<td>A register called A must be loaded with Decimal 163</td>
</tr>
<tr>
<td>Arithmetic</td>
<td>INCB</td>
<td>01011100</td>
<td>The contents of a register called B must be incremented by 1</td>
</tr>
<tr>
<td>Logic</td>
<td>ORA #13</td>
<td>10001010</td>
<td>Bits 0–3 of the 8-bit register A must be set to decimal 13. Bits 4–7 are unaffected</td>
</tr>
<tr>
<td>Test</td>
<td>CMPB #60</td>
<td>11000001</td>
<td>The contents of register B are Compared to the decimal number 60</td>
</tr>
<tr>
<td>Branch</td>
<td>BNE L3</td>
<td>00100110</td>
<td>If a previous comparison shows inequality then branch to address L3: Branch if Not Equal</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>RTS</td>
<td>011111001</td>
<td>Return To Subroutine: similar to END or RETURN in BASIC, used to finish a program</td>
</tr>
</tbody>
</table>

As with any classification system, some individual instructions may not belong only to one group, there may be overlap; also, different ideas on how the groups are to be formed can result in a different way of ordering the instructions. The way we classify things varies according to our ideas about what it is that we are classifying. Consequently, another way of grouping an instruction set might be

1. Movement of data
2. Manipulation of data
3. Transfer of control
4. Input–output operations.

The number of instructions in a set varies from computer to computer and can range from less than 20 to more than 200. The total number of instructions in a set,