```
    // 将 1 相加 100 次
    for (i = 1; i <= 100; i++) {
        sum += 1;
    }

    // 总和结果除以 10
    sum /= 10;

    // 显示结果
    printf("%d\n", sum);
}
```

除此之外，BCD（Binary Coded Decimal）[①] 也是一种使用二进制表示十进制的方法。简单来讲，BCD 就是用 4 位来表示 0～9 的 1 位数字的处理方法，这里不再做详细说明。在涉及财务计算等不允许出现误差的情况下，一定要将小数转换成整数或者采用 BCD 方法，以确保最终得到准确的数值。

3.8　二进制数和十六进制数

最后再补充说明一下二进制数和十六进制数的关系。在以位为单位表示数据时，使用二进制数很方便，但如果位数太多，看起来就比较麻烦。因此，在实际程序中，也经常会用十六进制数来代替二进制数。在 C 语言程序中，只需在数值的开头加上 0x（0 和 x）就可以表示十六进制数。

二进制数的 4 位，正好相当于十六进制数的 1 位。例如，32 位二进制数 00111101110011001100110011001101 用十六进制数来表示的话，

① 计算机中用到的数据表现形式中，有一种叫作 BCD（Binary Coded Decimal，二进制化十进制数）的方法。这种方法常被用于老式的大型计算机中。编程语言中，COBOL 也会使用 BCD。BCD 分为 Zone 十进制数形式和 Pack 十进制数形式两种。

就是 3DCCCCCD 这个 8 位数。由此可见，通过使用十六进制数，二进制数的位数能够缩短至原来的 1/4。位数变少之后，看起来也就更清晰了（图 3-9）。

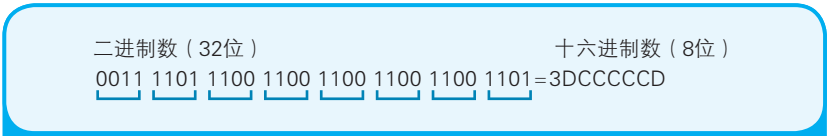

图 3-9　通过使用十六进制数使位数变短

用十六进制数来表示二进制小数时，小数点后的二进制数的 4 位也同样相当于十六进制数的 1 位。不够 4 位时用 0 填补二进制数的低位即可。例如，1011.011 的低位补 0 后为 1011.0110，这时就可以表示为十六进制数 B.6（图 3-10）。十六进制数的小数点后第 1 位的位权是 16^{-1}，即 1/16 = 0.0625，这个大家应该能理解吧。

图 3-10　小数点后的二进制数的 4 位也相当于十六进制数的 1 位

通过学习第 2 章和本章的内容，想必大家已经掌握了计算机通过二进制数来处理数据（数值）的机制。接下来的一章，我们将继续向大家介绍用于数据存储的内存。如果大家在编程时能够时刻考虑到内存问题，那么就一定能彻底理解被公认为复杂难懂的 C 语言的数组和指针了。

第4章 熟练使用有棱有角的内存

热身问答

阅读正文前，让我们先回答下面的问题来热热身吧。

问题

1. 有十个地址信号引脚的内存 IC（集成电路）可以指定的地址范围是多少?
2. 高级编程语言中的数据类型表示的是什么?
3. 在 32 位内存地址的环境中，指针变量的长度是多少位?
4. 与物理内存有着相同构造的数组的数据类型长度是多少?
5. 用 LIFO 方式进行数据读写的数据结构称为什么?
6. 根据数据的大小链表分叉成两个方向的数据结构称为什么?

怎么样？是不是发现有一些问题无法简单地解释清楚呢？下面是笔者的答案和解析，供大家参考。

答案

1. 用二进制数来表示的话是 0000000000 ~ 1111111111（用十进制数来表示的话是 0 ~ 1023）
2. 占据内存区域的大小和存储在该内存区域的数据类型
3. 32 位
4. 1 字节
5. 栈
6. 二叉查找树（binary search tree）

解析

1. 地址信号引脚是十个时表示 2^{10}= 1024 个地址。
2. 例如，C 语言数据类型中的 short 类型，它表示的就是占据 2 字节的内存区域，并且存储整数。
3. 指针指的是用于存储内存地址的变量。
4. 物理内存是以字节为单位进行数据存储的。
5. 栈是一种后入先出（LIFO = Last In First Out）式的数据结构。
6. 二叉查找树指的是从节点分成两个叉的树状数据结构。

本章重点

计算机是进行数据处理的设备，而程序表示的就是处理顺序和数据结构。由于处理对象数据是存储在内存和磁盘上的，因此程序必须能自由地使用内存和磁盘。因此，大家有必要对内存和磁盘的构造有一个物理上的（硬件的）和逻辑上的（软件的）认识。

本章的主题是内存（磁盘部分会在第 5 章中讲解）。其实，从物理上来看，内存的构造非常简单。只要在程序上花一些心思，就可以将内存变换成各种各样的数据结构来使用。譬如，物理上有棱有角的内存，在程序上是可以按照逻辑很流畅地使用的。而且这并不特别，它是很多程序中都会用到的一般方法。

4.1 内存的物理机制很简单

为了能够对内存有一个整体把握，首先让我们来看一下内存的物理机制。内存实际上是一种名为内存 IC 的电子元件。虽然内存 IC 包括 DRAM、SRAM、ROM[①] 等多种形式，但从外部来看，其基本机制都是一样的。内存 IC 中有电源、地址信号、数据信号、控制信号等用于输入输出的大量引脚（IC 的引脚），通过为其指定地址（address），来进行数据的读写。

图 4-1 是内存 IC（在这里假设它为 RAM[②]）的引脚配置示例。虽然这是一个虚拟的内存 IC，但它的引脚和实际的内存 IC 是一样的。VCC

① ROM（Read Only Memory）是一种只能用来读取的内存。

② RAM（Random Access Memory）是可被读取和写入的内存，分为需要经常刷新（refresh）以保存数据的 DRAM（Dynamic RAM），以及不需要刷新电路即能保存数据的 SRAM（Static RAM）。

和 GND 是电源，A0～A9 是地址信号的引脚，D0～D7 是数据信号的引脚，RD 和 WR 是控制信号的引脚。将电源连接到 VCC 和 GND 后，就可以给其他引脚传递比如 0 或者 1 这样的信号。大多数情况下，+5V 的直流电压表示 1，0V 表示 0。

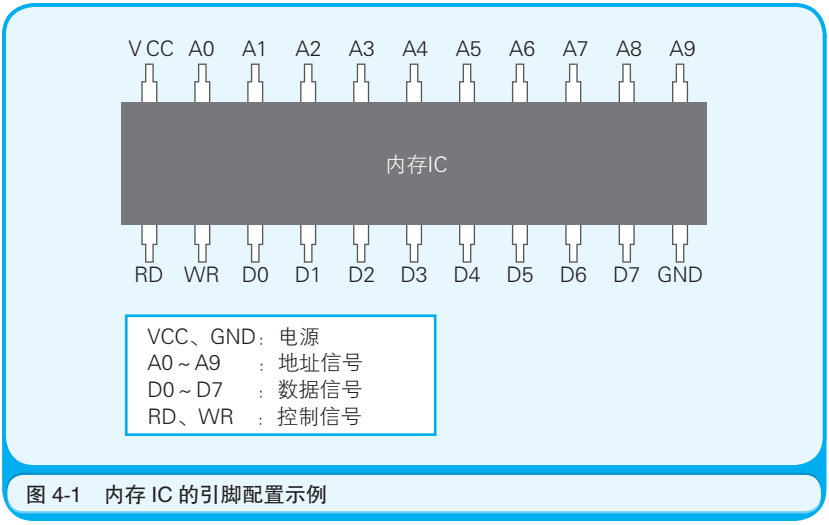

图 4-1　内存 IC 的引脚配置示例

那么，这个内存 IC 中能存储多少数据呢？数据信号引脚有 D0～D7 共八个，表示一次可以输入输出 8 位（= 1 字节）的数据。此外，地址信号引脚有 A0～A9 共十个，表示可以指定 0000000000～1111111111 共 1024 个地址。而地址用来表示数据的存储场所，因此我们可以得出这个内存 IC 中可以存储 1024 个 1 字节的数据。因为 1024 = 1K[①]，所以该内存 IC 的容量就是 1KB。

现在大家使用的计算机至少有 512M 的内存。这就相当于 512000

① 在计算机领域，大写字母 K 表示的并不是 1000，而是 2 的 10 次幂的结果 1024。1000 通常用小写 k 来表示。

个（512MB ÷ 1KB = 512K）1KB 的内存 IC。当然，一台计算机中不太可能放入如此多的内存 IC。通常情况下，计算机使用的内存 IC 中会有更多的地址信号引脚，这样就能在一个内存 IC 中存储数十兆字节的数据。因此，只用数个内存 IC，就可以达到 512MB 的容量。

下面让我们继续来看刚才所说的 1KB 的内存 IC。首先，我们假设要往该内存 IC 中写入 1 字节的数据。为了实现该目的，可以给 VCC 接入 + 5 V，给 GND 接入 0V 的电源，并使用 A0～A9 的地址信号来指定数据的存储场所，然后再把数据的值输入给 D0～D7 的数据信号，并把 WR（write = 写入的简写）信号设定成 1。执行完这些操作，就可以在内存 IC 内部写入数据（图 4-2 (a)）了。

读出数据时，只需通过 A0～A9 的地址信号指定数据的存储场所，然后再将 RD（read = 读出的简写）信号设成 1 即可。执行完这些操作，指定地址中存储的数据就会被输出到 D0～D7 的数据信号引脚（图 4-2(b)）中。另外，像 WR 和 RD 这样可以让 IC 运行的信号称为**控制信号**。其中，当 WR 和 RD 同时为 0 时，写入和读出的操作都无法进行。

由此可见，内存 IC 的物理机制实质上是很简单的。总体来讲，内存 IC 内部有大量可以存储 8 位数据的地方，通过地址指定这些场所，之后即可进行数据的读写。

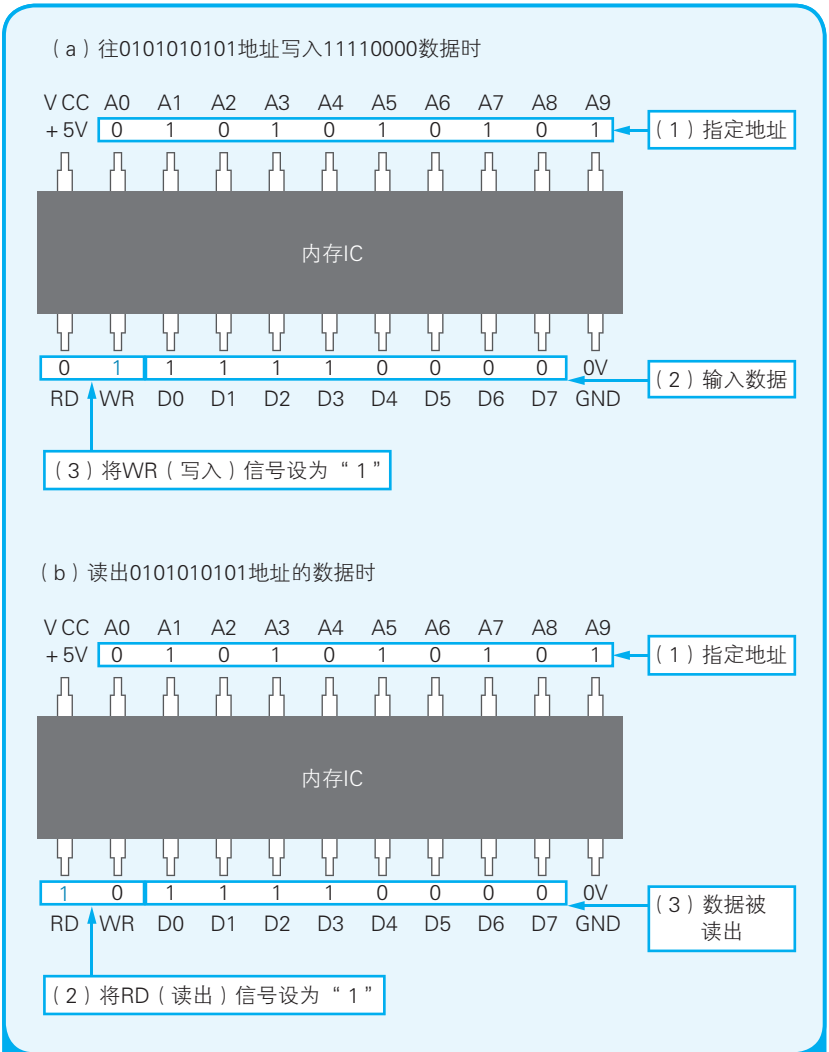

图 4-2　向内存 IC 中写入和读出数据的方法

4.2 内存的逻辑模型是楼房

在介绍程序时，大部分参考书都会用类似于楼房的图形来表示内存。在这个楼房中，1 层可以存储 1 个字节的数据，楼层号表示的就是地址。对于程序员来说，这种形象的解说有助于了解内存。

虽然内存的实体是内存 IC，不过从程序员的角度来看，也可以把它假想成每层都存储着数据的楼房，并不需要过多地关注内存 IC 的电源和控制信号等。因此，之后的讲解中我们也同样会使用楼房图（或者与楼房相似的图）。内存为 1KB 时，表示的是如图 4-3 所示的有 1024 层的楼房（这里地址的值是从上往下逐渐变大，不过也有与此相反的情况）。

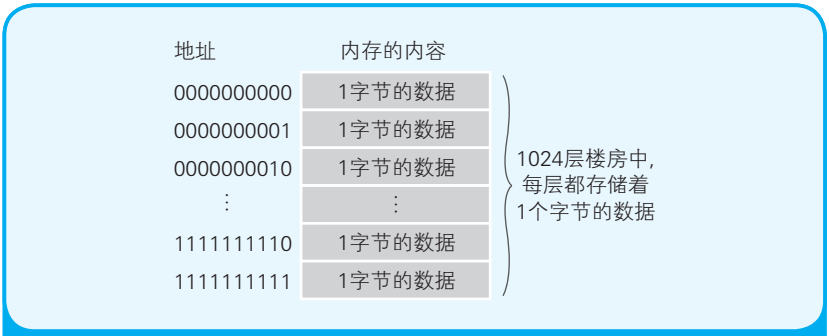

图 4-3 1KB 内存的模型

不过，程序员眼里的内存模型中，还包含着物理内存中不存在的概念，那就是数据类型。编程语言中的**数据类型**表示存储的是何种类型的数据。从内存来看，就是占用的内存大小（占有的楼层数）的意思。即使是物理上以 1 个字节为单位来逐一读写数据的内存，在程序中，通过指定其类型（变量的数据类型等），也能实现以特定字节数为