描述了在C语言中插入汇编代码的方法。对于一些应用程序，程序员必须用汇编代码来访问机器的低级特性。一种方法是用汇编代码编写整个函数，在链接阶段把它们和C函数组合起来。另一种方法是利用GCC的支持，直接在C程序中嵌入汇编代码。

旁注　ATT与Intel汇编代码格式

我们的表述是ATT(根据“AT&T”命名的，AT&T是运营贝尔实验室多年的公司)格式的汇编代码，这是GCC、OBJDUMP和其他一些我们使用的工具的默认格式。其他一些编程工具，包括Microsoft的工具，以及来自Intel的文档，其汇编代码都是Intel格式的。这两种格式在许多方面有所不同。例如，使用下述命令行，GCC可以产生multstore函数的Intel格式的代码：

```
linux> gcc -Og -S -masm=intel mstore.c
```

这个命令得到下列汇编代码：

```
multstore:
  push    rbx
  mov     rbx, rdx
  call    mult2
  mov     QWORD PTR [rbx], rax
  pop     rbx
  ret
```

我们看到Intel和ATT格式在如下方面有所不同：

- Intel代码省略了指示大小的后缀。我们看到指令push和mov，而不是pushq和movq。
- Intel代码省略了寄存器名字前面的‘%’符号，用的是rbx，而不是%rbx。
- Intel代码用不同的方式来描述内存中的位置，例如是‘QWORD PTR [rbx]’而不是‘(%rbx)’。
- 在带有多个操作数的指令情况下，列出操作数的顺序相反。当在两种格式之间进行转换的时候，这一点非常令人困惑。

虽然在我们的表述中不使用Intel格式，但是在来自Intel和Microsoft的文档中，你会遇到它。

网络旁注 ASM:EASM　把C程序和汇编代码结合起来

虽然C编译器在把程序中表达的计算转换到机器代码方面表现出色，但是仍然有一些机器特性是C程序访问不到的。例如，每次x86-64处理器执行算术或逻辑运算时，如果得到的运算结果的低8位中有偶数个1，那么就会把一个名为PF的1位条件码(condition code)标志设置为1，否则就设置为0。这里的PF表示“parity flag(奇偶标志)”。在C语言中计算这个信息需要至少7次移位、掩码和异或运算(参见习题2.65)。即使作为每次算术或逻辑运算的一部分，硬件都完成了这项计算，而C程序却无法知道PF条件码标志的值。在程序中插入几条汇编代码指令就能很容易地完成这项任务。

在C程序中插入汇编代码有两种方法。第一种是，我们可以编写完整的函数，放进一个独立的汇编代码文件中，让汇编器和链接器把它和用C语言书写的代码合并起来。第二种方法是，我们可以使用GCC的内联汇编(inline assembly)特性，用asm伪指令可以在C程序中包含简短的汇编代码。这种方法的好处是减少了与机器相关的代码量。