过量的 register 变量声明 没有坏处，因为 register 只是 尽可能 ，不是 绝对

可以使用 register 最好 用下，因为编译器 并不是 智能的 很多时候 不会自动 判断

register：这个关键字请求编译器尽可能的将变量存在CPU内部寄存器中，而不是通过内存寻址访问，以提高效率。注意是尽可能，不是绝对。你想想，一个CPU 的寄存器也就那么几个或几十个，你要是定义了很多很多register 变量，它累死也可能不能全部把这些变量放入寄存器吧，轮也可能轮不到你。

二、举例

register修饰符暗示编译程序相应的变量将被频繁地使用，如果可能的话，应将其保存在CPU的寄存器中，以加快其存储速度。例如下面的内存块拷贝代码，

```
#ifdef NOSTRUCTASSIGN
memcpy (d, s, l)
    {
                register char *d;
            register char *s;
            register int i;
            while (i--)
                        *d++ = *s++;
    }
#endif
```

使用register 修饰符的注意点

但是使用register修饰符有几点限制。

首先，register变量必须是能被CPU所接受的类型。这通常意味着register变量必须是一个单个的值，并且长度应该小于或者等于整型的长度。不过，有些机器的寄存器也能存放浮点数。

其次，因为register变量可能不存放在内存中，所以不能用“&”来获取register变量的地址。

由于寄存器的数量有限，而且某些寄存器只能接受特定类型的数据（如指针和浮点数），因此真正起作用的register修饰符的数目和类型都依赖于运行程序的机器，而任何多余的register修饰符都将被编译程序所忽略。

在某些情况下，把变量保存在寄存器中反而会降低程序的运行速度。因为被占用的寄存器不能再用于其它目的；或者变量被使用的次数不够多，不足以装入和存储变量所带来的额外开销。

早期的C编译程序不会把变量保存在寄存器中，除非你命令它这样做，这时register修饰符是C语言的一种很有价值的补充。然而，随着编译程序设计技术的进步，在决定那些变量应该被存到寄存器中时，现在的C编译环境能比程序员做出更好的决定。实际上，许多编译程序都会忽略register修饰符，因为尽管它完全合法，但它仅仅是暗示而不是命令。

什么情况用寄存器变量：

当对一个变量频繁被读写时，需要反复访问内存，从而花费大量的存取时间。为此，C语言提供了一种变量，即寄存器变量。这种变量存放在CPU的寄存器中，使用时，不需要访问内存，而直接从寄存器中读写，从而提高效率。寄存器变量的说明符是register。对于循环次数较多的**循环控制变量**及循环体内反复使用的变量均可定义为寄存器变量，而**循环计数**是应用寄存器变量的最好候选者。

什么变量可以声明为寄存器变量：

. 只有局部自动变量和形参才可以定义为寄存器变量。因为寄存器变量属于动态存储方式，凡需要采用静态存储方式的量都不能定义为寄存器变量，包括：模块间全局变量、模块内全局变量、局部static变量。