原文地址：[I/O操作函数inb, outb, inw, outw](http://www.live83.cn/?p=103)  作者：[alen](http://www.live83.cn/?author=2)

; Copyright (C) 2011 Alen D. Archuleta (zeafoo@gmail.com) ; 设备和芯片的I/O端口操作实现，其实没有复杂的东西在里边 ; I/O端口操作主要是看一堆文档，把整个X86架构的PC机所有I/O端口记住， ; 并记住它们每一个数据寄存器、命令寄存器等操作访问标准（也可以称之协议） ; 记住之后，整个过程中就是按标准使用I/O指令： ; in, out（只能与DX,AX,AL寄存器结合使用） ; 下面的实现是提供给C使用，因为不太喜欢GNU的inline asm，语法太 ; 晦涩，所以直接使用汇编实现。 ; inb 从I/O端口读取一个字节(BYTE, HALF-WORD) ; outb 向I/O端口写入一个字节（BYTE, HALF-WORD） ; inw 从I/O端口读取一个字（WORD，即两个字节） ; outw 向I/O端口写入一个字（WORD，即两个字节） ; byte inb(word port); ; word inw(word port); ; void outb(word port, byte value); ; void outw(word port, word value); ; 编译： ; nasm -f elf -o io.o io.asm ; 与内核一起链接使用 global inb, outb, inw, outw [section .text] inb: xor eax, eax ; 在C语言中，都是以EAX寄存器作为返回值 push dx ; 这个过程要对DX修改，所以先保存一下 ; 返回地址占加个字节，所以偏移4开始读取，第一个参数 ; 对于参数传递过程不作详细的记录，另写日志详细介绍 mov dx, [esp + 4] in al, dx pop dx ; 恢复DX寄存器 ret outb: push dx mov dx, [esp + 4] mov al, [esp + 6] out dx, al pop dx ret inw: xor eax, eax push dx mov dx, [esp + 4] in ax, dx pop dx ret outw: push dx mov dx, [esp + 4] mov ax, [esp + 6] out dx, ax pop dx ret