 本质上来说, 这篇文章是把我最感兴趣的两样编程东西: Linux 操作系统和汇编语言程序设计结合在一起. 这两个都不(或者说应该不)需要介绍. 像 Win32 的汇编,Linux 的汇编运行在 32 位的保护模式下...但它又有一个截然不同的优势就是它允许你调用 C 的标准库函数和 Linux 的共享库函数。 我开始给 Linux 下的汇编语言编程来个简要介绍. 为了更好读一点, 你可能要跳过这个基本的小节。 编译和链接 Linux 下两个最主要的汇编器是 Nasm(free, Netwide Assembler)和 GAS(free, Gnu Assembler),后一个和 GCC 结合在一起. 在这篇文章里我将集中在 Nasm 上, 把 GAS 放在后面,因为它使用 AT&amp.T 的语法, 需要一个长的介绍.Nasm 调用时应该带上 ELF 格式选项("nasm -f elf hello.asm"). 产生的目标文件用GCC 来链接("gcc hello.o"), 产生最终的 ELF 二进制代码. 下面的这个脚本可用来编译 ASM 的模块. 我尽量把它写得简单, 所以所有它做的就是接受传给它的第一个文件名, 用 Nasm 编译, 用 GCC 来链接。 #!/bin/sh # assemble.sh outfile=${1%%.*} tempfile=asmtemp.o nasm -o $tempfile -f elf $1 gcc $tempfile -o $outfile rm $tempfile -f #EOF基本知识: 当然最好的就是在了解系统细节之前从一个例子开始. 这里是一个最基本的"hello-word" 形式的程序: . asmhello.asm global main extern printf section .data msg db "Hellooooooo, nurse!",0Dh,0Ah,0 section .text main: push dword msg call printf pop eax ret . EOF