来源：6.828->Operating System Engineering->Labs->Tools

作者：Sim

转载：

链接：

开始日期：2017-1-10

截止日期：

问题：

**1 开发环境框架**

**编译xv6 kernel的环境+运行xv6 kernel的环境**

**1.1 编译xv6 kernel的环境**

   首先，课程强烈推荐使用**雅典娜神机**(MIT Athena machines which run linux），然而我们并没有。退而求其次，我们可以选择虚拟机+Linux（物理机+Linux也可以）的方式来编译xv6 kernel，比如常见的选择：

      virtualBox（或vmWave）+Ubuntu

   选择好开发环境后，接下来准备编译工具链。通常我们的Linux系统已经默认支持6.828的工具链，可以用以下两条命令测试：

1. $ objdump -i   //在第二行可以看到打印： elf32-i386
2. $ gcc -m32 -print-libgcc-file-name    //通常打印：/usr/local/gcc/i486-linux-gnu/version/libgcc.a 或者/usr/local/gcc/x86\_64-linux-gnu/version/32/libgcc.a

   如果两条命令测试成功，说明我们的编译环境没问题。否则需要下载开发环境，下载命令如下：

1. $ sudo apt-get install -y build-essential gdb

   此外，64位的机器上可能需要安装32位的库支持(原因是可能会产生符号链接错误：\_\_udivdi3 not found and \_\_muldi3 not found)，安装命令如下：

1. $ sudo apt-get install gcc-multilib

   当然，我们也可以构建自己的编译工具链，虽然这样会比较麻烦，但可以提升效率以及加深自己对编译环境的了解。

  构建步骤：<https://pdos.csail.mit.edu/6.828/2016/tools.html>  ->Building Your Own Compiler Toolchain

**特别注意，经测试，即使以上命令都测试（配置）成功，在编译xv6时也可能报错：**

**1.1.1 错误1：boot block too large: xxx bytes (max 510) //我电脑上 xxx bytes为： 610**

    错误的原因是Ubuntu自带的gcc版本过高，导致编译的bootblock超过510字节，解决办法是降低gcc版本(官方表示gcc 3.x满足)，降低步骤如下：

        1.更新带有gcc 3.x apt 源：

                $ sudo vim /etc/apt/sources.list ，最后加入：

deb http://snapshot.debian.org/archive/debian/20070730T000000Z/ lenny main

deb-src http://snapshot.debian.org/archive/debian/20070730T000000Z/ lenny main

deb http://snapshot.debian.org/archive/debian-security/20070730T000000Z/ lenny/updates main

deb-src http://snapshot.debian.org/archive/debian-security/20070730T000000Z/ lenny/updates main

$ sudo apt-get update

            2.下载gcc-3.4 和 g++-3.4 :

                    $ sudo apt-get install gcc-3.4

                    $ sudo apt-get install g++-3.4

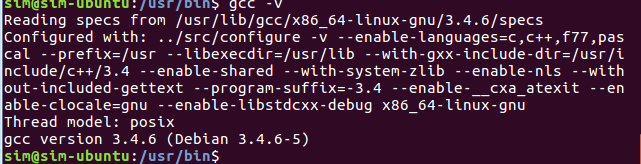
            3.更改系统gcc符号链接：

                    $ sudo ln -sf  【低版本gcc路径】 【符号链接路径】

                        在我的电脑上，低版本gcc路径：/usr/bin/gcc-3.4，符号链接路径：/usr/bin/gcc

                    $ sudo ln -sf /usr/bin/g++-3.4 /usr/bin/g++

               $ gcc -v 可以看到gcc版本已经降低



**1.1.2 错误2：cannot find crti.o: No such file or directory等**

         错误原因是库没找到，或者缺少该库。

       1. 先看看这些库文件是否存在

            $ find /usr/ -name crti\*

            能找到说明存在，不是缺少的问题，没找到则表示没有这些库，需要去下载

        2. 如果存在，却找不到，说明环境变量配置有问题。

            $ vim .bashrc，最后加入:

             LIBRARY\_PATH=/usr/lib/x86\_64-linux-gnu:$LIBRARY\_PATH

                export LIBRARY\_PATH

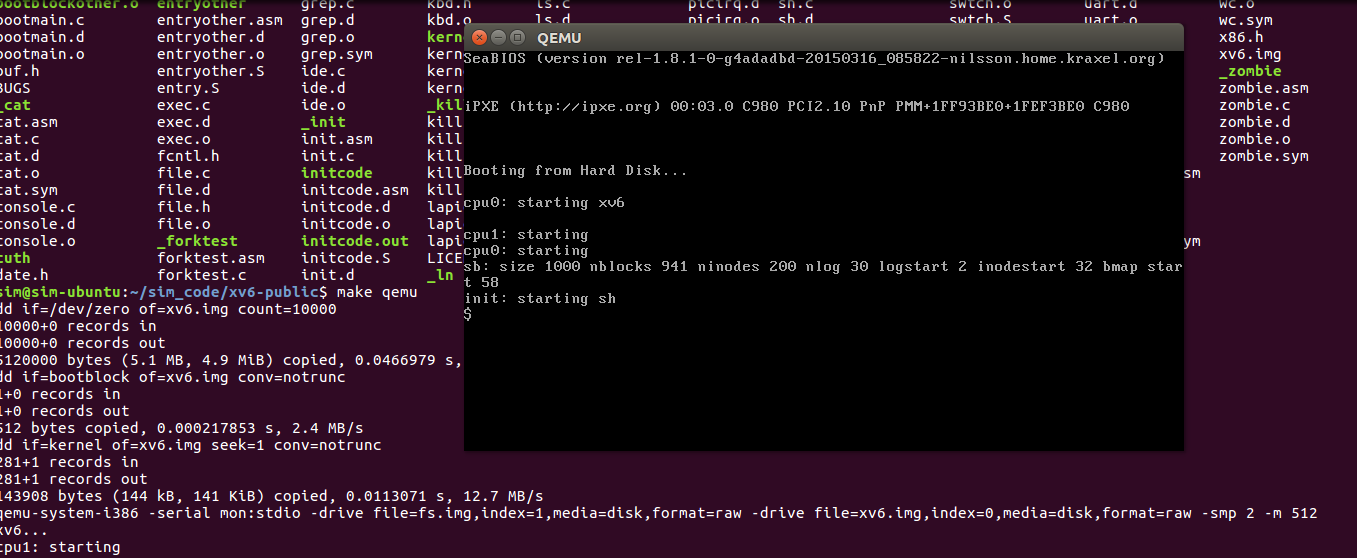
                (库路径以1 中找到的实际路径为准)

               $ source .bashrc ，使环境变量立即生效

**1.1.3 错误3：/usr/bin/ld: connot find -lgcc\_s**

        问题其实很简单，主要是libgcc\_s.so.1的符号链接挂了（为什么挂了不清楚），解决办法推荐看此笔记，讲得很清楚：<http://www.cnblogs.com/cassvin/archive/2011/07/24/Linux_Qtopia_firstBlogOncnblogs.html>

    编译成功后，直接执行： make qemu ，即可看到xv6在qemu中启动，如下图



**1.2 运行xv6 kernel的环境**

   QEMU (an x86 emulator)

   首先，还是课程的推荐：原版的QEMU虽然功能强大，但是不稳定，为此课程提供补丁支持，补丁的构建过程如下：

* 获取QEMU源码：clone IAP 6.828 QEMU git respository到本地，命令如下：

1. $ git clone http://web.mit.edu/ccutler/www/qemu.git -b 6.828-2.3.0

* 安装相应库： libsdl1.2-dev,libtool-bin,libglib2.0-dev,libz-dev,libpixman-1-dev.
* 配置源码，配置命令如下：

1. ./configure --disable-kvm [--prefix=PFX] [--target-list="i386-softmmu x86\_64-softmmu"]
2. (PFX：安装目录)
3. 实际安装中执行的命令： ./configure --disable-kvm --target-list="i386-softmmu x86\_64-softmmu"
4. (执行--prefix=PFX时报错，去掉该选项，安装在默认位置/usr/locals)

* 安装QEMU：make && make install （需要root权限，否则make install会失败）
* 安装成功后，执行：qemu-system-i386可打开虚拟机，否则表示安装失败