实验一 开发环境的搭建与配置

【实验目的】

1. 熟悉嵌入式Linux开发平台。
2. 掌握嵌入式Linux开发平台的开发环境搭建与配置。
3. 了解minicom配置串口通信参数的过程。
4. 了解嵌入式Linux的启动过程。
5. 掌握程序交叉编译运行及调试的一般方法。

【实验内容】

1. 连接实验开发板与宿主机。
2. 在虚拟机中的Ubuntu 10.04（宿主机）搭建开发环境。
3. 在宿主机中配置minicom。
4. 分析嵌入式Linux的启动过程。
5. 在宿主机上编写简单的C语言程序并用交叉编译工具进行编译，然后传输到目标机上运行。

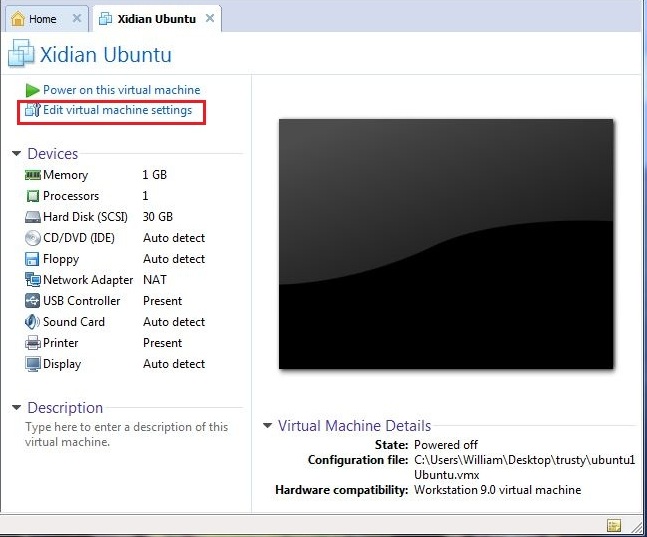
6) 在宿主机上编写简单的C语言程序并用交叉编译工具进行编译，用gdbserver进行远程调试。

【实验步骤】

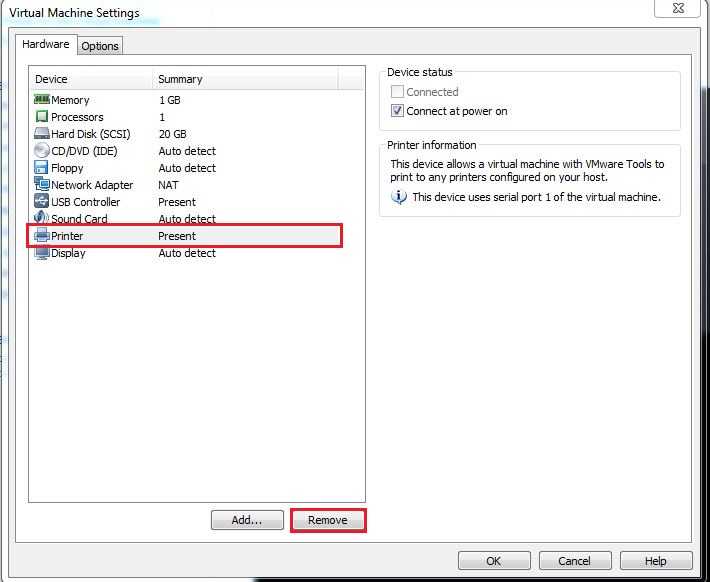
连接实验开发板，对虚拟机进行设置

1)首先把实验开发板打开，串口线连接到开发板上到com0,连接网线，USB线并连接电源线。

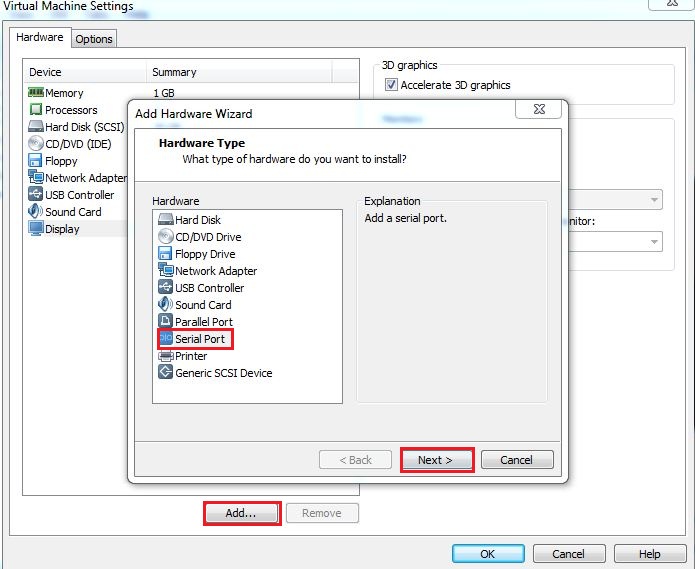
2）在桌面上点击打开vmware 软件，选择“编辑虚拟机设置”，如下图所示



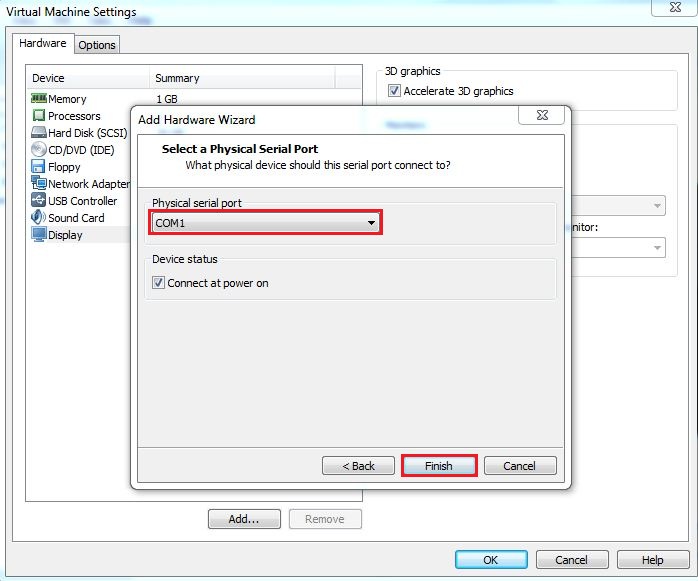
3)把打印机“Printer”设置删掉，如下图，



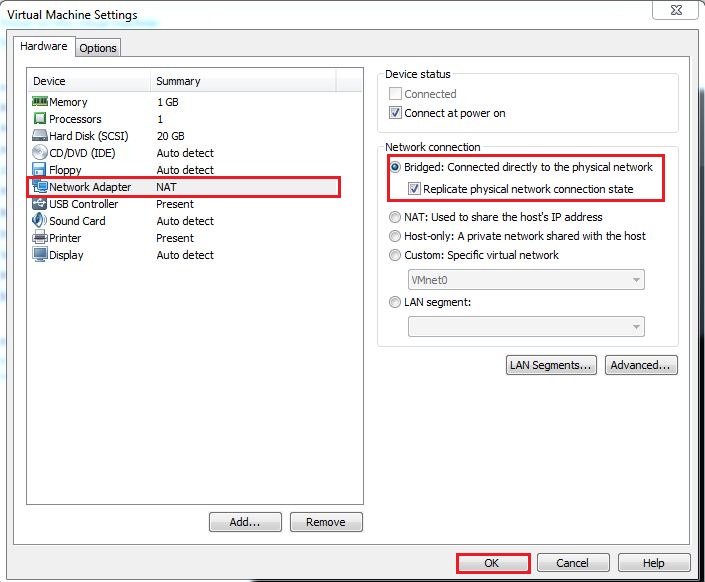
4）在虚拟机设置中添加串口“Serial Port”. 如下图。



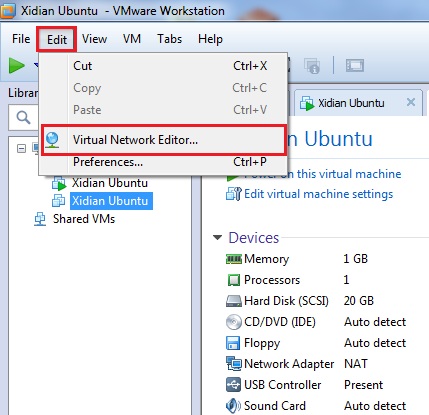
5）把串口设用COM 1. 如下图.其他选项直接点“Next” 和“Finish”



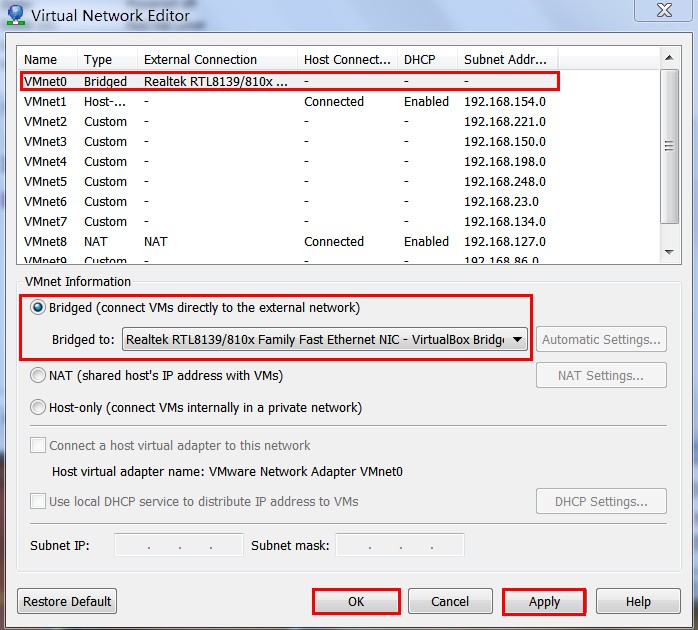
6）把网络连接方式设置为“桥接方式”，然后点击“OK”，如图所示：



7）选择虚拟机的“Edit”、“Virtual Network Editor...”，如下图所示：



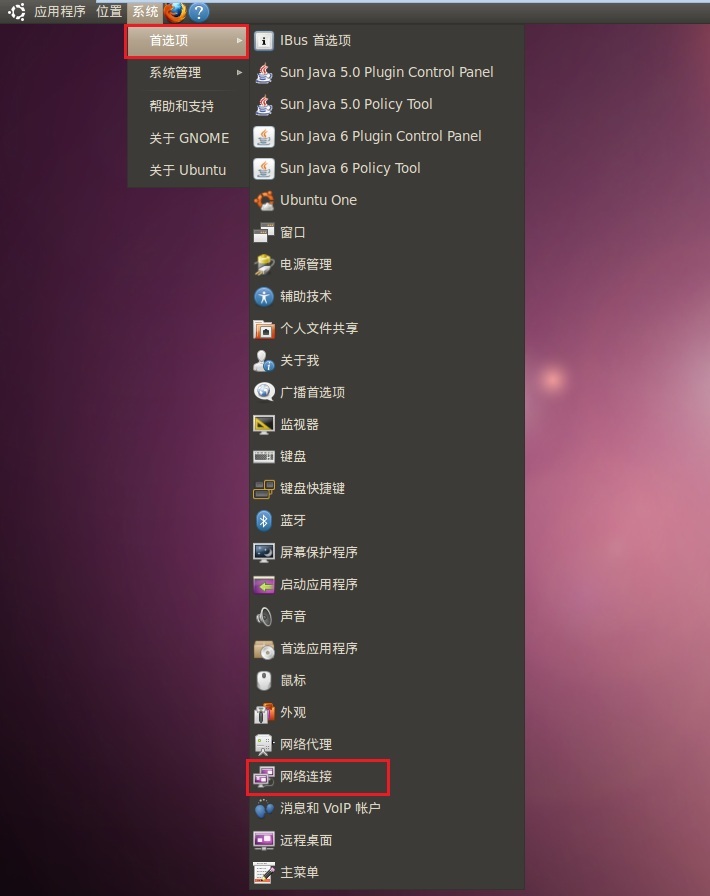
8）进入虚拟机网络参数设置界面后对VMnet0进行设置（注意这里桥接的网卡应选择与实验开发板相连 接的那块儿网卡），然后点击“Apply”、“OK”如下图所示：



9）上述设置完成后启动”Xidian Ubuntu”,选择其他用户，用户名为 “root”,密码为 “xidianembed”

### Ip地址配置

1）系统启动了以后，就得把IP地址配置下来。从菜单选，“系统”->”首选项”->“网络连接”。



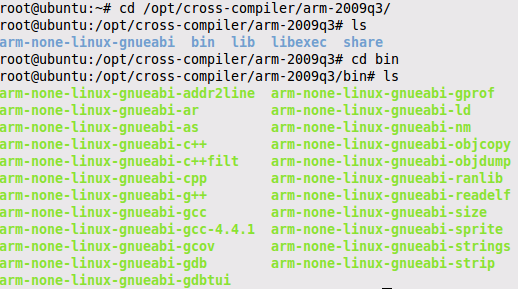
2） 在出的窗口下选“有线”->“Auto eth1(或eth0)”->”编辑“。



 3）在新的窗口下选“IPV4设置”->方法“手动”->”添加“->添加ip地址，子网掩码，和网关->点击”应用“，然后关闭。

工具链的配置

1）交叉编译工具链位于/opt/cross-compiler/arm-2009q3/可进入/bin/ 目录中查看所有梦用到的编译工具。



2）为了之后在任何目录下都能够使用上述目录下的命令，必须对系统环境变量PATH进行设置。这个可以通过修改/etc/bas.bashrc文件来实现.我们这就会使用vi编译器来查看本文件的内容。



3）在本文件最后部分添加 “export PATH=/opt/cross-compiler/arm-2009/bin:$PATH” 然后保存并退出。

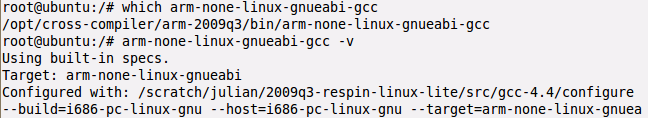
如果已经有的话就不用再写了。

4）使用source 命令来使路径生效。



5）使用echo $PATH 查看环境变量PATH的当前值，看修改是否生效。

6）也可以使用which命令。也可以查看编译器的详细设置。如下图。



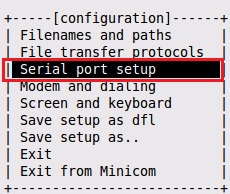
注意：如果路径没有生效，则可以查看export 的那个路径是否写对了，如果写对了但是没有生效的话可以重启一下Xidian Ubuntu 系统，再使用which命令来查看路径。

进入minicom软件，配置串口通信参数

1）在终端中输入minicom –s，进入minicom配置界面。



在弹出的窗口选择“Serial port setup”进行串口设置。安回车。



有关串口通信选项的含义：

"Filenames and paths"：选择需要传输的文件和路径

"File transfer protocols"：选择传输文件的通信协议

"Serial port setup"：设置串口通信参数

"Save setup as dfl"：将设置好的各项参数保存为dfl

"Save setup as"：将设置好的各项参数保存为自定义的文件名

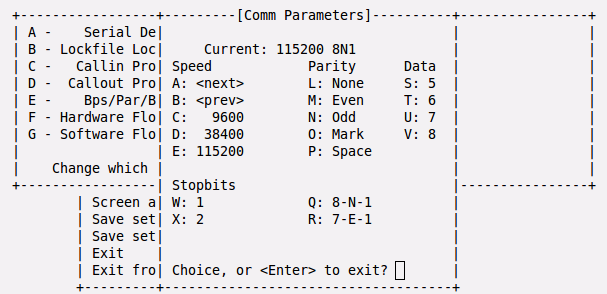
"Exit"：退出返回到minicom设置好后的终端

"Exit from Minicom"：从minicom命令中退出返回Linux终端

2）串口通信口的选择：（A - Serial Device）

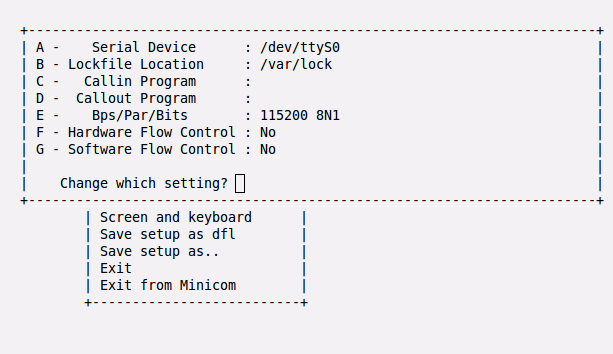
按"A"键把光标移动到Serial Device。如果串口线连在PC机的串口1上，则把Serial device设置为/dev/ttyS0。如果连在串口2上，则把Serial device设置为/dev/ttyS1，然后按下回车键。

3）串口参数的设置（E - Bps/Par/Bits）

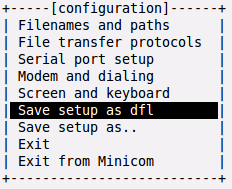
按"E"键来设置通信波特率、数据位、奇偶校验位和停止位（可以通过按下不同的键来设置通信参数）。分别按"I"、"V"、"L"、"W"键把波特率设为115200，数据位设为8，奇偶校验位设为无，停止位设为1。设置完后按"Esc"。

4）数据流的控制选择（F - Hardware Flow Control、G - Software Flow Control）

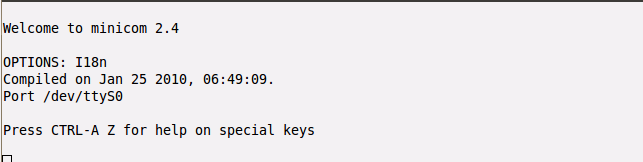
按"F"键可以完成硬件流控制切换，即完成"Yes"与"No"之间的切换。按"G"键完成软件流控制切换，即完成"Yes"与"No"之间的切换。下图显示的是串口配置好的后完整信息。



5）配置完成后，按下“Esc”键，将会出现下图的配置菜单。选择“Save setup as dfl”按ENTER键来保存。

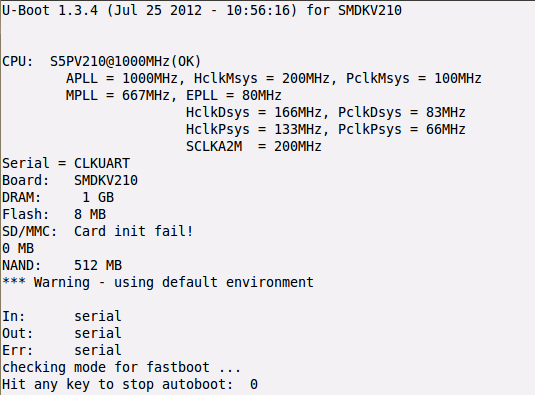


6）配置完之后选择Exit退出菜单进入已设置好的串口。会有以下提示。



实验开发板的启动

1）拨动实验开发板的电源开关，启动目标板。



在本过程中不要在键盘按任何建，要不就会进入Bootloader.请耐心等待到系统提示“Please press Enter to activate this Console”, 这时按回车就行。

嵌入式Linux系统的启动过程分析

1. 启动Bootloader

Bootloader是嵌入式系统的引导加载程序，它是系统上电后运行的第一段程序，其作用类似于PC机上的BIOS。在本系统中这段程序的起始地址为0x00000000。Bootloader在完成初始化RAM、初始化串口、检测处理器类型、设置Linux启动参数后，开始调用Linux内核。本系统Linux内核镜像zImage放在Flash中，Bootloader首先把它拷贝到RAM中，然后跳转到RAM中对zImage进行解压缩。解压缩后启动内核。

1. 加载内核

内核启动后先进行一系列与内核相关的初始化，然后调用第一个用户进程——init进程并等待用户进程的执行。具体的过程如下：

进行与体系结构相关的第一个初始化工作，首先通过检测出来的处理器类型进行处理器内核的初始化，然后进行内存结构的初始化，最后开启MMU，创建内核页表，映射所有的物理内存和IO空间；

创建异常向量表和初始化中断处理函数；

初始化系统核心进程调度器和时钟中断处理机制；

初始化串口控制台，在minicom中看到的系统启动过程中的信息都是通过串口输出的；

创建和初始化系统cache，为各种内存调用机制提供缓存，包括动态内存分配、虚拟文件系统及页缓存；

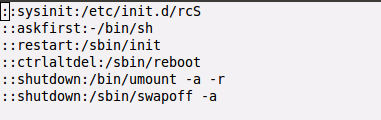
初始化内存管理，检测内存大小及被内核占用的内存情况；

初始化系统的进程间通信机制（IPC）；

创建init进程，结束内核的启动。

1. 执行init进程。

内核被加载后，第一个运行的程序便是/sbin/init，init进程是所有进程的发起者和控制者，它的进程号是1。init进程首先读取/etc/inittab文件，并依据此文件来进行初始化工作（首先进行一系列的硬件初始化，然后通过命令行传递过来的参数挂载根文件系统。最后执行一些其它的进程）。下面是/etc/inittab文件的内容。



init配置文件每行的基本格式为“id:runlevel\_ignored:action:process”，其中某些部分可以为空。各部分的具体内容如下：

id：指定启动进程的控制终端，如果所启动的进程并不是可以交互的shell，应该会有个控制终端（在PC机上该字段表示配置行的惟一标识）。

runlevel\_ignored：该字段是忽略掉的，配置inittab时空着它就行了（在PC机上该字段用来配置所启动进程适用的系统运行级别）。

action：下面是对几种行为的描述。

|  |  |
| --- | --- |
| 行为 | 描述 |
| sysinit | 为init提供初始化命令脚本的路径 |
| respawn | 每当相应的进程终止时，重新启动该进程。 |
| askfirst | 类似respawn，主要用途是减少系统上执行的终端应用程序的数量。它将会促使init在控制台上显示“Please press Enter to active this console”的信息，并在重新启动之前等待用户按下“Enter”键 |
| ctrlaltdel | 当用户按下【Ctrl+Alt+Del】时执行相应的进程 |
| shutdown | 当系统关机时，执行相应的进程 |

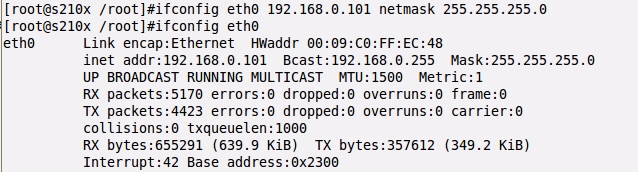
process：process为init执行的进程。

1. 执行/bin/login程序。

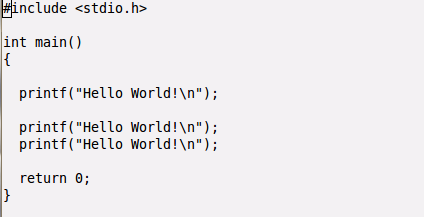
有些嵌入式系统在init进程执行完后会执行/bin/login。login程序会提示使用者输入账号及密码，接着编码并确认密码的正确性，如果账号与密码相符，则为使用者初始化环境，并将控制权交给shell，即等待用户登录。

本系统在执行完init进程后直接开始执行/bin/sh，进入shell交互程序（跳过了执行/bin/login这一步）。

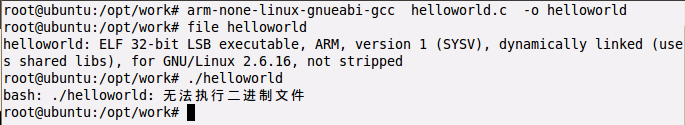
程序的交叉编译及运行

 1）用ifconfig 命令来查看并设置目标机的ip地址。默认的ip为192.168.0.101,可以不用该。用一下方式。

2）在宿主机的/opt/work/目录下用vi命令创建一个文件helloworld.c,编辑后退出并保存。



3)用arm-none-linux-gnueabi-gcc命令对helloworld.c进行编译。然后用file命令查看新生成的文件helloworld的信息。最后看helloworld能否正常运行。



4)用arm-linux-gcc编译生成的文件只能在嵌入式系统中运行，下面用scp命令把文件helloworld拷贝到目标机的/root目录下，然后运行该文件，查看运行结果。

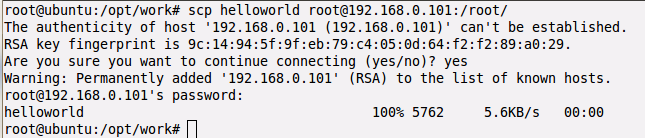
用scp命令可实现文件传输，它有多种格式，下面只列出其中的两个，分别用来把文件从远程复制到本地和从本地复制到远程。

scp remote\_username@remote\_ip:remote\_file local\_folder

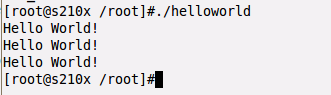
scp local\_file [remote\_username@remote](mailto:remote_username@remote)\_ip:remote\_folder

在第一次使用scp命令时会提示“Do you want to continue connecting?(y/n)”直接输入个“y“按回车，输入正确的密码（宿主机root用户的密码为“xidianembed”），连接成功后就可以直接进行文件拷贝了。

以后再使用scp进行文件拷贝时，连接已经存在，只用输入宿主机密码就可以进行文件拷贝了。

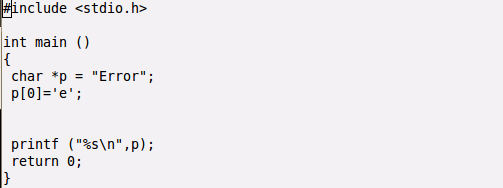


5）现在可以在目标机上执行helloworld



gdbserver远程调试

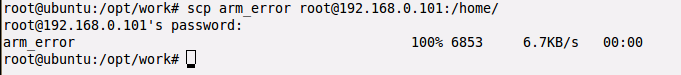
1. 在宿主机的/opt/work/目录下用vi命令创建文件arm\_error.c，编辑后保存并退出。



1. 使用arm-none-linux-gnueabi-gcc命令编译arm\_error.c。注意这里要加上参数“-g”，它表示附加调试信息，不加的话后面将无法进行调试。



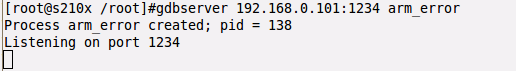
3)用scp命令把arm\_error拷贝到目标机的/root目录下。



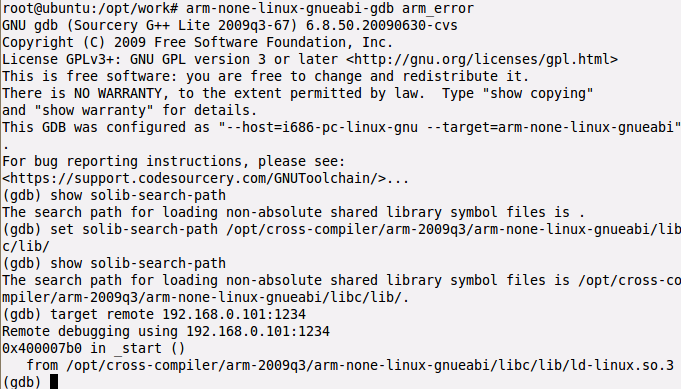
4）在目标机的/root目录下运行arm\_error，查看运行结果。



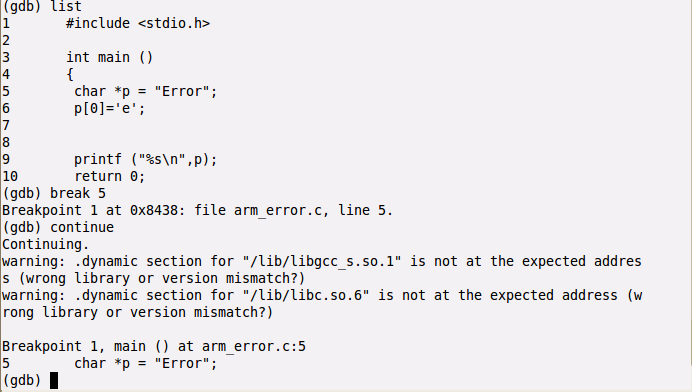
5)在目标机上启动gdbserver。



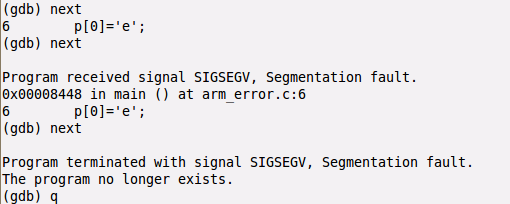
6)在宿主机上启动arm-none-linux-gnueabi-gdb，show命令查看一下solib-search-path是否已经配置，如果没有则通过set命令进行配置。然后通过target remote命令连接到目标机的1234端口。



7)下面开始对arm\_error进行调试，用list命令列出源程序，用break命令添加断点，然后用continue命令继续运行程序（目标机中已经启动了arm\_error，该程序被gdbserver阻塞）。在这的告提示是可以不管的。



8）用next命令进行单步调试。



在调试的过程中，很容易发现问题出在“p[0] = ‘e’;”语句上。因对一个字符串常量进行修改而发生了段错误。

9）查看目标机上的显示信息。

