实 验 报 告 书

课程名称 ARM原理及体系结构实验

系（部）： 智能装备学院

专业班级： 电科18-2

学生姓名： 孙启凯

学生学号： 201823010225

1. **实验原理**
2. 本地编译

我们之前常见的软件开发，都是属于本地编译：在当前的PC下，x86的CPU下直接编译出来程序，其可以直接在当前的环境（即x86的CPU下）中运行。此时的编译，可以叫做本地编译，即在当前目标平台下编译出来的程序，也只是放到当前平台下就可以运行的。

1. 交叉编译

交叉编译是一个和本地编译相对应的概念，当我们开始接触嵌入式开发后，事情变的不一样了，你在电脑上写程序，在电脑上编译出可执行文件，最后这个可执行文件需要下载到你的开发板上运行。程序最后运行的环境变了，比如你的开发板是基于 ARM 的——程序在X86上编辑和编译，最终运行在另一个和X86完全不同的架构的 Arm 芯片上。

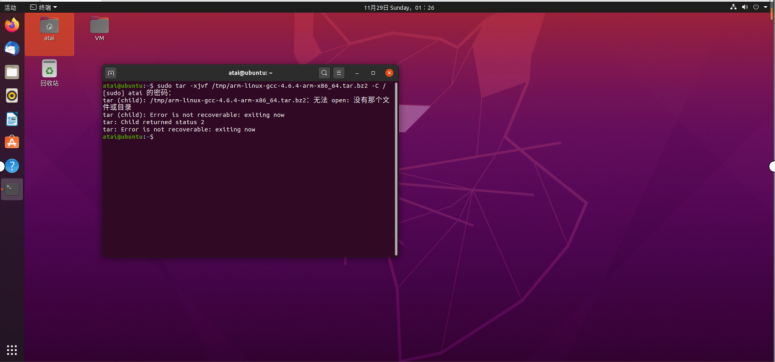
之所以整个流程变成了这个样子，这是由嵌入式系统的特性决定的：一般嵌入式系统里面使用的芯片性能都比较弱，而且绝大部分都不能像 X86 一样运行 Windows/Ubuntu 桌面系统，即使能运行，性能也很弱，无法给你提供一个在开发板上写代码、编译代码的环境。所以我们还是离不开 X86 电脑强大高效的桌面环境进行软件开发。

但是这样有一个问题，X86、Arm、MIPS、RISC-V 这些芯片，它们的指令集是由不同的组织或者公司设计的，彼此并不兼容——Arm 和 MIPS 的 CPU 无法运行以 X86 的指令集编码的程序，反之亦然。所以我们要在 X86 的电脑上编译出能够在 Arm 上运行的程序，我们必须明确告诉编译器，编译生成的可执行文件需要以 Arm 指令集的标准编码。为了让这个流程变得简单，开发者们为不同的芯片开发了不同的编译器，比如针对 Arm 平台的 arm-linux-gcc，针对 mips 平台的 mips-linux-gnu-gcc，这些编译器都是基于 GCC 针对具体的架构指令集进行对应配置，所以它们在运行的时候就就会生成和该目标平台对应的可执行文件。

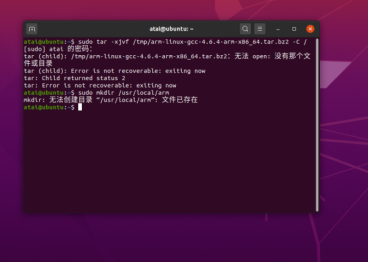
1. **实验工具**

装有虚拟机的电脑

1. **实验过程**
2. 通过网上搜索找到arm-linux-gcc-4.4.3.tar.gz的下载连接，我找的是一个网盘链接。在主机上下载好arm-linux-gcc-4.4.3.tar.gz后，需要将该文件拷贝到虚拟机上。
3. 安装arm-linux-gcc交叉编译器：
   1. Ubuntu有一个专门安装软件的工具——apt，正常情况可以通过apy直接安装arm-linux-gcc。但是因为这个功能国内还没有实现，需要通过连接国际网络。所以我们换一个比较麻烦的方式。先把下载好的安装包移动到根目录下的Tmp目录中,进行接下来的操作。使用tar命令解压安装包，即在Terminal中输入以下命令：（前面的sudo表示使用root权限执行该命令，需要输入本用户的密码）sudo tar -xjvf /tmp/arm-linux-gcc-4.6.4-arm-x86\_64.tar.bz2 -C /注意是大写的字母C，此命令会把安装包解压到根目录下的opt的TuxamitoSoftToolchains里面（/opt/TuxamitoSoftToolchains）。



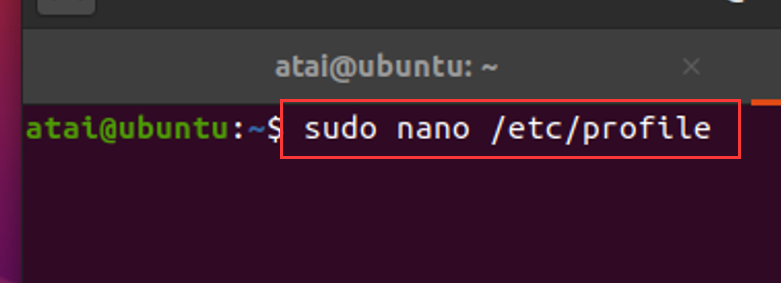
* 1. 解压完成后，再在（/usr/local）中创建一个新目录arm，即在Terminal中输入以下命令：sudo mkdir /usr/local/arm创建arm目录成功后，还需要给它解放全部权限，即在Terminal中输入以下命令（依环境而定，可能不需要）：sudo chmod 777 /usr/local/arm

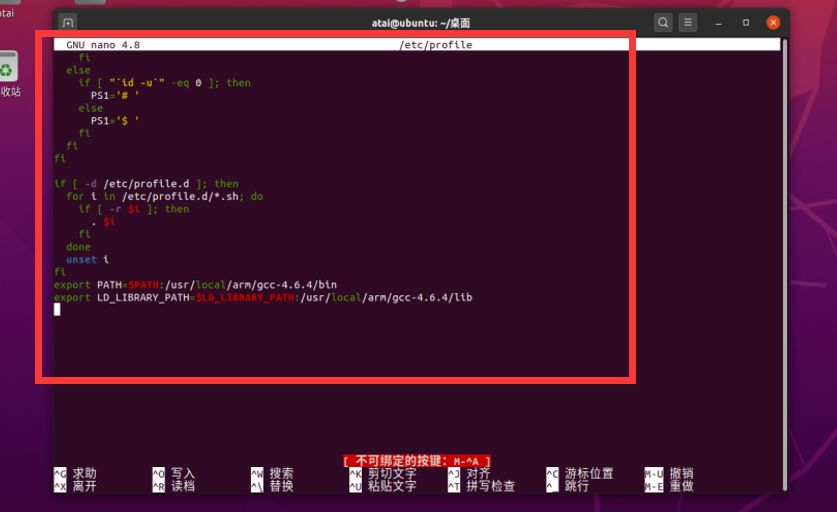


* 1. 在解压出来的目录中找到并把整个gcc-4.6.4目录复制到刚刚建好的arm目录中，命令如下：先cd切换到gcc-4.6.4所在目录（切换后先ls看一下有没有gcc-4.6.4目录）：cd /opt/TuxamitoSoftToolchains/arm-arm1176jzfssf-linux-gnueabi/再执行 cp 复制命令，-r 表示整个目录以及里面的任何东西sudo cp -r gcc-4.6.4 /usr/local/arm
  2. 打开（/etc/profile）配置环境变量和库变量，目的是以后可以在任何位置使用该交叉编译器，命令如下：

nano vi /etc/profile用vi或者vim打开后，这里推荐可以用nano，在文件最后添加两行，并输入以下代码：第一行是添加执行程序的环境变量，第二行是库文件的路径export PATH=$PATH:/usr/local/arm/gcc-4.6.4/bin

exportLD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/usr/local/arm/gcc-4.6.4/lib



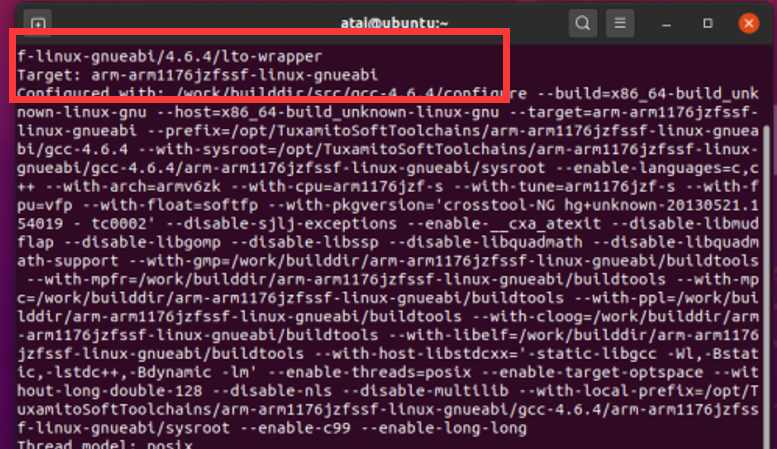


* 1. 使用source命令重新加载生效该配置文件，打开（/etc/profile）配置环境变量和库变量，目的是以后可以在任何位置使用该交叉编译器，命令如下

source /etc/profile

* 1. 检验是否安装成功，在 Terminal 输入以下命令输出版本信息：

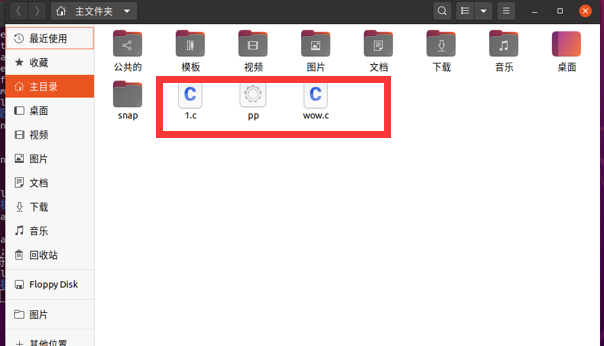
arm-linux-gcc -v

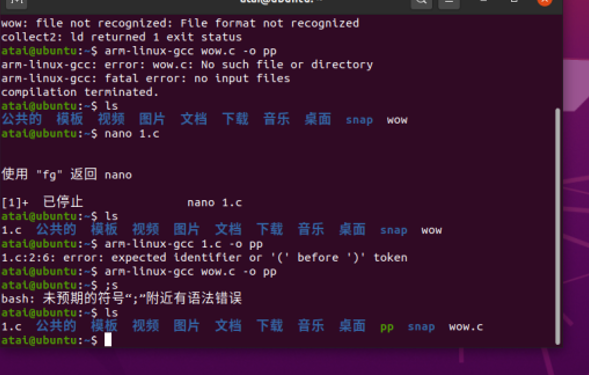


如上截图，版本为4.6.4.这样表示安装成功，可以进行编译了。

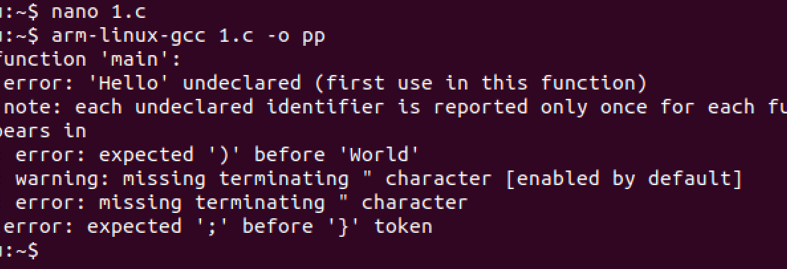
* 1. 再随便写一个sdkjdx.c文件，能编译成功说明已经完美安装。例如：arm-linux-gcc sdkjdx.c -o pp
  2. 再file命令查看编译后的是不是arm的可执行文件：

file pp





1. **实验结果**



1. **实验过程中存在的问题**

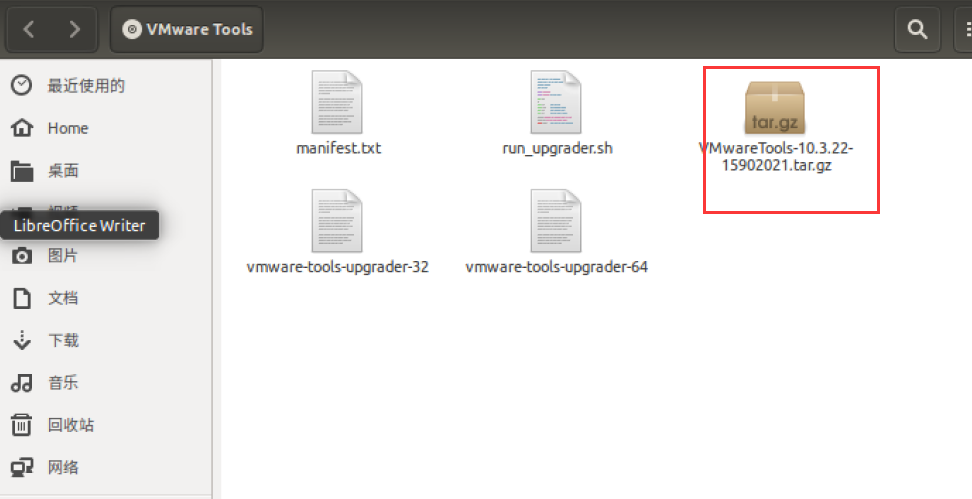
（一）无法将电脑上的arm-linux-gcc-4.4.3.tar.gz拷贝到虚拟机中。

解决办法：

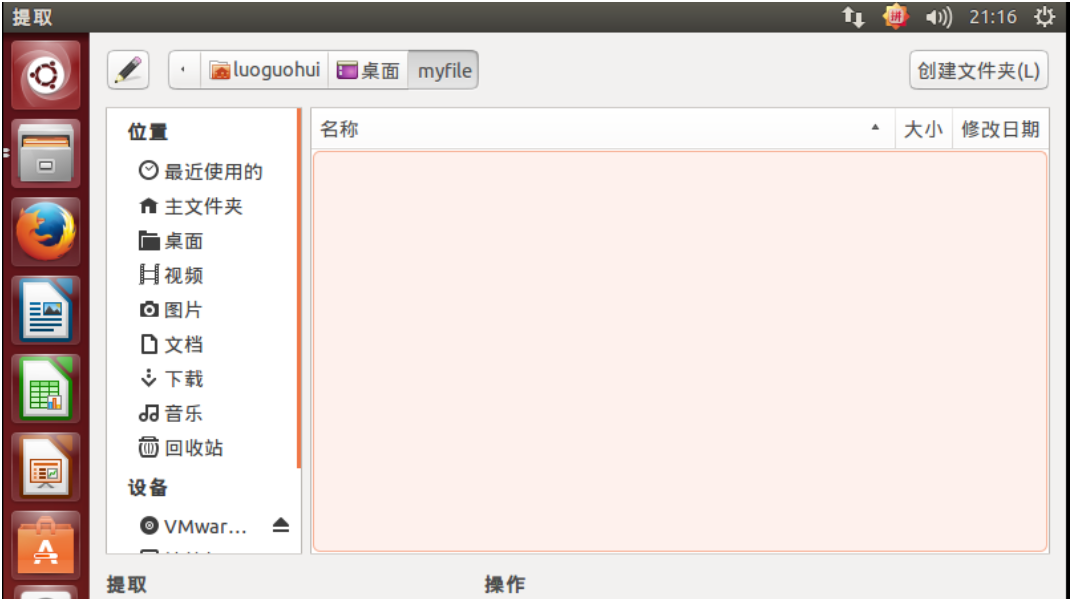
1. 打开虚拟机VMware Workstation，启动Ubuntu系统，菜单栏 - 虚拟机 - 安装VMware Tools，不启动Ubuntu系统是无法点击“安装VMware Tools”选项的，如下图：



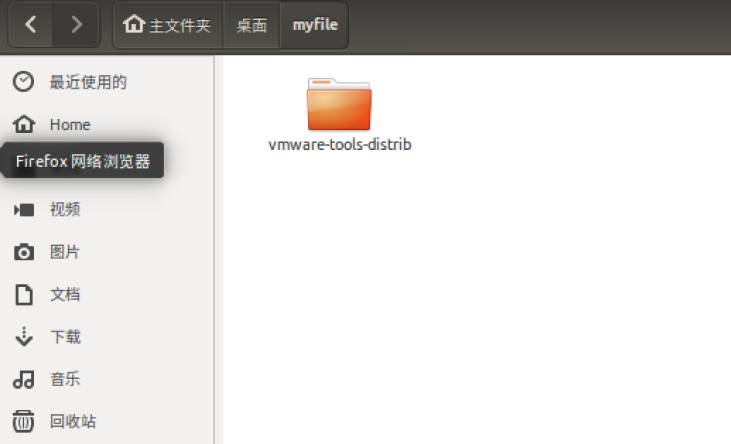
1. 此时，会发现虚拟机设备下多了VMware Tools这一项，点击它，其里面有一个VMwareTools…tar.gz文件



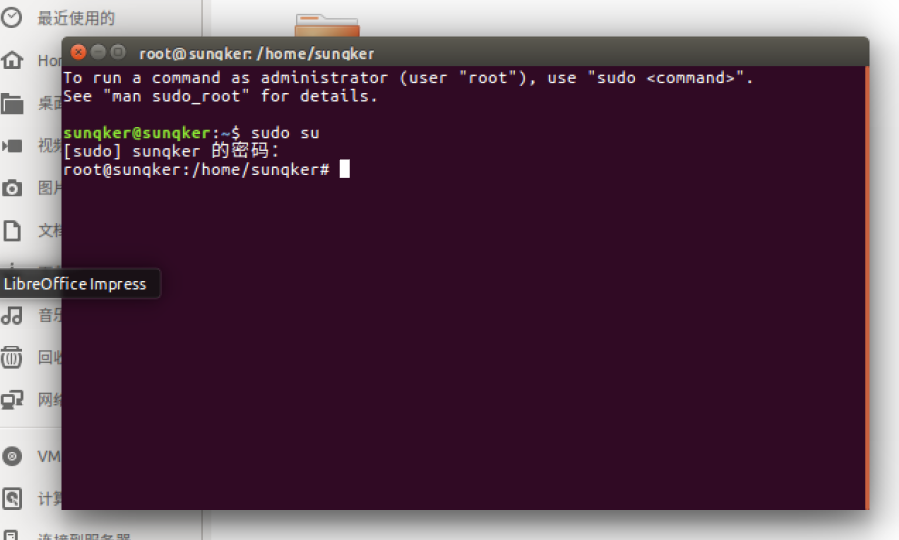
1. 接下来我们把VMwareTools…tar.gz文件提取到某个目录下，如下图提取到桌面刚刚新建的myfile目录下：



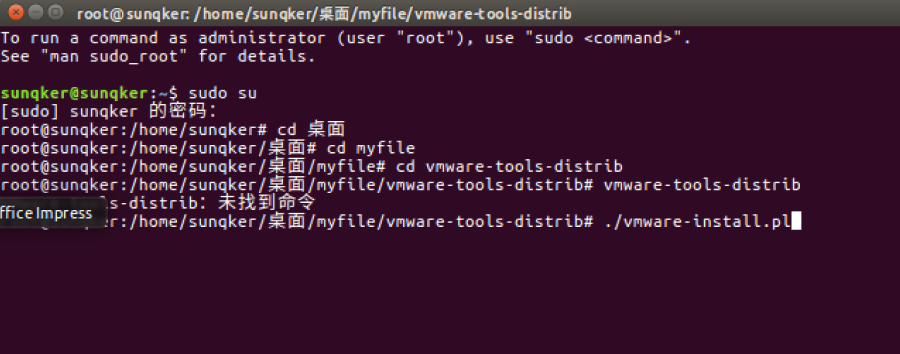
1. 提取完成后会发现桌面的myfile里面多了一个vmware-tools-distrib文件夹，这个正是我们待会安装需要用到的：



1. 下面我们快捷键Ctrl+Alt+T，启动命令框，然后切换到root用户（切换命令为：sudo su，回车然后会提示你输入当前登录用户的密码，输入成功后即可进入root用户）：



1. 然后我们以root用户进入到刚刚提取到的vmware-tools-distrib文件夹下，然后输入命令：./vmware-install.pl，然后回车，如下图：



1. 上面的操作后就开始安装VMware Tools了，根据其提示输入yes/no，直到出现Enjoy, –the VMware team如下图，就表示安装成功了，然后手动重启虚拟机：
2. **实验总结**

在这次实验中完成了准备交叉编译工具以及交叉编译环境的安装，还有交叉编译环境的简单熟悉。

通过自己的亲自动手配置交叉编译环境，学会了Linux虚拟机的安装以及简单的配置，学会了如何在虚拟机中安装arm-linux-gcc交叉编译器。

知道了交叉编译工具链是为了编译、链接、处理和调试跨平台体系结构的程序代码。对于交叉开发的工具链来说，在文件名称上加了一个前缀，用来区别本地的工具链。例如，arm-linux-表示是对arm的交叉编译工具链；arm-linux-gcc表示是使用gcc的编译器。除了体系结构相关的编译选项以外，其使用方法与Linux主机上的gcc相同，所以Linux编程技术对于嵌入式同样适用。不过，并不是任何一个版本拿来都能用，各种软件包往往存在版本匹配问题。例如，编译内核时需要使用arm-linux-gcc-4.3.3版本的交叉编译工具链，而使用arm-linux-gcc-3.4.1的交叉编译工具链，则会导致编译失败。

了解到gcc和arm-linux-gcc的区别是是gcc是linux下的C语言编译器，编译出来的程序在本地执行，而arm-linux-gcc用来在linux下跨平台的C语言编译器，编译出来的程序在目标机(如ARM平台)上执行，嵌入式开发应使用嵌入式交叉编译工具链。