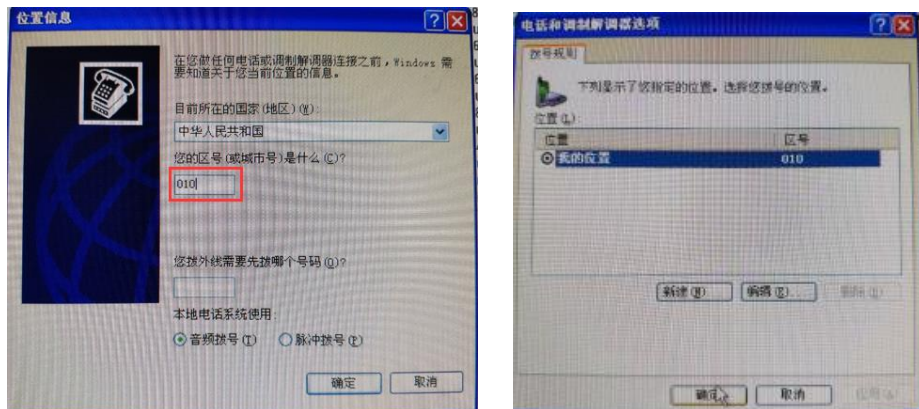


《嵌入式软件设计》实验报告

学院： 计算机学院 专业： 软件工程 2018 年 12 月 6 日

实验名称	熟悉嵌入式 Linux 开发环境				
姓 名	刘瑞康	课头号	20181021026	学 号	2016302580242
一、实验目的	四、实验步骤				
二、实验内容	五、实验结果（数据表格、现象等）				
三、实验设备及工具	六、实验结果分析（实验现象分析、实验中存在问题的讨论）				
一、实验目的 熟悉嵌入式 Linux 开发环境，学会基于 UP-CUP 经典 2440 教学科研平台的 Linux 开发环境的配置和使用 利用 arm-linux-gcc 交叉编译器编译程序，使用基于 NFS 的挂载方式进行实验，了解嵌入式开发的基本					
二、过程实验内容 ◆ 本次实验使用 Fedora8 操作系统环境,安装 ARM-Linux 的开发库及编译器。创建一个新目录，并在其中编 ◆ 写 hello.c 和 Makefile 文件。 ◆ 学习在 Linux 下的编程和编译过程，以及 ARM 开发板的使用和开发环境的设置。将已经编译好的文件通 ◆ 过 NFS 方式挂载到目标开发板上运行					
三、实验设备及工具 ◆ 硬件:UP-CUP2440 型嵌入式实验平台，PC 机 Pentium 500 以上，硬盘 40G 以上，内存大于 256M。 ◆ 软件:Vmware Workstation +Fedora Core 8 + MiniCom/Xshell + ARM-LINUX 交叉编译开发环境。					
四、实验步骤 实验目录:/UP-CUP2440/SRC/exp/basic/01_hello 1) 开发板配置 a. 将开发板接通电源并与主机连接。 b. 在 winXP 系统中，选择开始-所有程序-附件-通讯-超级终端 设置区号为 010。					



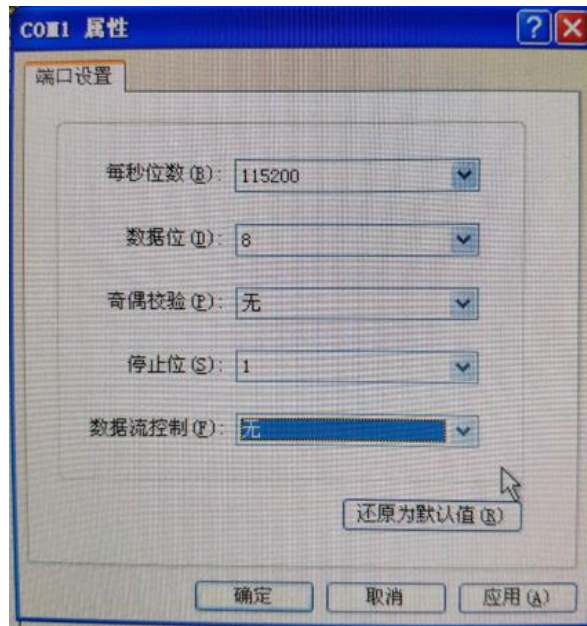
c. 设置连接的名称为 COM2440。



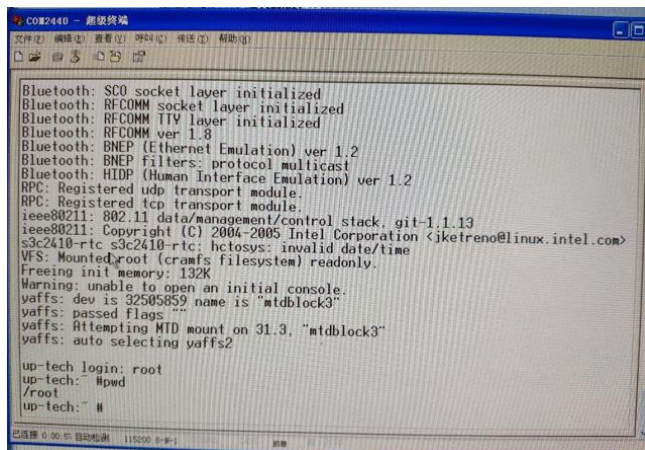
d. “连接时使用”选项选择 COM1。



e. 端口设置，配置选择如下图。



确定后，屏幕应显示空白内容，此时打开开发板上的开关，将会有输出内容，使用账户 root 和密码 pwd 登录。开发板配置完毕。

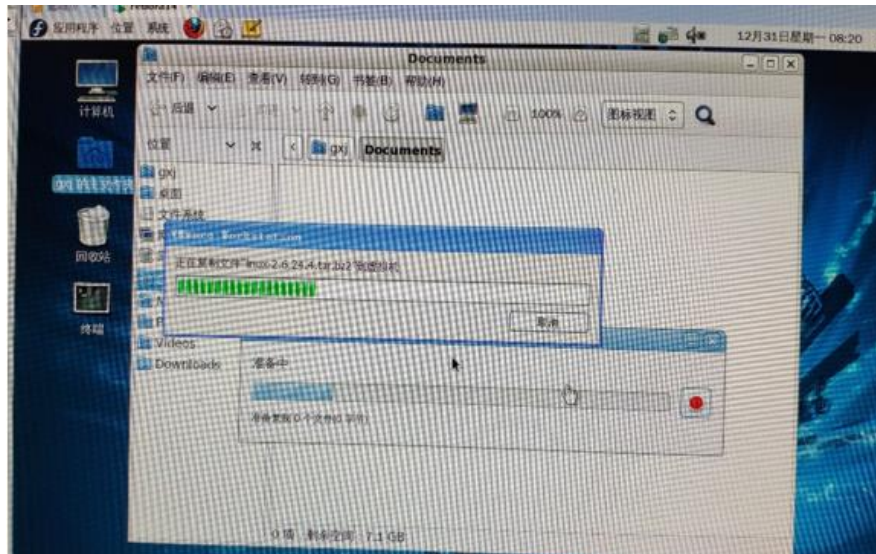


2) 配置编译环境

a. 打开 VMware，打开虚拟机。



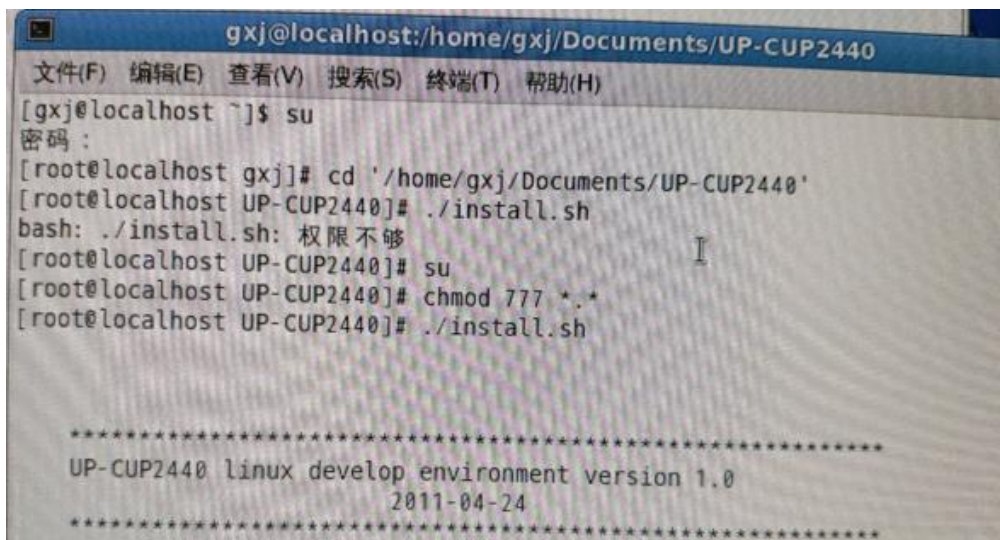
- b. 将 UP-CUP2440 文件夹拷贝到虚拟机的 Documents 文件夹中。



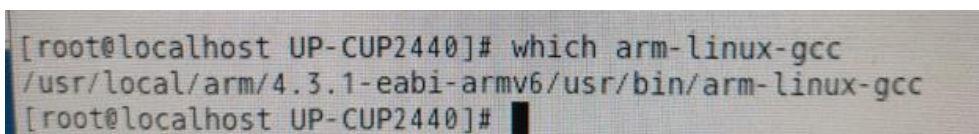
- c. 打开终端，使用 su 开启超级用户模式。



- d. 进入 UP-CUP2440 文件夹下，使用 chmod 授予最高权限后，输入 ./install.sh 开始部署交叉编译环境。



- e. 部署完成后使用命令 which arm-linux-gcc 确定是否部署正确，若输出路径则正确，编译环境配置完毕。



3) 配置网络设备

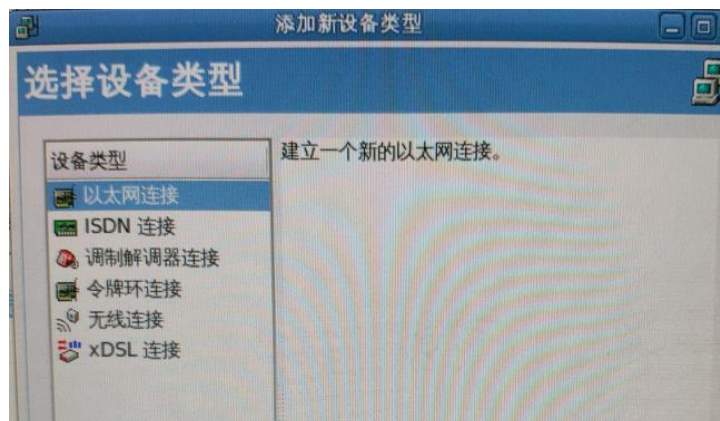
a. 选择 系统-管理网络。



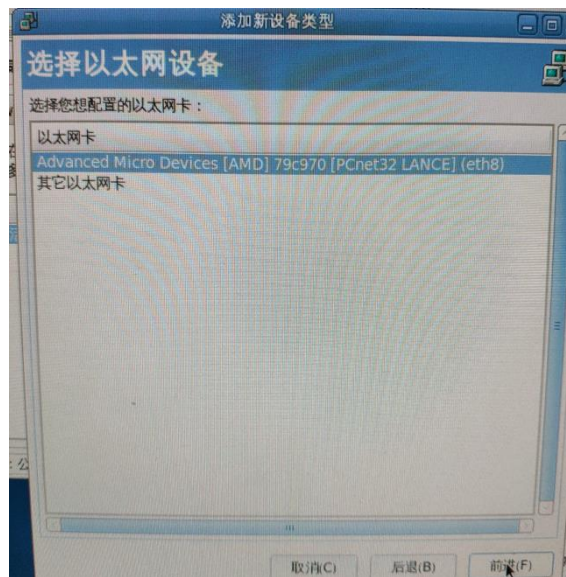
b. 点击新建



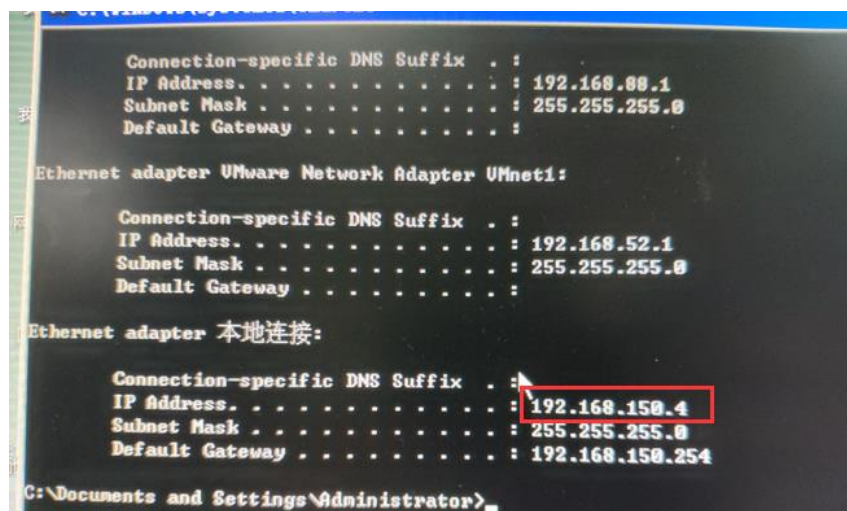
c. 类型选择以太网连接



- d. 设备选择 AMD，点击前进。



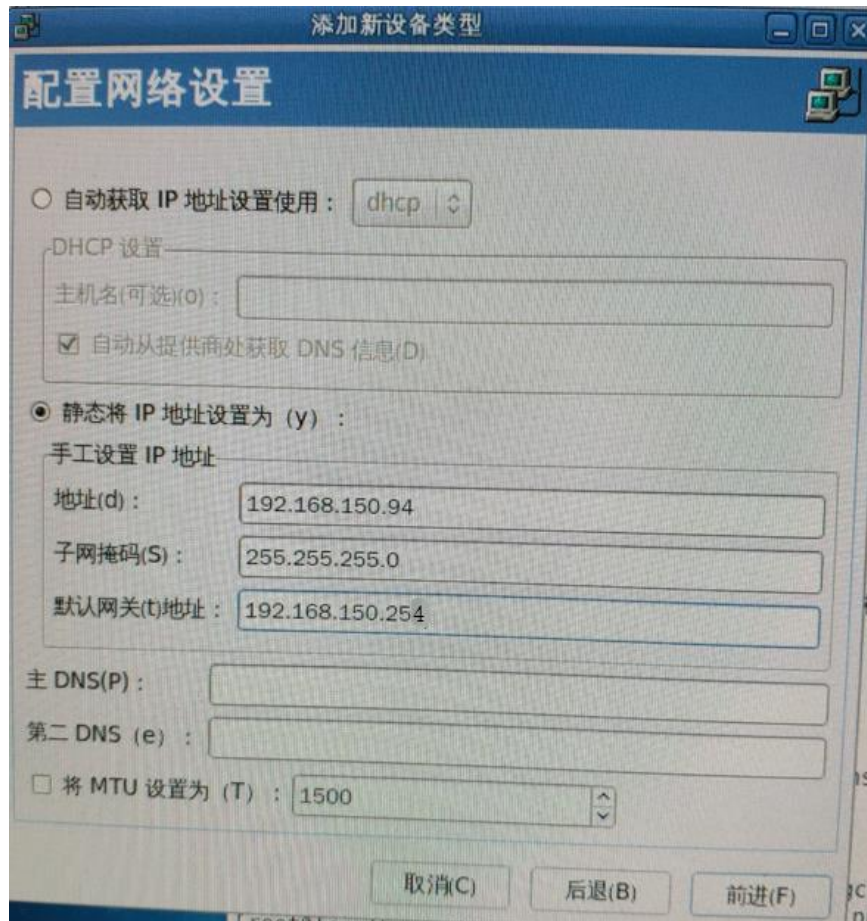
- e. 回到 winXP 系统，在命令行通过 ipconfig 查看本地连接 ip，下图为 192.168.150.4。



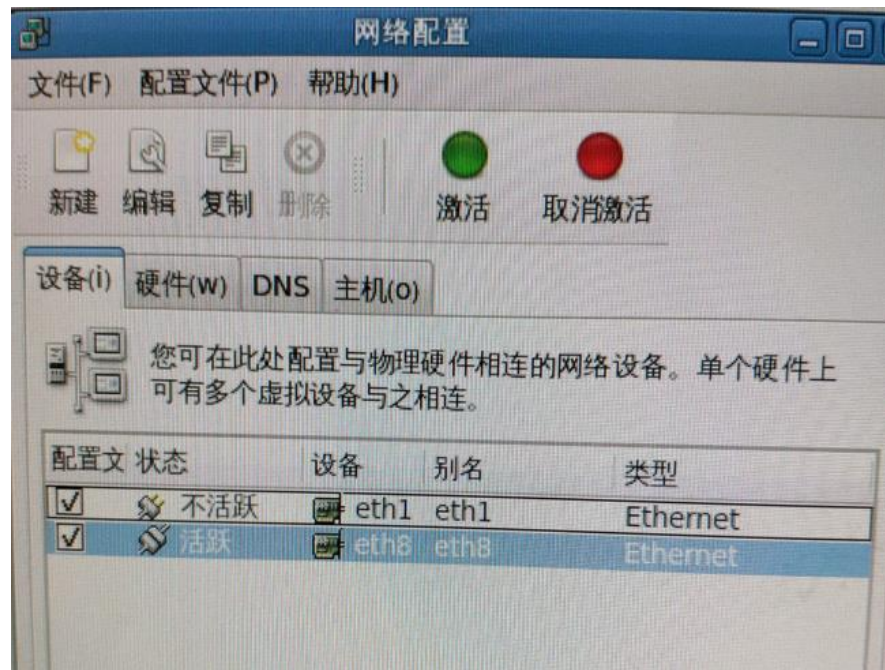
- f. 可以尝试在虚拟机终端中 ping 此 ip 确认是否能连接。

```
[root@localhost UP-CUP2440]# ping 192.168.150.4
PING 192.168.150.4 (192.168.150.4) 56(84) bytes of data.
64 bytes from 192.168.150.4: icmp_req=1 ttl=64 time=0.160 ms
64 bytes from 192.168.150.4: icmp_req=2 ttl=64 time=0.103 ms
64 bytes from 192.168.150.4: icmp_req=3 ttl=64 time=0.110 ms
64 bytes from 192.168.150.4: icmp_req=4 ttl=64 time=0.111 ms
64 bytes from 192.168.150.4: icmp_req=5 ttl=64 time=0.109 ms
64 bytes from 192.168.150.4: icmp_req=6 ttl=64 time=0.112 ms
64 bytes from 192.168.150.4: icmp_req=7 ttl=64 time=0.106 ms
64 bytes from 192.168.150.4: icmp_req=8 ttl=64 time=0.128 ms
^C
--- 192.168.150.4 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7009ms
rtt min/avg/max/mdev = 0.103/0.117/0.160/0.019 ms
[root@localhost UP-CUP2440]#
```

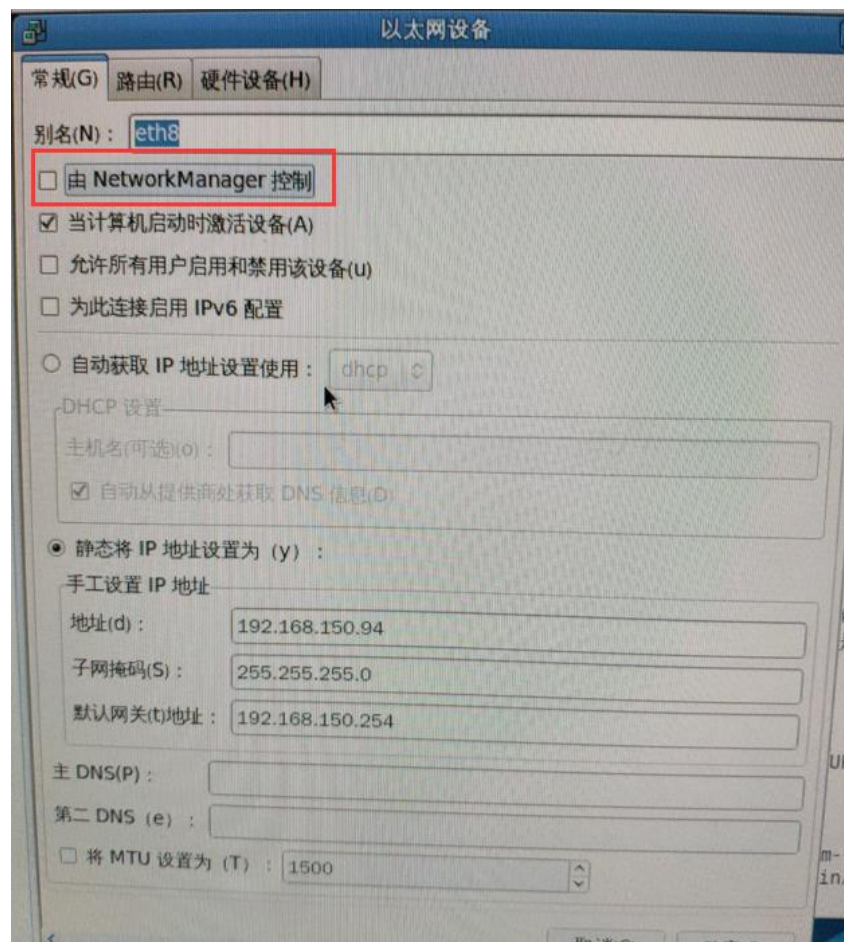
g. 在配置中选择静态设置 ip 地址，第一行前三字段设置为前一步获得的 ip 内容，第四字段随机设置，后续会用到该地址;第二行为 255.255.255.0；第三行前三字段同第一行，最后字段为 254。



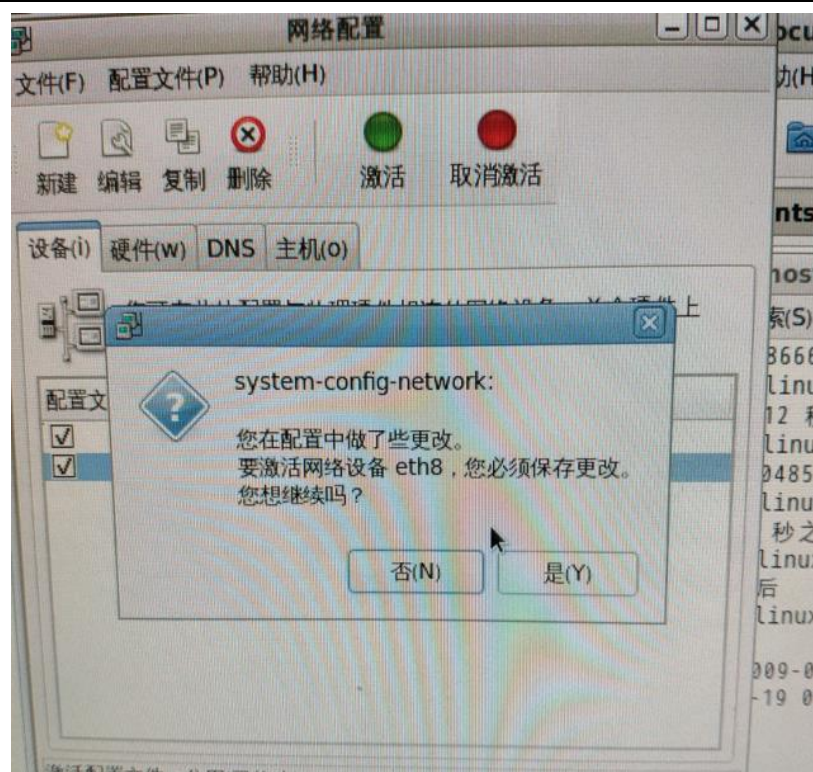
h. 选择刚刚添加的设备进行编辑



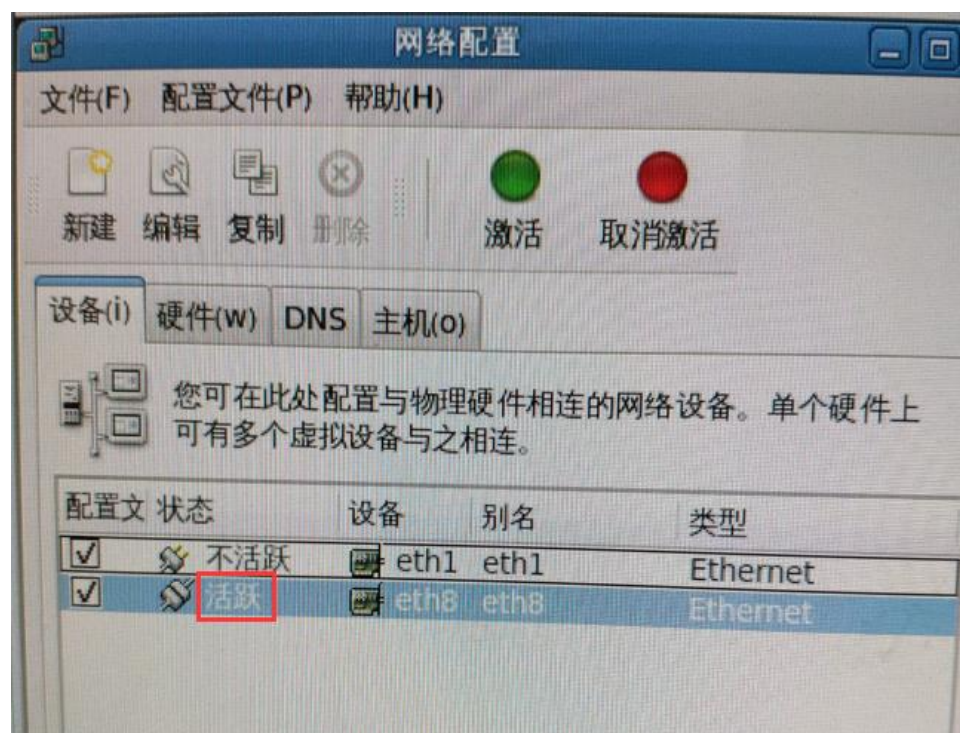
i. 将第一个复选框取消选中，点击确定返回



j. 保存更改并点击激活



k. 激活后设备应处于活跃状态



1. 此时可通过在终端输入 `ifconfig` 查看配置是否正确，若正确则网络设备配置完毕。

```

[root@localhost UP-CUP2440]# ifconfig
eth8      Link encap:Ethernet  HWaddr 00:0C:29:79:86:C3
          inet addr:192.168.150.94  Bcast:192.168.150.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe79:86c3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10 errors:0 dropped:0 overruns:0 frame:0
          TX packets:21 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:600 (600.0 b)  TX bytes:4429 (4.3 KiB)
          Interrupt:19 Base address:0x2024

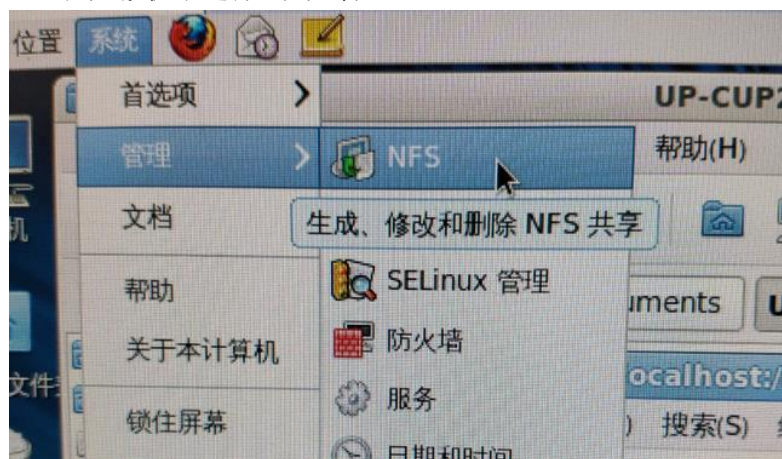
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:123 errors:0 dropped:0 overruns:0 frame:0
          TX packets:123 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:12624 (12.3 KiB)  TX bytes:12624 (12.3 KiB)

[root@localhost UP-CUP2440]#

```

4) 设置 NFS 挂载实验目录

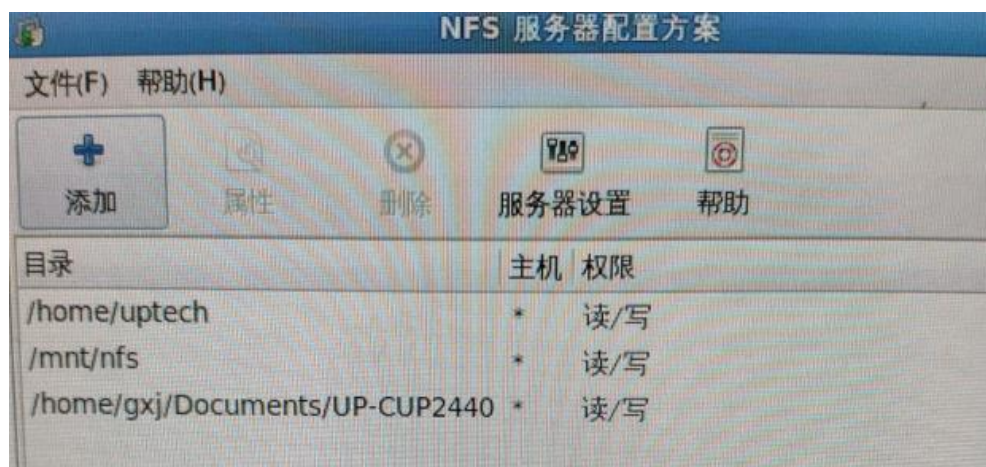
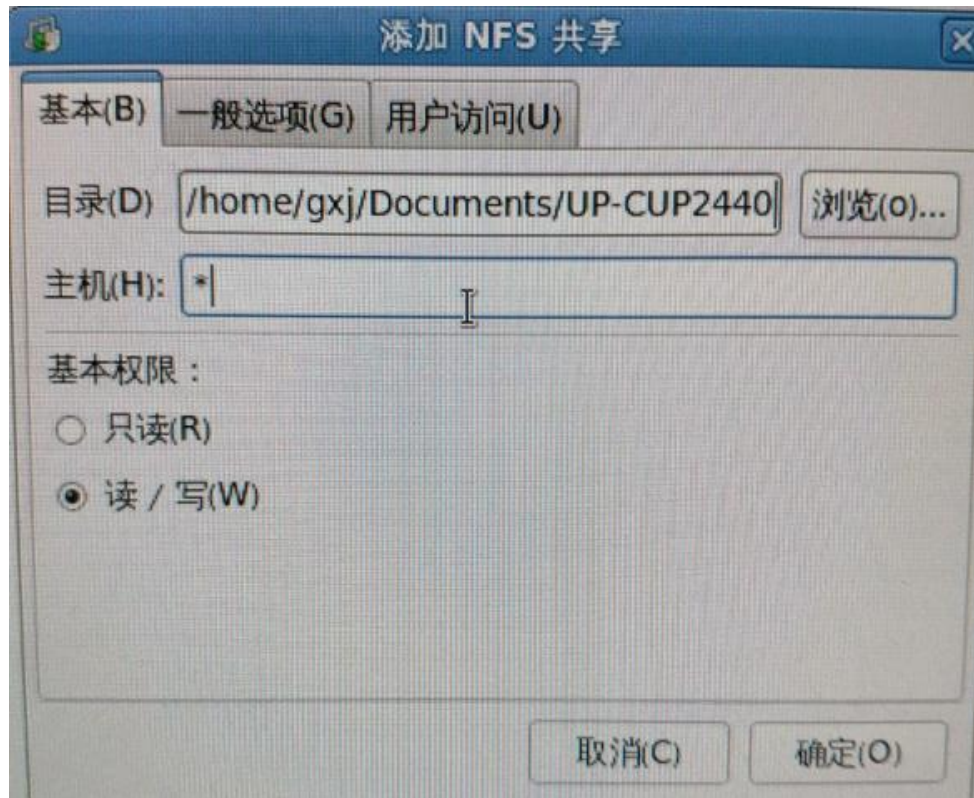
- a. 在虚拟机中选择 系统-管理-NFS。



- b. 添加配置



- c. 输入 UP-CUP2440 的文件夹目录，主机为任意，即*，权限为读写。



d. 打开超级终端，使用 ifconfig 设置开发板的 ip 为 192.168.150.199，并尝试 ping 虚拟机 ip。

```
up-tech:~ #ifconfig eth0 192.168.150.199
up-tech:~ #ping 192.168.150.94
PING 192.168.150.94 (192.168.150.94): 56 data bytes
64 bytes from 192.168.150.94: seq=0 ttl=64 time=3.692 ms
64 bytes from 192.168.150.94: seq=1 ttl=64 time=0.685 ms
64 bytes from 192.168.150.94: seq=2 ttl=64 time=0.755 ms

--- 192.168.150.94 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.685/1.710/3.692 ms
up-tech:~ #_
```

e. 若 ping 成功，通过 mount 命令设置 nfs 到虚拟机 UP-CUP2440 文件夹的映射。

- f. 设置成功后进入 nfs 目录，使用 ls 查看内容，若存在 CrossTools 等文件夹则利用 NFS 挂载实验目录成功。

```
up-tech:~ #cd /mnt/nfs
up-tech:/mnt/nfs #ls
CrossTools  IMG  TOOLS  readme.txt  tftpd32.exe
DOC  SRC  install.sh  root.cramfs
up-tech:/mnt/nfs #
```

5) 编译源程序

- a. 进入实验文件夹，使用 vi 编辑并保存 hello.c 文件以及 Makefile 文件

```
[root@localhost hello]# vi hello.c
[root@localhost hello]# ls
hello.c
44 [root@localhost hello]# vi Makefile
```

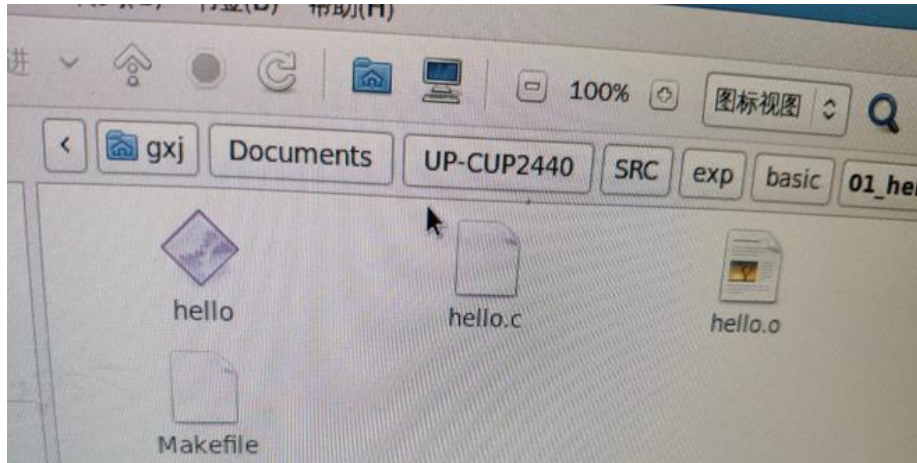
```
gxj@localhost:/home/gxj/Documents/UP-CUP24
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
#include <stdio.h>
main()
{
44 printf("hello world \n");
}
```

```
gxj@localhost:/home/gxj/Documents/UP-CUP24
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
TOPDIR = ../
include $(TOPDIR)Rules.mak
EXEC = hello
OBS = hello.o
all: $(EXEC)
$(EXEC): $(OBS)
$(CC) $(LD_FLAGS) -o $@ $(OBS)
install:
$(EXP_INSTALL) $(EXEC) $(INSTALL_DIR)
clean:
-rm -f $(EXEC) *.elf *.gdb *.o
```

- b. 使用 make 命令进行编译

```
[root@localhost 01_hello]# make
make: Warning: File '../Rules.mak' has modification time 288301375 s in the future
arm-linux-gcc -c -o hello.o hello.c
arm-linux-gcc -o /UP-CUP2440/SRC/bin/hello hello.o
arm-linux-gcc -o hello hello.o
make: 警告：检测到时钟错误。您的创建可能是不完整的。
[root@localhost 01_hello]#
```

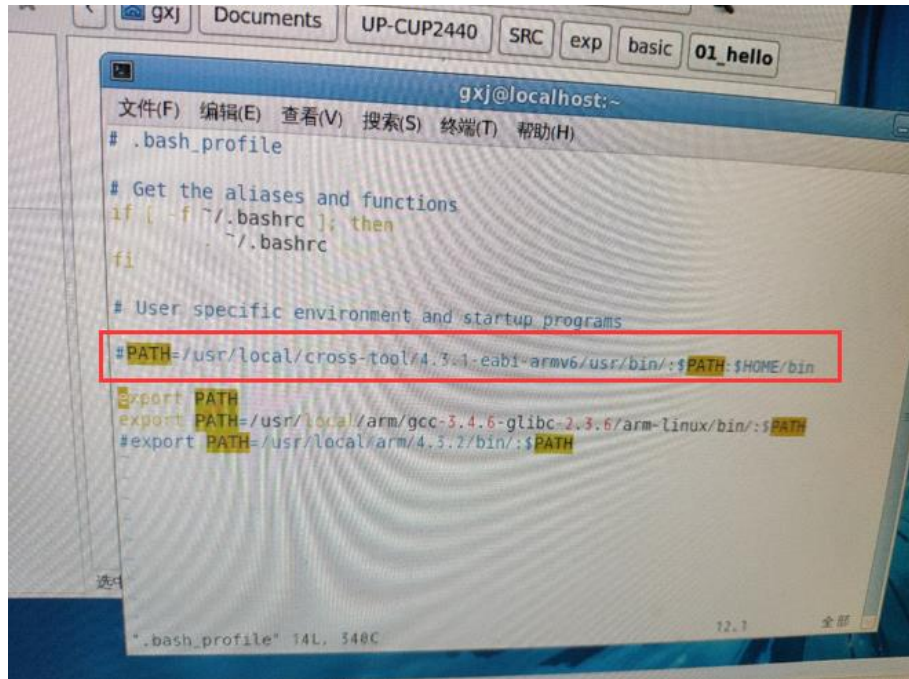
c. 生成文件



d. 在超级终端中尝试运行编译后 hello 文件，但运行失败，报错如下

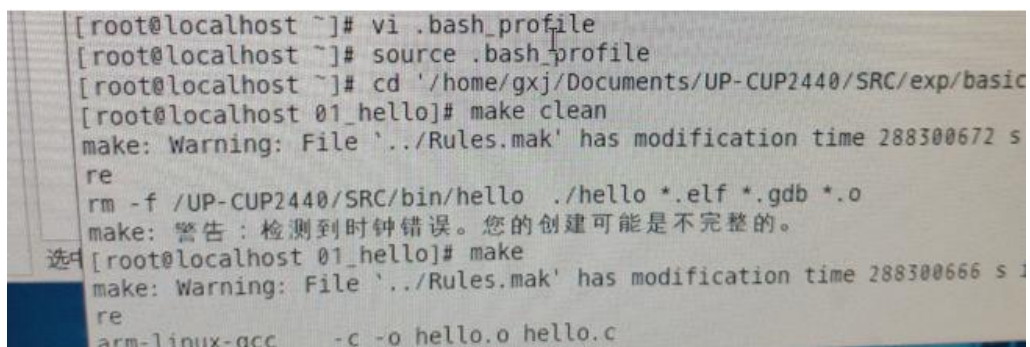
```
CO#2440 - 超级终端
up-tech:/mnt/nfs/SRC/exp/basic #ls
01_hello 03_ad 05_httpd 07_dcmotor 09_int Rules.mak
02_pthread 04_da 06_485 08_led 10_cancel
up-tech:/mnt/nfs/SRC/exp/basic #cd 01_hello/
up-tech:/mnt/nfs/SRC/exp/basic/01_hello #ls
Makefile hello hello.c hello.o
up-tech:/mnt/nfs/SRC/exp/basic/01_hello #./hello
./hello: /lib/libc.so.6: version 'GLIBC_2.4' not found (required by ./hello)
up-tech:/mnt/nfs/SRC/exp/basic/01_hello #./hello
./hello: /lib/libc.so.6: version 'GLIBC_2.4' not found (required by ./hello)
up-tech:/mnt/nfs/SRC/exp/basic/01_hello #./hello
./hello: /lib/libc.so.6: version 'GLIBC_2.4' not found (required by ./hello)
up-tech:/mnt/nfs/SRC/exp/basic/01_hello #./hello
./hello: /lib/libc.so.6: version 'GLIBC_2.4' not found (required by ./hello)
```

e. 解决方案如下，使用 vi 修改 .bash_profile 文件，将 export 前的一行 “PATH=...” 注释后，保存并退出



```
gxj@localhost:~  
# .bash_profile  
  
# Get the aliases and functions  
if [ -f ~/.bashrc ]; then  
    . ~/.bashrc  
fi  
  
# User specific environment and startup programs  
#PATH=/usr/local/cross-tool/4.3.1-eabi-armv6/usr/bin/:$PATH:$HOME/bin  
export PATH  
export PATH=/usr/local/arm/gcc-3.4.6-glibc-2.3.6/arm-linux/bin/:$PATH  
#export PATH=/usr/local/arm/4.3.2/bin/:$PATH
```

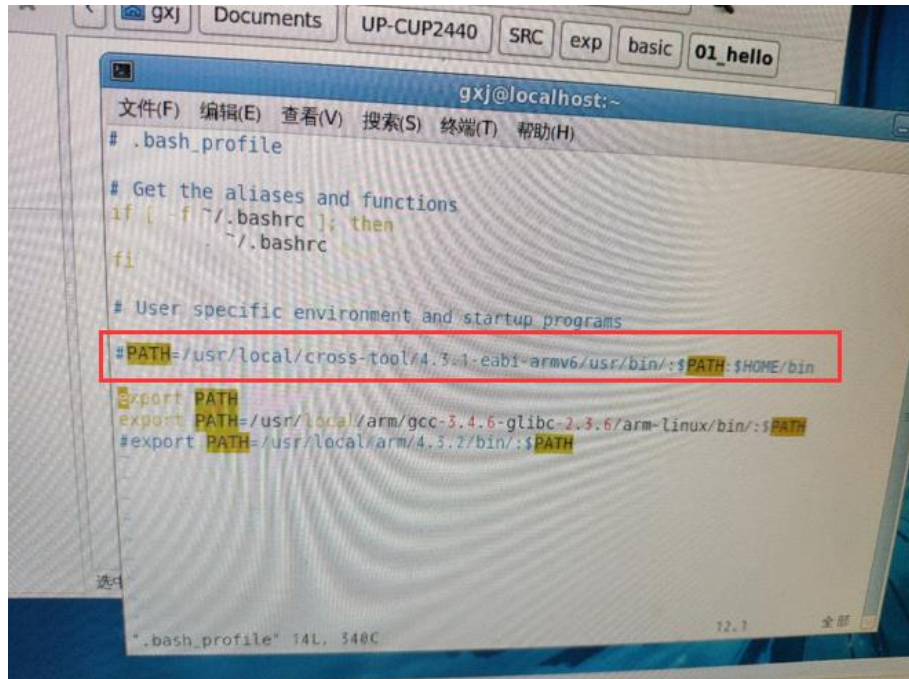
f. 使用 make clean 后，重新使用 make 编译



```
[root@localhost ~]# vi .bash_profile  
[root@localhost ~]# source .bash_profile  
[root@localhost ~]# cd '/home/gxj/Documents/UP-CUP2440/SRC/exp/basic'  
[root@localhost 01_hello]# make clean  
make: Warning: File '../Rules.mak' has modification time 288300672 s  
rm -f /UP-CUP2440/SRC/bin/hello ./hello *.elf *.gdb *.o  
make: 警告：检测到时钟错误。您的创建可能是不完整的。  
[root@localhost 01_hello]# make  
make: Warning: File '../Rules.mak' has modification time 288300666 s  
arm-linux-gcc -c -o hello.o hello.c
```

五、实验结果

在超级终端中运行，得到下图显示，运行成功



```
gxj@localhost:~  
# .bash_profile  
  
# Get the aliases and functions  
if [ -f ~/.bashrc ]; then  
    . ~/.bashrc  
fi  
  
# User specific environment and startup programs  
#PATH=/usr/local/cross-tool/4.3.1-eabi-armv6/usr/bin/:$PATH:$HOME/bin  
export PATH  
export PATH=/usr/local/arm/gcc-3.4.6-glibc-2.3.6/arm-linux/bin/:$PATH  
#export PATH=/usr/local/arm/4.3.2/bin/:$PATH
```

使用 make clean 后，重新使用 make 编译

说明：1、每位同学按照规定的编组和实验编号独立完成实验内容，并提交一份实验报告。

2、实验报告页码可增减。