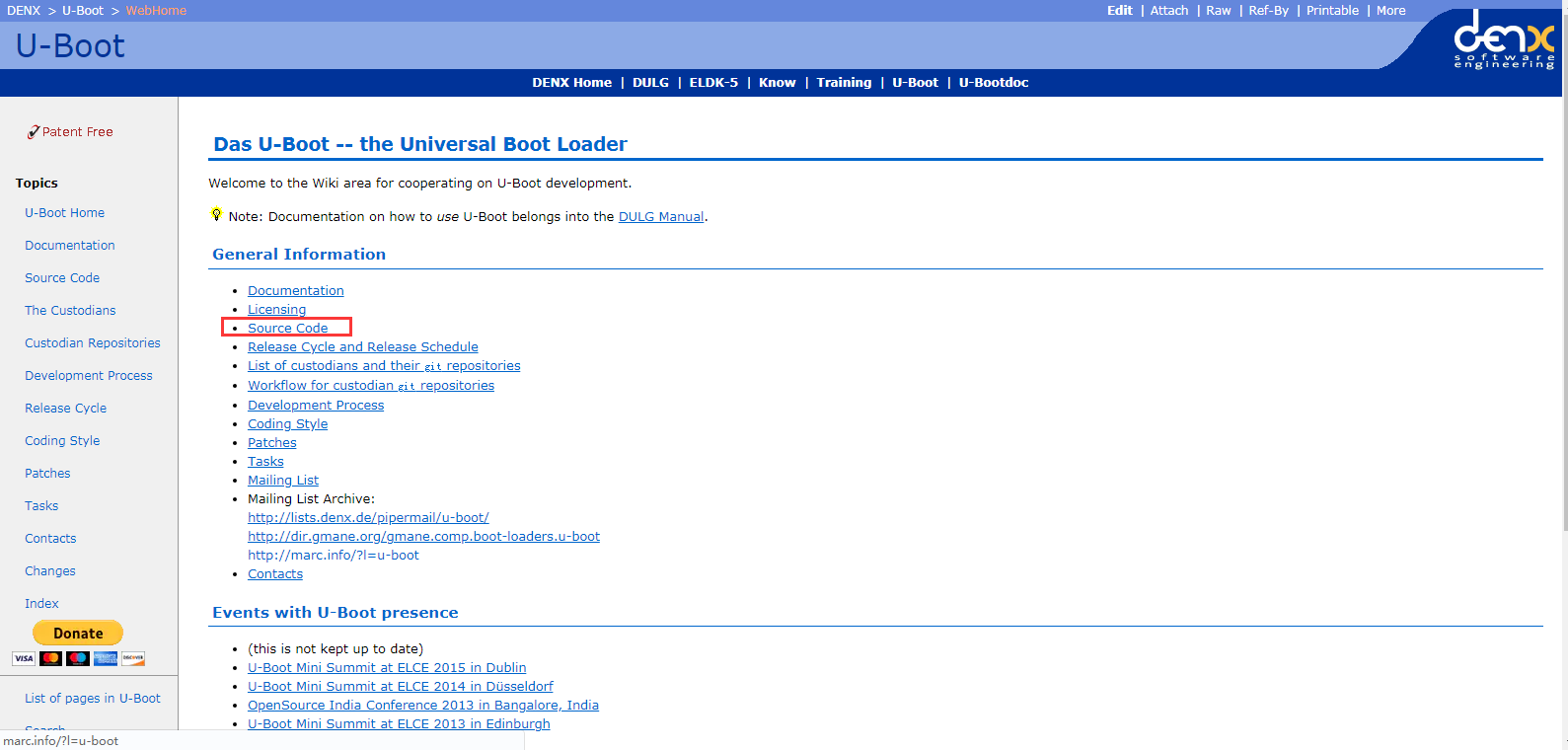
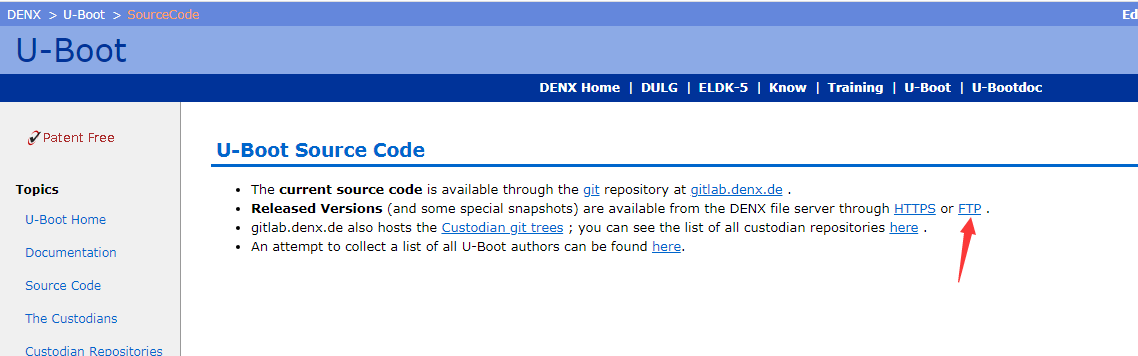
进入uboot官网：http://www.denx.de/wiki/U-Boot/



点击source code，选择ＦＴＰ下载方式



＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊

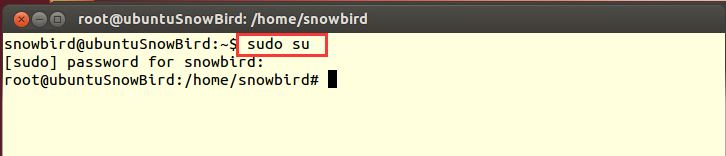
关于bootloader和u-boot讲的最透彻一篇

<http://blog.chinaunix.net/uid-26085226-id-2105173.html>

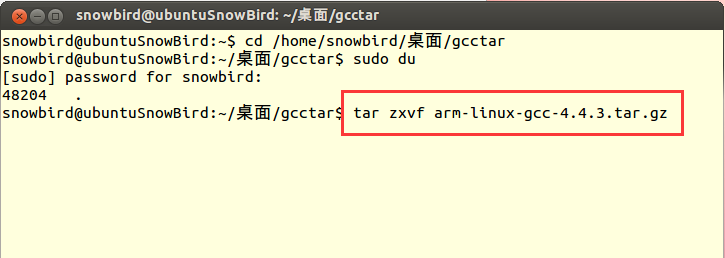
＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊

# 一、Uboot移植前要先进行交叉编译工具gcc的安装

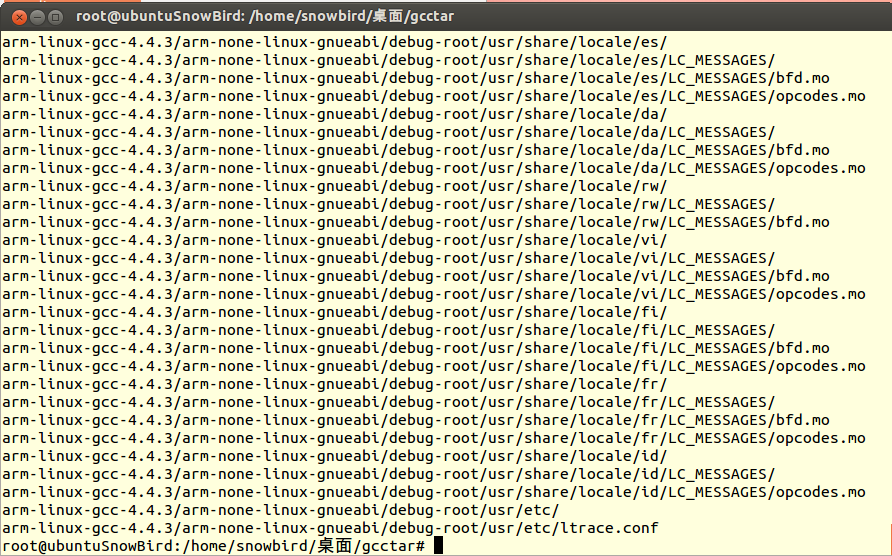
1. 进入管理员模式



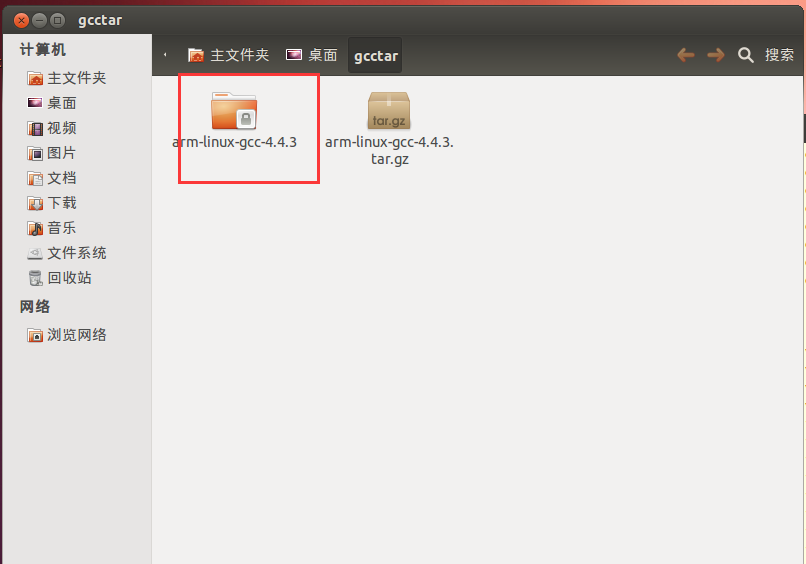
1. 解压gcc安装包



回车后，进行解压，命令框输出如下：



解压出来的文件夹如下：



1. 配置环境变量

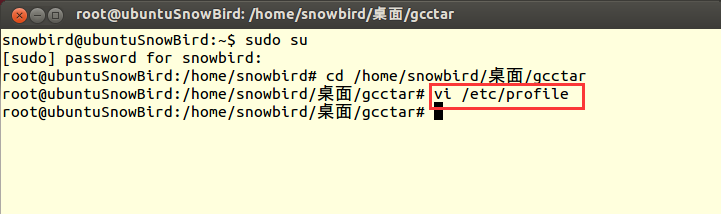
配置环境变量，把交叉编译工具链的路径添加到环境变量PATH中去，这样就可以在任何目录下使用这些工具。记下上一步中的安装路径，使用命令：

vi /etc/profile编辑profile文件；

添加环境变量，在profile中最后一行添加

export PATH=$PATH:/usr/local/arm-linux-gcc-4.4.3/bin

输入：vi /etc/profile



回车，在最下面添加：export PATH=$PATH:/usr/local/arm-linux-gcc-4.4.3/bin，然后按“esc”，输入“:wq”回车退出vi编辑

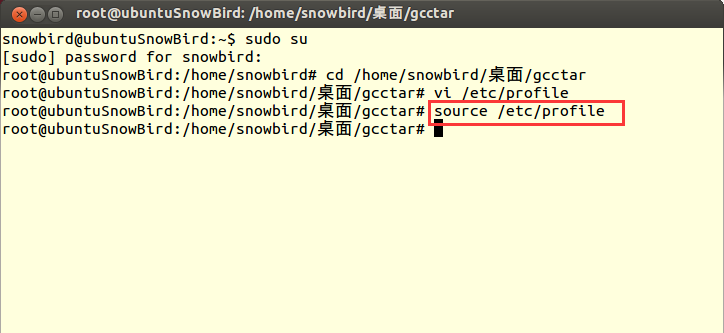




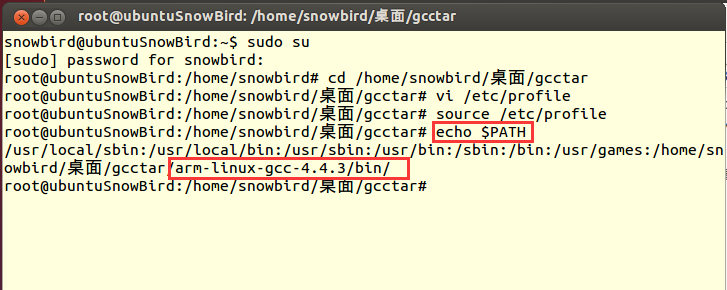
使环境变量生效并验证，

然后使用命令：source /etc/profile

使环境变量生效（在不同用户下最好都运行一下此语句，例如shgao和root用户下都运行一下）。



检查是否将路径加入到PATH：echo $PATH：



显示的内容中有/usr/local/arm-linux-gcc-4.4.3/bin，说明已经将交叉编译器的路径加入PAT

利用arm-linux-gcc -v查看环境是否搭建成功：

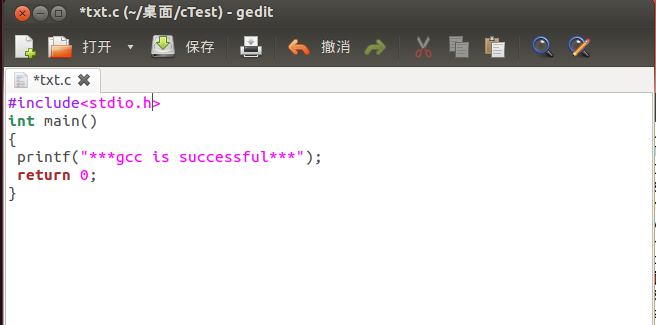


交叉编译环境搭建成功！

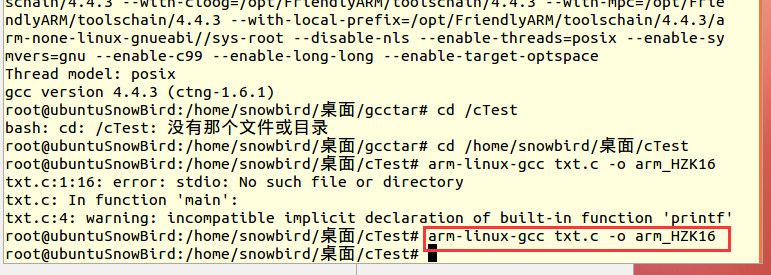
可以看出环境已经搭建成功

编译生成二进制文件

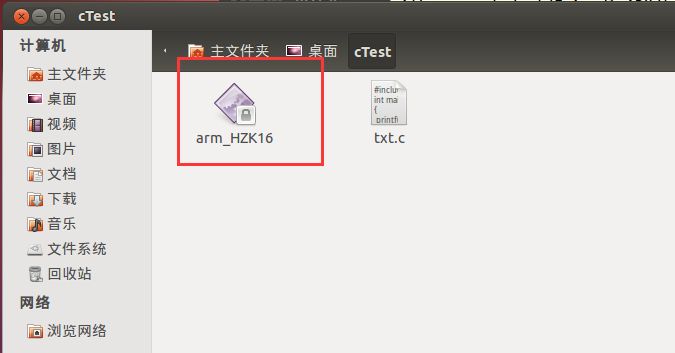
新建一个C语言文件夹：



编译生成可以在arm上运行的二进制文件：



回车后，可以看到生成了arm\_HZK16文件，arm\_HZK16文件为arm上运行的二进制文件，是一个可执行文件。



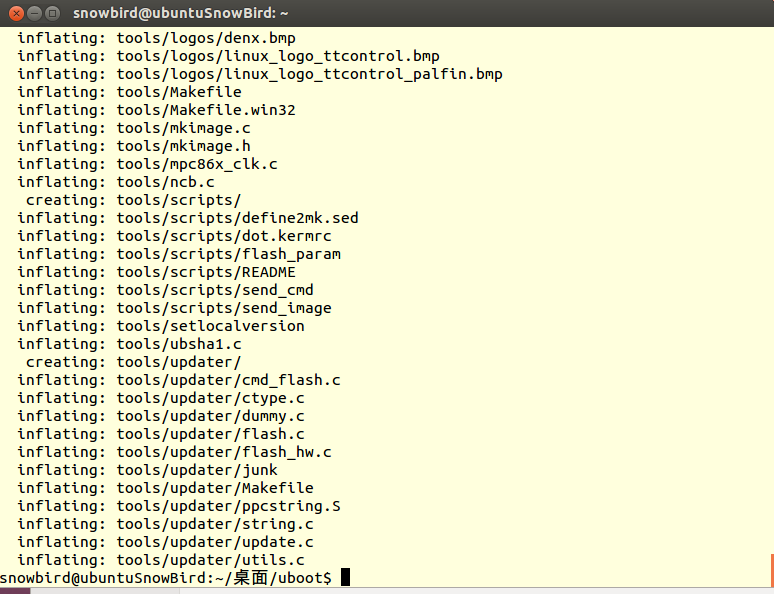
到这里已经完成了Ubuntu系统下arm-linux-gcc交叉编译环境搭建过程。

# 二、移植Uboot

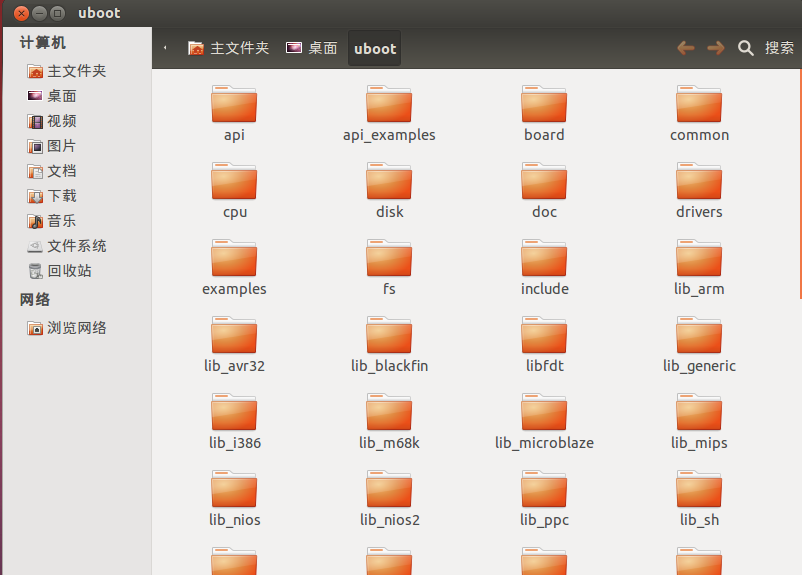
https://blog.csdn.net/qq\_16777851/article/details/81543373

解压；

输入命令：unzip u-boot-1.3.4.zip 进行解压



解压完成后显示如下的目录：

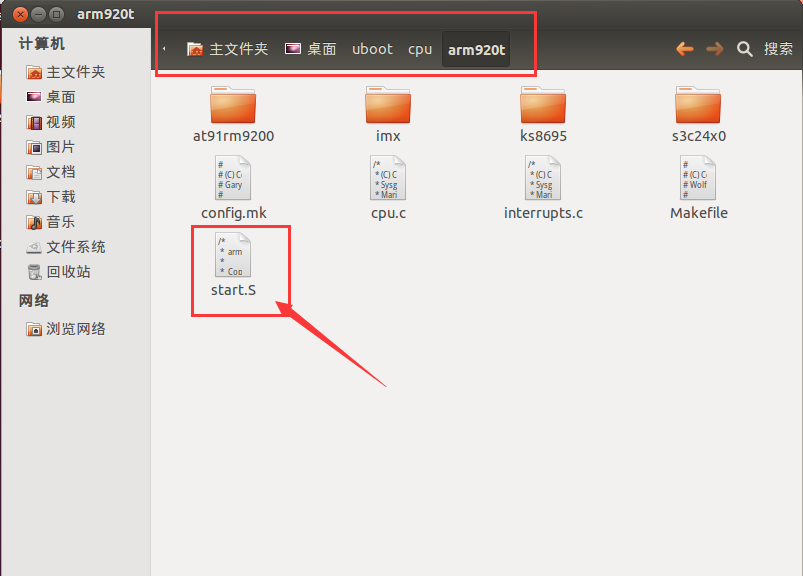


1、首先下载u-boot的源码包，然后解压，进入源码包目录。

2、修改源码让u-boot支持从Nand Flash启动。

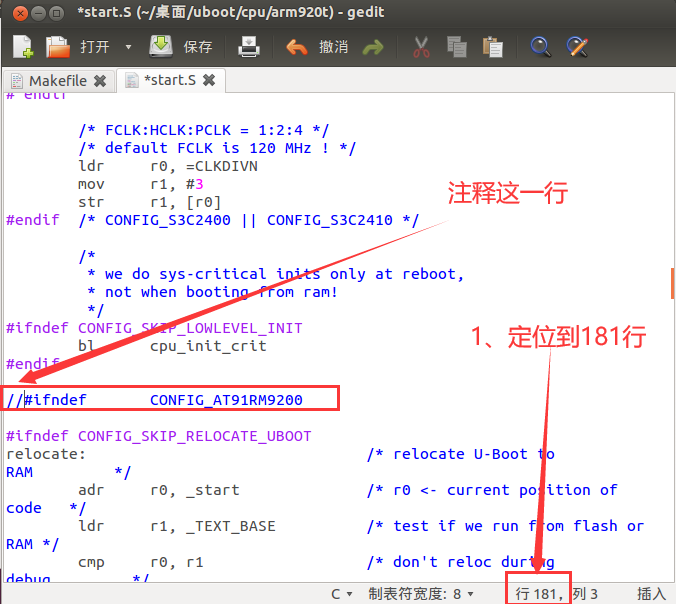
**具体如下：**

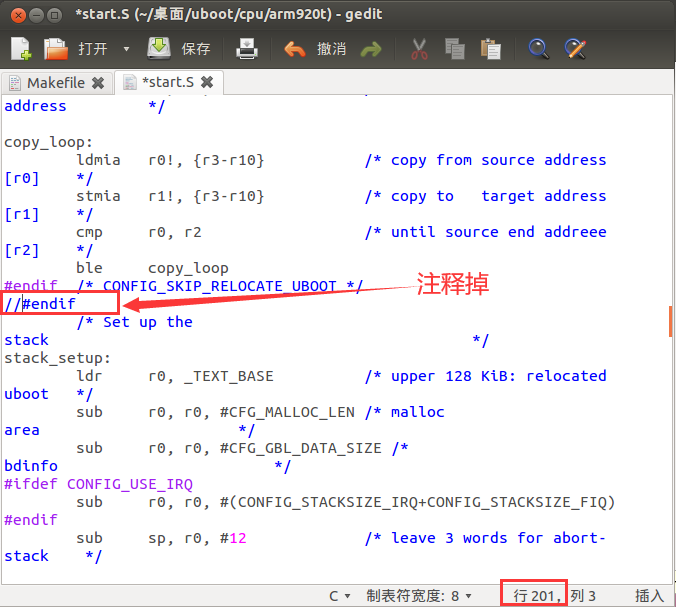
1、修改“uboot/cpu/arm920t/start.S”文件，执行以下命令：



打开该文件，

（1）181行和201行前加上 // 注释掉，



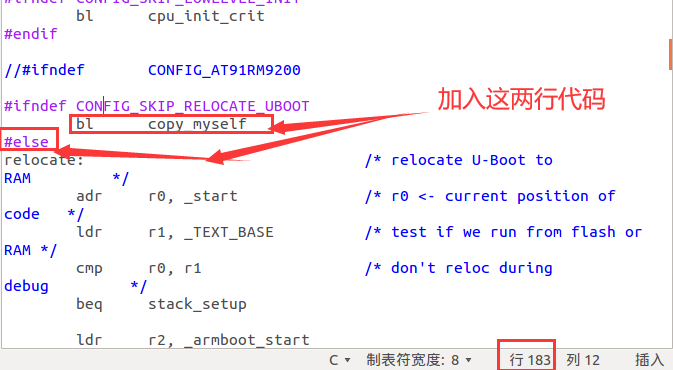


（2）找到183行，在#ifndef CONFIG\_SKIP\_RELOCATE\_UBOOT之后加入：

#ifdef CONFIG\_S3C2410\_NAND\_BOOT

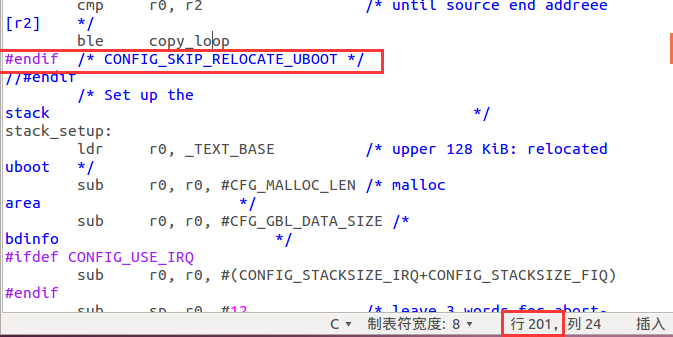
bl copy\_myself

#else



（3）“ble copy\_loop”语句之后加入：

#endif



（4）“\_start\_armboot: .word start\_armboot”语句之后加入copy\_loop子程序：

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* copy u-boot to ram

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

#ifdef CONFIG\_S3C2410\_NAND\_BOOT

copy\_myself:

mov r10, lr @save return address to r10

ldr sp, DW\_STACK\_START

mov fp, #0

bl NF\_Init

ldr r0, =UBOOT\_RAM\_BASE

mov r1, #0x0

mov r2, #0x30000

bl nand\_read\_whole

tst r0, #0x0

beq ok\_nand\_read

1: b 1b

ok\_nand\_read:

mov r0, #0x00000000

ldr r1, =UBOOT\_RAM\_BASE

mov r2, #0x400

go\_next:

ldr r3, [r0], #4

ldr r4, [r1], #4

teq r3, r4

bne notmatch

subs r2, r2, #4

beq done\_nand\_read

bne go\_next

notmatch:

1: b 1b

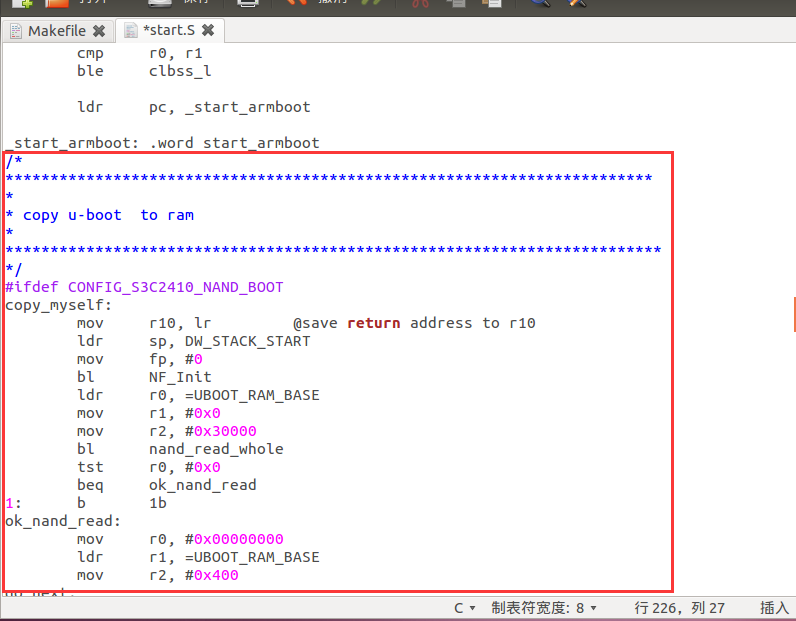
done\_nand\_read:

mov pc, r10

#endif

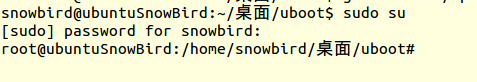
DW\_STACK\_START:

.word STACK\_BASE+STACK\_SIZE-4



2、新建“u-boot-1.3.2/ board/smdk2410/nand.c”文件

进入管理员模式



在nand.c文件里添加如下内容：

#include <common.h>

#include <s3c2410.h>

#include <config.h>

#define TACLS 0

#define TWRPH0 3

#define TWRPH1 0

#define U32 unsigned int

extern unsigned long nand\_probe(unsigned long physadr);

static void NF\_Reset(void)

{

int i;

NF\_nFCE\_L();

NF\_CMD(0xFF);

for(i=0;i<10;i++);

NF\_WAITRB();

NF\_nFCE\_H();

}

void NF\_Init(void)

{

rNFCONF=(1<<15)|(1<<14)|(1<<13)|(1<<12)|(1<<11)|(TACLS<<8)|(TWRPH0<<4)|(TWRPH1<<0);

NF\_Reset();

}

int nand\_read\_whole(unsigned char \*buf, unsigned long start\_addr, int size)

{

int i, j;

if((start\_addr & NAND\_BLOCK\_MASK) || (size & NAND\_BLOCK\_MASK))

return 1;

NF\_nFCE\_L();

for(i=0; i<10; i++);

i = start\_addr;

while(i < start\_addr + size) {

rNFCMD = 0;

rNFADDR = i & 0xff;

rNFADDR = (i >> 9) & 0xff;

rNFADDR = (i >> 17) & 0xff;

rNFADDR = (i >> 25) & 0xff;

NF\_WAITRB();

for(j=0; j < NAND\_SECTOR\_SIZE; j++, i++) {

\*buf = (rNFDATA & 0xff);

buf++;

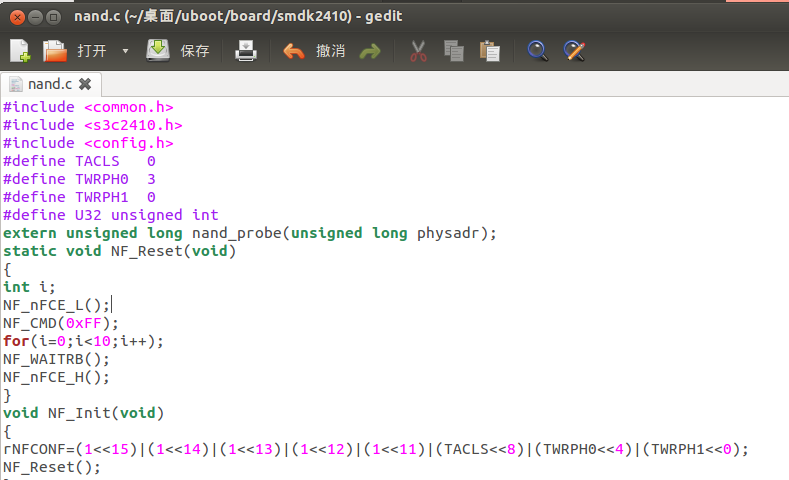
}

}

NF\_nFCE\_H();

return 0;

}



保存退出

3、修改“u-boot-1.3.2/board/smdk2410/Makefile”文件

找到该文件夹，然后将“COBJS:= smdk2410.o flash.o”改为“COBJS:=smdk2410.o flash.o nand.o”

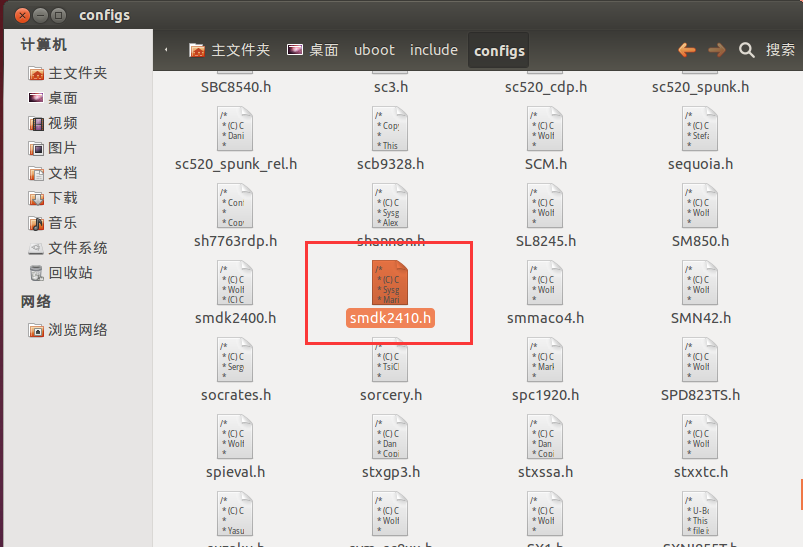


即改完后：



4、修改“uboot/include/configs/smdk2410.h”文件

找到文件“uboot/include/configs/smdk2410.h”：



打开smdk2410.h，在该文件的最后一个#endif的前面添加如下内容：

#define CONFIG\_S3C2410\_NAND\_BOOT 1

#define STACK\_BASE 0x33f00000

#define STACK\_SIZE 0x8000

#define UBOOT\_RAM\_BASE 0x33f80000

#define CFG\_NAND\_BASE 0x4E000000

#define CFG\_MAX\_NAND\_DEVICE 1

#define SECTORSIZE 512

#define NAND\_SECTOR\_SIZE SECTORSIZE

#define NAND\_BLOCK\_MASK (NAND\_SECTOR\_SIZE - 1)

#define ADDR\_COLUMN 1

#define ADDR\_PAGE 2

#define ADDR\_COLUMN\_PAGE 3

#define NAND\_ChipID\_UNKNOWN 0x00

#define NAND\_MAX\_FLOORS 1

#define NAND\_MAX\_CHIPS 1

#define WRITE\_NAND\_COMMAND(d, adr) do {rNFCMD = d;} while(0)

#define WRITE\_NAND\_ADDRESS(d, adr) do {rNFADDR = d;} while(0)

#define WRITE\_NAND(d, adr) do {rNFDATA = d;} while(0)

#define READ\_NAND(adr) (rNFDATA)

#define NAND\_WAIT\_READY(nand) {while(!(rNFSTAT&(1<<0)));}

#define NAND\_DISABLE\_CE(nand) {rNFCONF |= (1<<11);}

#define NAND\_ENABLE\_CE(nand) {rNFCONF &= ~(1<<11);}

#define NAND\_CTL\_CLRALE(nandptr)

#define NAND\_CTL\_SETALE(nandptr)

#define NAND\_CTL\_CLRCLE(nandptr)

#define NAND\_CTL\_SETCLE(nandptr)

#define CONFIG\_MTD\_NAND\_VERIFY\_WRITE 1

#define rNFCONF (\*(volatile unsigned int \*)0x4e000000)

#define rNFCMD (\*(volatile unsigned char \*)0x4e000004)

#define rNFADDR (\*(volatile unsigned char \*)0x4e000008)

#define rNFDATA (\*(volatile unsigned char \*)0x4e00000c)

#define rNFSTAT (\*(volatile unsigned int \*)0x4e000010)

#define rNFECC (\*(volatile unsigned int \*)0x4e000014)

#define rNFECC0 (\*(volatile unsigned char \*)0x4e000014)

#define rNFECC1 (\*(volatile unsigned char \*)0x4e000015)

#define rNFECC2 (\*(volatile unsigned char \*)0x4e000016)

#define NF\_CMD(cmd) {rNFCMD=cmd;}

#define NF\_ADDR(addr) {rNFADDR=addr;}

#define NF\_nFCE\_L() {rNFCONF&=~(1<<11);}

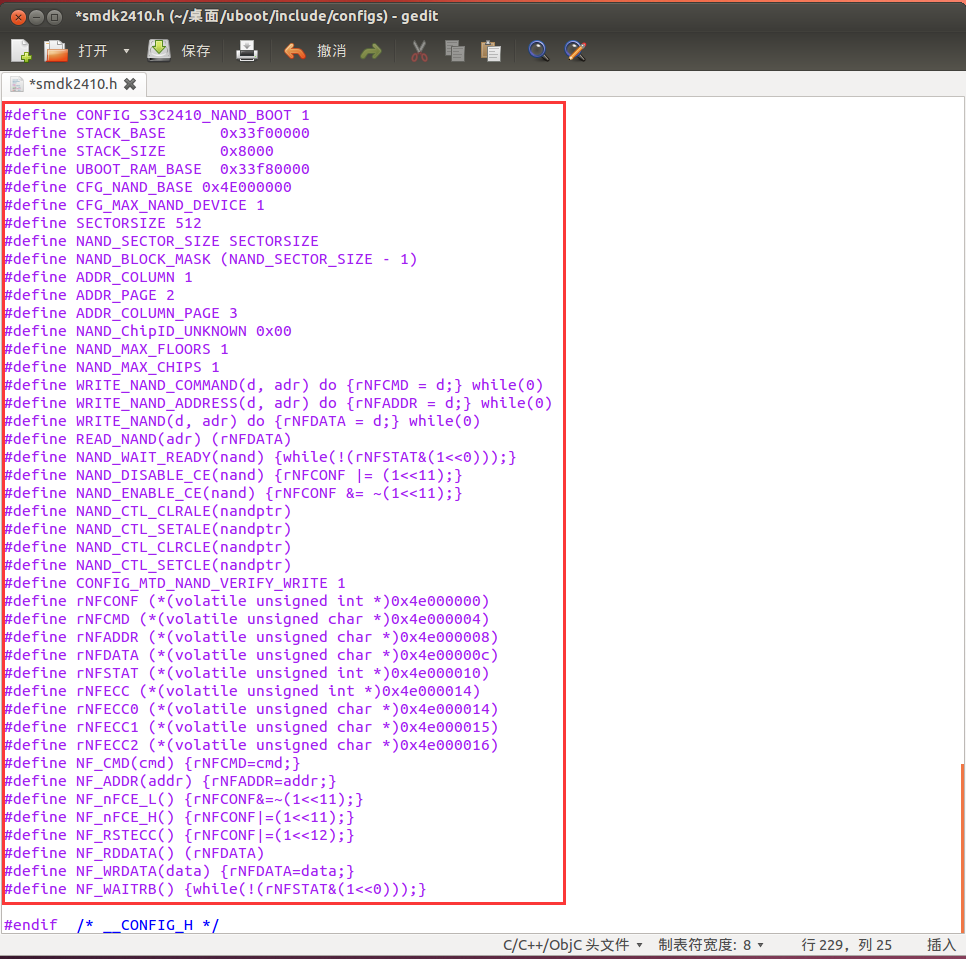
#define NF\_nFCE\_H() {rNFCONF|=(1<<11);}

#define NF\_RSTECC() {rNFCONF|=(1<<12);}

#define NF\_RDDATA() (rNFDATA)

#define NF\_WRDATA(data) {rNFDATA=data;}

#define NF\_WAITRB() {while(!(rNFSTAT&(1<<0)));}



5、使u-boot支持dm9000网卡

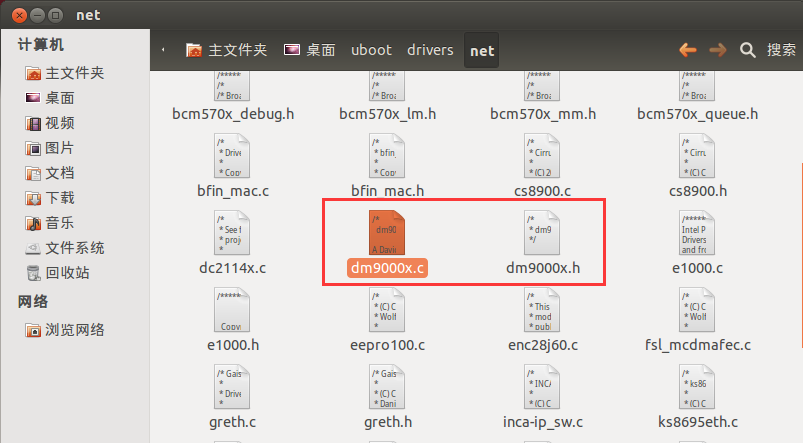
步骤如下：

1. 将dm9000x.h和dm9000x.c两个文件拷贝到drivers/net/下。

这两个的文件路径如下：

dm9000x.h：uboot/drivers/net/dm9000x.h

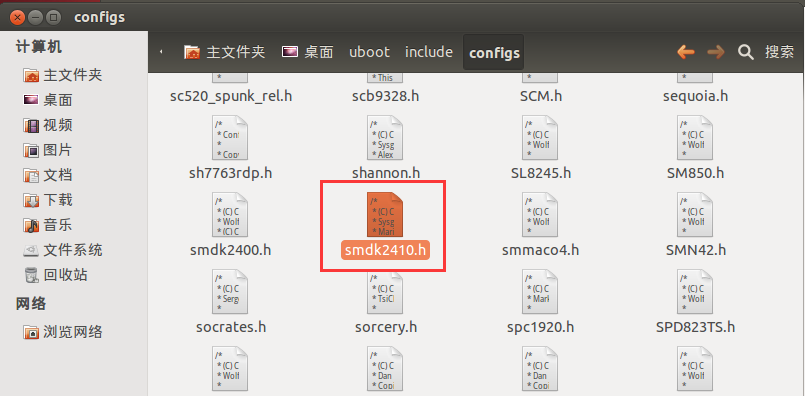
dm9000x.c：uboot/drivers/net/dm9000x.c



发现两个文件已经在drivers/net/下了。

6、 修改“uboot/include/configs/smdk2410.h”文件

找到文件



打开：

（1）将原网卡“CS8900”改为“DM9000”

（2）添加网络命令

（3）修改IP地址

（4）修改命令提示符

（5）修改默认下载地址

（6）修改环境变量在Nand Flash中的存储地址

**具体如下：**

（1）将原网卡“CS8900”改为“DM9000”

注释掉：

// #define CONFIG\_DRIVER\_CS8900 1 /\* we have a CS8900 on-board \*/

// #define CS8900\_BASE 0x19000300

// #define CS8900\_BUS16 1 /\* the Linux driver does accesses as shorts \*/

这3行之后添加：

#define CONFIG\_DRIVER\_DM9000 1

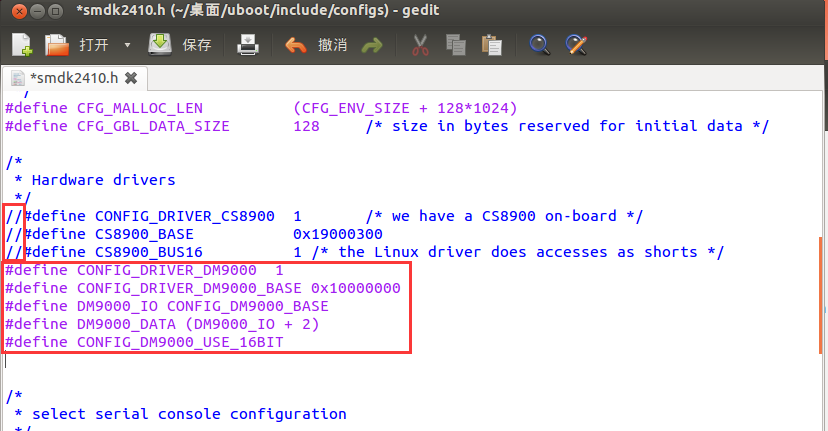
#define CONFIG\_DRIVER\_DM9000\_BASE 0x10000000

#define DM9000\_IO CONFIG\_DM9000\_BASE

#define DM9000\_DATA (DM9000\_IO + 2)

#define CONFIG\_DM9000\_USE\_16BIT

如下图



（2）添加网络命令

找到下面的几行：

#define CONFIG\_CMD\_CACHE

#define CONFIG\_CMD\_DATE

#define CONFIG\_CMD\_ELF

在其下面添加下面几行：

#define CONFIG\_CMD\_REGINFO

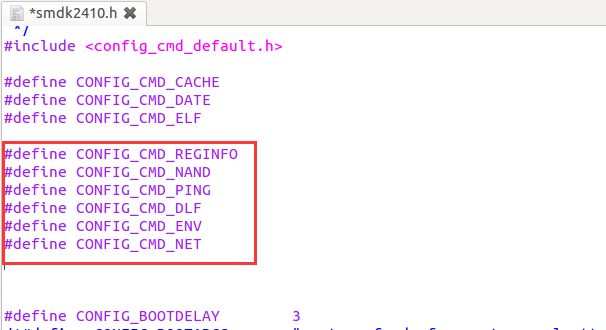
#define CONFIG\_CMD\_NAND

#define CONFIG\_CMD\_PING

#define CONFIG\_CMD\_DLF

#define CONFIG\_CMD\_ENV

#define CONFIG\_CMD\_NET



（3）修改IP地址

找到下面几行：

#define CONFIG\_BOOTDELAY 3

/\*#define CONFIG\_BOOTARGS "root=ramfs devfs=mount console=ttySA0,9600" \*/

/\*#define CONFIG\_ETHADDR 08:00:3e:26:0a:5b \*/

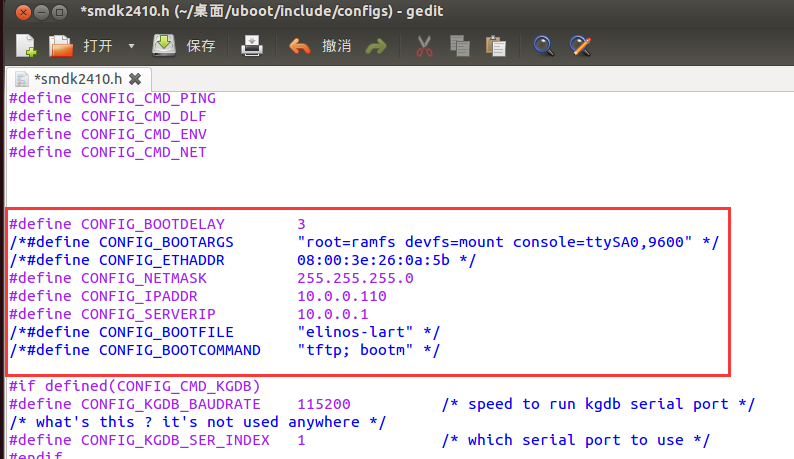
#define CONFIG\_NETMASK 255.255.255.0

#define CONFIG\_IPADDR 10.0.0.110

#define CONFIG\_SERVERIP 10.0.0.1

/\*#define CONFIG\_BOOTFILE "elinos-lart" \*/

/\*#define CONFIG\_BOOTCOMMAND "tftp; bootm" \*/



修改为：

#define CONFIG\_BOOTDELAY 3

#define CONFIG\_BOOTARGS "root=/dev/mtdblock3 init=/linuxrc console=ttySAC0,115200"

#define CONFIG\_ETHADDR 08:00:3e:26:0a:5b

#define CONFIG\_NETMASK 255.255.255.0

#define CONFIG\_IPADDR 192.168.1.13

#define CONFIG\_SERVERIP 192.168.1.12

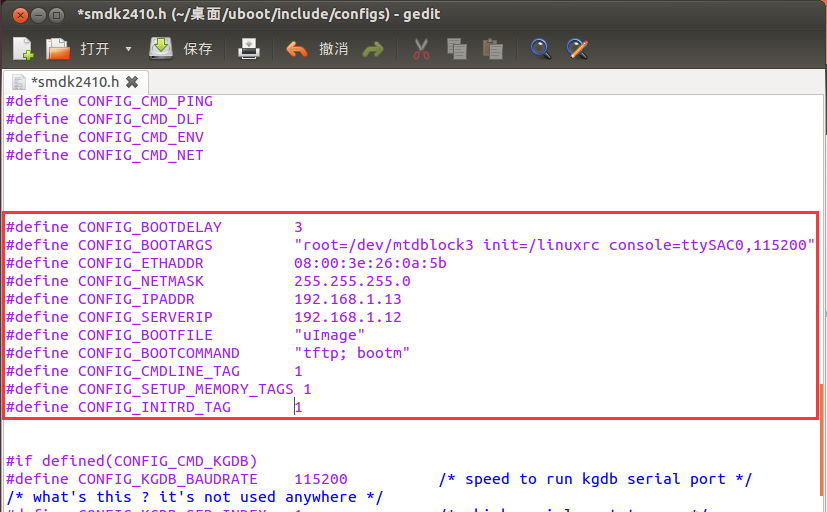
#define CONFIG\_BOOTFILE "uImage"

#define CONFIG\_BOOTCOMMAND "tftp; bootm"

#define CONFIG\_CMDLINE\_TAG 1

#define CONFIG\_SETUP\_MEMORY\_TAGS 1

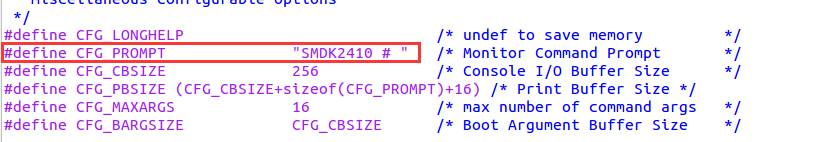
#define CONFIG\_INITRD\_TAG 1



（4）修改命令提示符

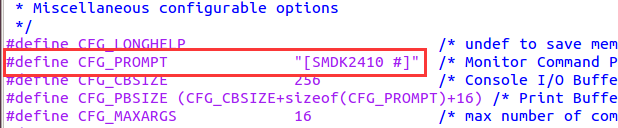
找到下面一行：

#define CFG\_PROMPT "SMDK2410 # "



修改为：

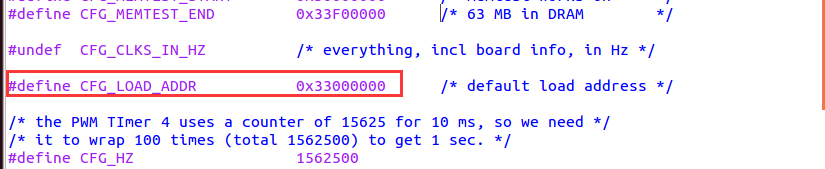
#define CFG\_PROMPT "[ smdk2410 #]"



（5）修改默认下载地址

找到下面一行：

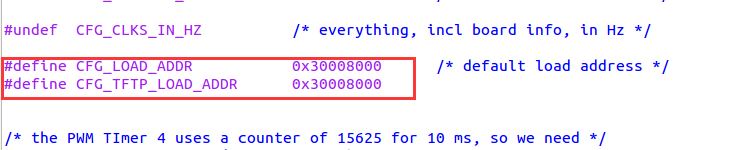
#define CFG\_LOAD\_ADDR 0x33000000



修改为：

#define CFG\_LOAD\_ADDR 0x30008000

#define CFG\_TFTP\_LOAD\_ADDR 0x30008000

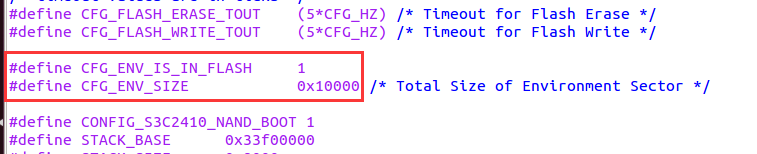


（6）修改环境变量在Nand Flash中的存储地址

找到下面两行：

#define CFG\_ENV\_IS\_IN\_FLASH 1

#define CFG\_ENV\_SIZE 0x10000

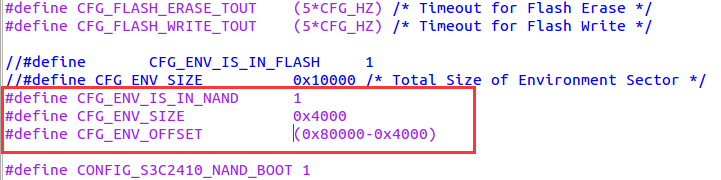


修改为：

#define CFG\_ENV\_IS\_IN\_NAND 1

#define CFG\_ENV\_SIZE 0x4000

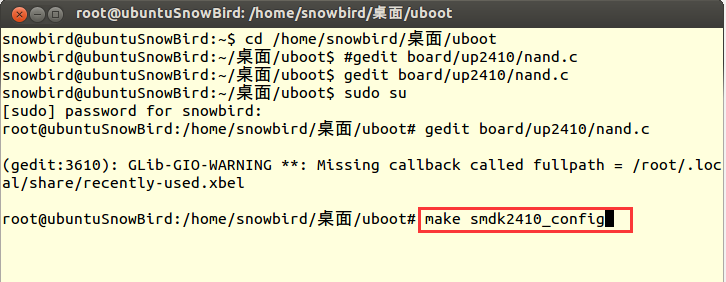
#define CFG\_ENV\_OFFSET (0x80000-0x4000)



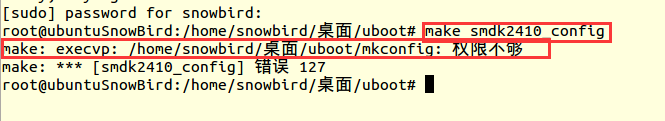
7、编译生成u-boot

依次执行以下命令：

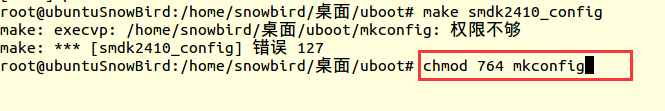
#make smdk2410\_config



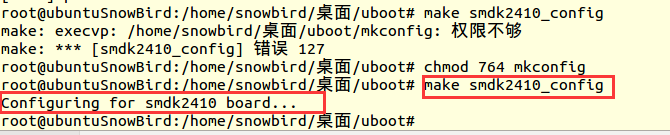
发现提示权限不够，



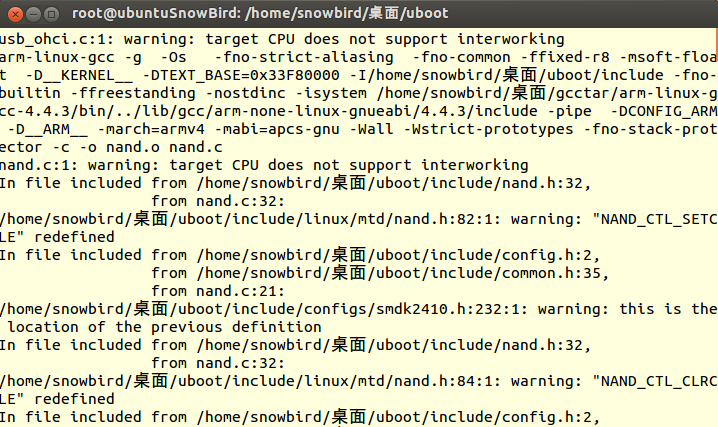
加上权限，



然后再执行make smdk2410\_config，发现可成功执行



#make



编译完成之后就生成了u-boot.bin文件，这个文件就是我们要烧写到开发板上的文件。

# 三、内核移植

1、

# 四、根文件系统移植

# 五、串行端口程序设计与移植