

实验报告

课程：嵌入式系统A

第四次实验

**姓 名 凌智城**

**学 号 201806061211**

**专业班级 通信工程1803班**

**老 师 黄国兴**

**学 院 信息工程学院**

**提交日期**  **2021年5月26日**

实验12：mount挂载实验

重新提交第三次实验的挂载实验，上次实验后半部分没做完

1. 实验目的
2. 掌握配置NFS服务的方法。
3. 掌握mount挂在usb/sd的方法。
4. 实验内容
5. 配置NFS服务。
6. mount挂在usb/sd设备。
7. 实验步骤

**步骤1：**连接设备

打开PC机上的串口调试工具，开启实验箱上的电源，按空格键进入实验箱上的板上Linux系统。

若是连接实验室的三台服务器需要先SSH连接上服务器，若使用本地虚拟机则打开虚拟机bash即可。

接着查看串口号，通过 putty 软件使用串口通信方式连接实验箱，如图12-1所示所示：选择 putty 串口连接试验箱如图12-2所示：



图12-1 查看串口端号

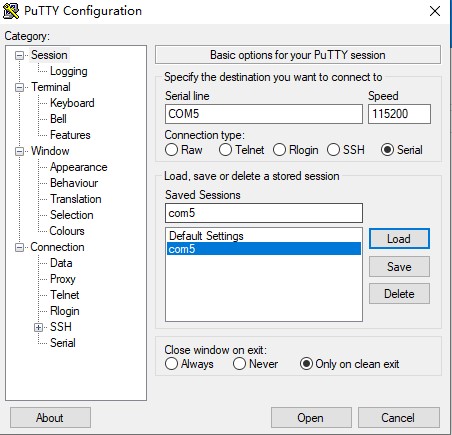


图12-2 putty串口连接配置

输入启动参数，接着启动内核，如图12-3所示：

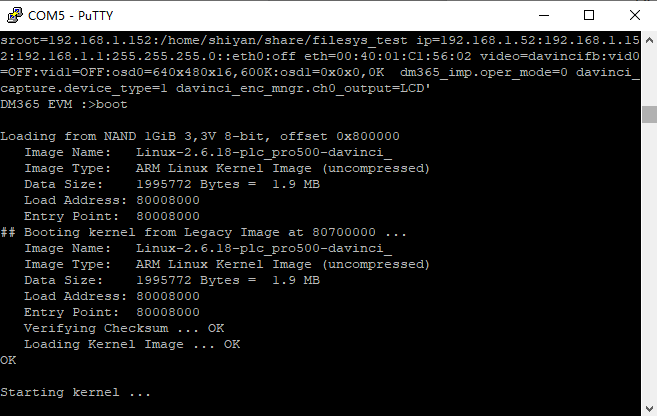


图12-3 输入启动参数

setenv bootargs 'mem=110M console=ttyS0,115200n8 root=/dev/nfs rw nfsroot=192.168.1.152:/home/shiyan/share/filesys\_test ip=192.168.1.52:192.168.1.152:192.168.1.1:255.255.255.0::eth0:off eth=00:40:01:C1:56:02 video=davincifb:vid0=OFF:vid1=OFF:osd0=640x480x16,600K:osd1=0x0x0,0K dm365\_imp.oper\_mode=0 davinci\_capture.device\_type=1 davinci\_enc\_mngr.ch0\_output=LCD'

使用实验室服务器的和使用虚拟机的配置有所不同，使用虚拟机的先用ifconfig命令在inet addr处查看IP，根据试验箱的MAC地址更改数据。

输入用户名root登录实验箱如图12-4所示

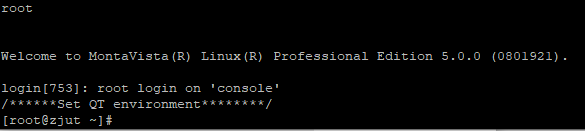


图12-4 登录实验箱

**步骤2：**配置nfs服务器设置（使用服务器用putty连接服务器，使用虚拟机则用VMware进入Ubuntu系统）

1. 进入Linux服务器系统的/etc目录，命令如下：  
   shiyan@ubuntu:/$ cd /etc/

shiyan@ubuntu:/etc$

1. 编辑/etc/exports 的文件，sudo 命令是进入 root 权限，这里需要输入登录密码，命令如下：

shiyan@ubuntu:/etc$ sudo vi exports

[sudo] password for shiyan:

进入如下所示exports文件，再exports文件中添加一行：

/home/挂载目录 192.168.\*.\*(rw,sync,no\_root\_squash)

（即在文件底部添加，保存退出，192.168.\*.\*根据服务器IP和网关确定）

至此已完成NFS服务器配置，接下来启动NFS服务：

启动 NFS 的命令如下：

$ sudo ./etc/rc.d/init.d/nfs start

如果之前已启动 NFS，更改后可用以下命令：

$ sudo /etc/init.d/nfs-kernel-server restart

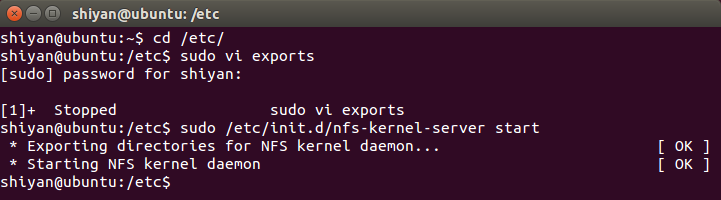


图12-5 启动NFS

**步骤3：**文件夹挂载

1. 挂载

服务器端的NFS服务配置完成以后启动实验板，在串口调试工具中开始挂在文件夹，在mount之前必须先配置。加上ifconfig eth0 192.168.1.\*\*\*命令，修改实验板IP。（上述IP为实验箱的具体IP，注意要和被挂载的服务器处在同一网段。）

mount过程如下：（实验箱上进行）

[root]# mount -t nfs -o nolock 192.168.1.152:/home/shiyan/share/filesys\_test /mnt/mtd/

[root]#

验证挂载是否成功，输入df命令查看，结果增加一行出现：

[root]#df

Filesystem 1K-blocks Used Availabled Use% Mounted on

192.168.1.\*\*\* ：/home/shiyan/share/ 193241632 102773502 80652000 56% /mnt/mtd

从上可以看出已经将服务器上的/home/shiyan/share/nfs （192.168.1.\*\*\*:/home/shiyan/shiare/nfs）目录挂载到了实验箱文件系统的/mnt/mtd 目 录下。也就是说此时实验箱可以通过/mnt/mtd 目录直接访问服务器上的/home/shiyan/share/nfs目录。可以在服务器端进入/home/shiyan/share/nfs目录和在实验箱中进入/mnt/mtd 目录对比里面的内容，可以发现内容是一样的，并且在任意端向目录中创建新文件，在另一端均可见。

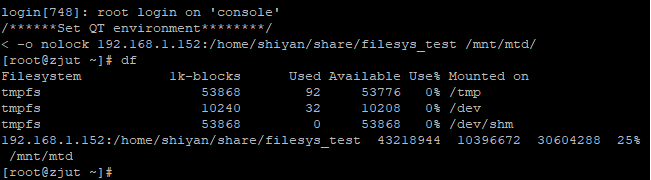


图12-6 文件夹挂载成功

1. 卸载

为了将/192.168.1.\*\*\*: /home/shiyan/shiare/nfs目录与/mnt/mtd 目录卸载分开，首先退到root 目录下（cd / 请注意卸载命令发生在实验箱端，且一定要在卸载挂载 前退出挂载目录，否则会报错，报错内容为设备忙），需要使用 umount 命令（umount 被挂载目录），如下所示：

[root]# umount /mnt/mtd

df查看后已无显示服务器的内容，完成卸载。

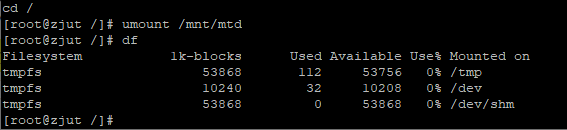


图12-7 文件夹卸载成功

**步骤4：**usb挂载（FAT32格式U盘）

1. 将U盘插入实验板的USB接口处，实验板中的串口调试工具出现以下信息提示：

[root]# [149.340000] usb 1-1.3:new high speed USB device using musb\_hdrc and address 4



图12-8 插入U盘

1. 使用fdisk-l查看盘符详细，如下所示：

[root]# fdisk -1

Disk /dev/sda: 4057 MB, 4057989120 bytes

91 heads, 45 sectors/track, 1935 cylinders

Units = cylinders of 4095 \* 512 = 2096640 bytes

Device Boot Start End Blocks Id System

/dev/sda1 1 1936 3962852 b Win95 FAT32

[root]#

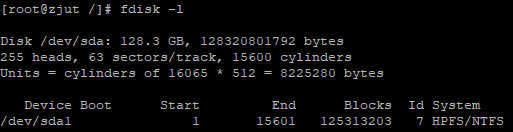


图12-9 输入fdisk -l查看盘符

1. 创建一个/mnt/usb 文件夹，如下所示：

[root]# mkdir /mnt/usb

[root]#

1. 把sda1盘符mount到/mnt/usb文件夹上，如下所示：  
   [root]# mount/dev/sda1/mnt/usb/

[root]#

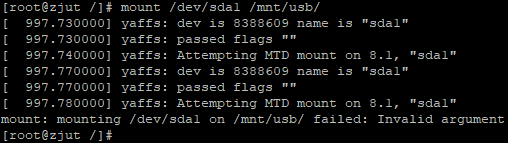


图12-10 尝试挂载U盘

但是因为插入U盘格式为exFAT，在实验箱中未能成功挂载

1. 进入/mnt/usb/文件夹，查看文件夹中的内容，如下所示：

[root]# cd /mnt/usb/

[root]# ls

h264 bin disk.tar.gz etc data.h264 dev init lib

linuxrc mnt proc root sbin share shm sys

[root]#

1. 卸载U盘，先退到根目录下，再解除挂载，如下所示：

[root@zjut usb]# cd /

[root@zjut ~]# umount /mnt/usb

1. 解除挂在以后，可再次进入/mnt/usb/文件夹，输入ls查看，若文件夹内已经没有内容，说明解除挂在成功：  
   [root@zjut ~]# cd /mnt/usb

[root@zjut usb]# ls

[root@zjut usb]#

1. 心得与体会

在挂载实验中部分同学使用的是实验室的三台服务器，部分同学使用的是VMware创建的Ubuntu虚拟机，IP等配置有所不同，并且由于没分配好用户和IP导致出现了比较多的冲突；同时实验箱卡顿也比较严重，常常出现死机的状况，重启后并不能进入实验箱系统，最后并没能在实验室中做完全部实验，总结出两个问题就是课前没有及时预习，对挂载实验的操作步骤已经IP分配并不熟悉，反复在同一步骤上浪费过多时间，第二个就是自主学习能力不够强，对出现的问题不能及时找到解决方法已知拖到了最后还没完成。但经过这次挂载实验，对如何加载文件系统和卸载有了更加清楚的认识，同时在之后的实验中也将更加重视实验的预习。

实验五：Linux交叉编译环境

1. 实验目的
2. 理解交叉编译的原理和概念。
3. 掌握在 Linux 下搭建交叉编译平台的方法。
4. 掌握使用交叉编译平台编译源代码。
5. 实验内容
6. 正确运行实验箱。
7. 通过串口线将实验箱和PC机链接。
8. 在PC机的Linux操作系统上搭建交叉编译平台，并编译程序。
9. 在实验箱上运行交叉编译程序结果。
10. 实验步骤

**步骤1：**直接进入虚拟机终端。（若是服务器择偶那个putty进入服务器）

**步骤2：**搭建交叉编译环境。

创建一个文件夹 mv\_pro\_5.0，进入文件夹 mv\_pro\_5.0，将/home/shiyan/2021（虚拟机是在/home/shiyan目录）目录下的软件包mvtools5\_0\_0801921\_update.tar复制到当前目录 mv\_pro\_5.0 下（注意不能省略最后一条语句中的“.”，且前面有空格）：

# mkdir mv\_pro\_5.0

# cd mv\_pro\_5.0

解压缩mvltools5\_0\_0801921\_update.tar软件包，解压缩后出现montavista文件夹：

# tar zxvf mvltools5\_0\_0801921\_update.tar.gz

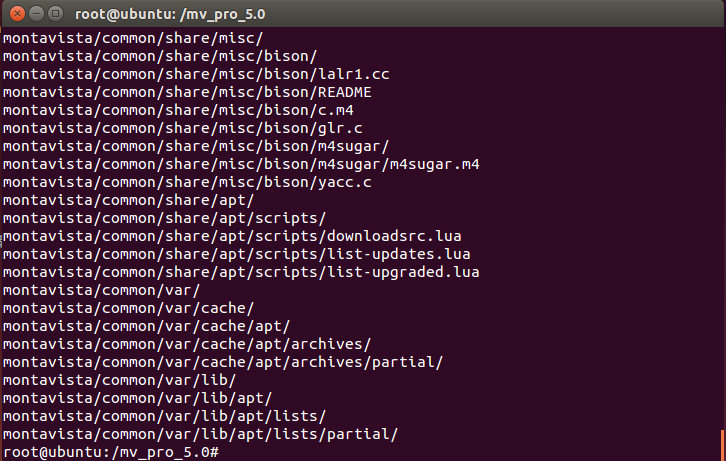


图5-1 创建文件夹拷贝压缩包并解压

配置系统环境变量，把交叉编译工具链路径添加到环境变量PATH钟去，使其可以在任何目录下使用，进入/etc/profile文档：

# vim /etc/profile

点击插入键i，在文件最后一行添加：

服务器：

export PATH=$PATH:/home/stX/mv\_pro\_5.0/montavista/pro/devkit/arm/v5t\_le/bin

虚拟机：

export PATH=$PATH:/home/mv\_pro\_5.0/montavista/pro/devkit/arm/v5t\_le/bin



图5-2 配置系统环境变量

点击ecs键退出输入，输入:wq!推出文档。使环境变量生效：

# source /etc/profile

检查交叉编译环境是否搭建成功：在命令行钟输入arm\_v5t\_le-gcc-v，打印出版本信息，表示交叉编译环境搭建成功。

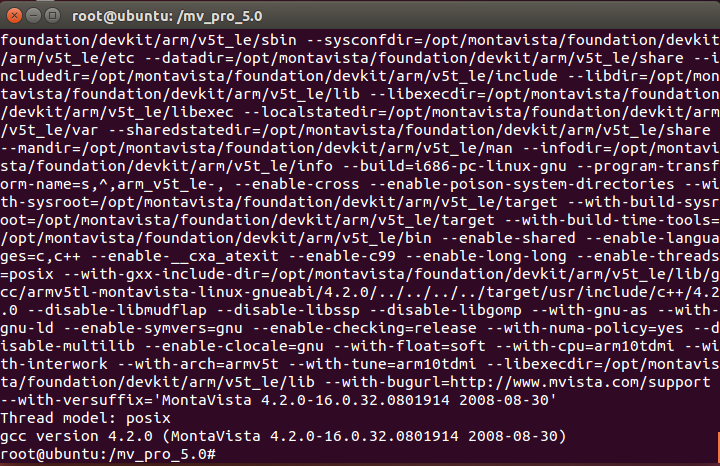


图5-3 交叉编译环境检测

**步骤3：**小程序测试

1. 步骤2钟搭建了交叉编译环境，接下来交叉编译一个小程序“helloworld”，挂载到实验箱上，新建vim文件（vim helloworld.c），输入以下内容：

#include<stdio.h>

int main()

{

printf("hello world !\n");

return 0;

}

点击esc键退出输入，输入:wq!退出文档。



图5-4 创建hello world程序

1. 交叉编译

生成二进制可执行文件helloworld，其中helloworld.c为交叉编译的程序，-o表示输出，helloworld表示生成的二进制可执行文件名：

# arm\_v5t\_le-gcc helloworld.c -o helloworld

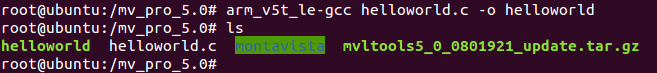


图5-5 生成二进制可执行文件

在PC上运行生成的二进制可执行文件helloworld：

# ./helloworld

提示-bash:./helloworld:cannot execute binary file: Exec format error，即不能执行该二进制可执行文件。



图5-6 报错不能执行可执行文件

1. 正确连接实验箱和PC机

将 PC 机与开发板通过 USB 转串口线正确连接，将开发板的电源线、网线 正确连接，插上电源。

1. 登录Putty的COM口端

打开设备开关，在PuTTY的COM口端进行操作，当实验板有打印消息时， 按enter键使系统停止启动，输入启动参数；

保存启动参数# saveenv

1. 通过实验箱运行交叉编译生成的可执行文件

在 PuTTY 的服务器端进行操作，将生成的二进制可执行文件helloworld由 它所在的目录复制到文件系统所在目录的/opt/dm365下（eg:服务器文件系统所在目录为/home/shiyan/filesys\_test，就复制到/home/stX/filesys\_test /opt/dm365目录）；（虚拟机文件系统所在目录为/home/shiyan/share/filesys\_test/，就复制到/ho me/shiyan/share/filesys\_test/opt/dm365目录）：

服务器：# sudo cp helloworld /home/stX/filesys\_test /opt/dm365

虚拟机：# sudo cp helloworld /home/shiyan/share/filesys\_test/opt/dm365

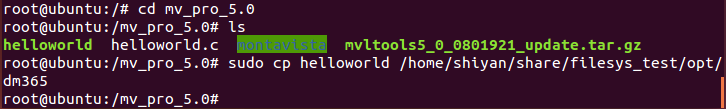


图5-7 将二进制可执行文件复制到文件系统的/opt/dm365下

在putty端口操作：

进入可执行文件所在目录/opt/dm365

# cd /opt/dm365

运行二进制可执行文件helloworld：

# helloworld

显示结果如下：

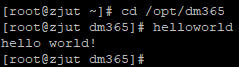


图5-8 交叉编译成功

1. 心得与体会

通过这次交叉编译实验，学会了如何搭建交叉编译环境，有了实验四挂载实验的基础后，实验进行地较为顺利，出现的问题也能够独立解决，对后续实验的顺利进行奠定了良好的基础。

实验六：Linux内核编译实验

1. 实验目的
2. 掌握配置和编译Linux内核的方法。
3. 掌握Linux内核的编译过程。
4. 熟悉Linux系统一些基本内核配置。
5. 熟悉熟悉内核编译的常见命令。
6. 实验内容
7. 利用编译进内核的方法配置内核。
8. 利用动态加载方法配置内核。
9. 编译Linux内核镜像。
10. 编译Linux内核模块。
11. 实验步骤

**步骤1：**登录虚拟机找到内核目录，移动物联网实验箱的内核基于2.6.18该进来的kernel-for-mecb。

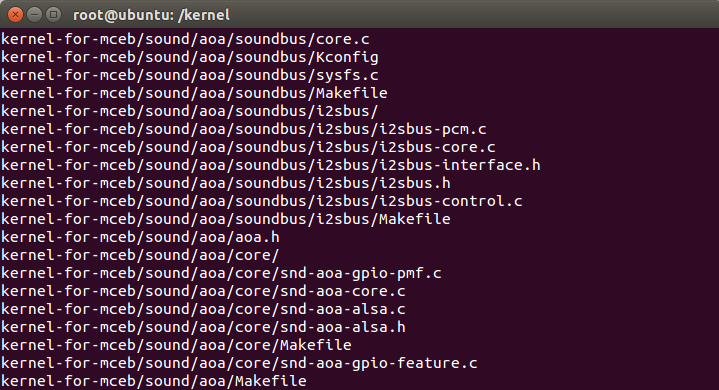


图6-1 创建文件夹kernel复制进压缩包后解压

服务器：#cp /home/shiyan/2021/ kernel-for-mceb.tar.gz ./

虚拟机：#cp /home/shiyan/Desktop/shiyan/ kernel-for-mceb.tar.gz ./

**步骤2：**进入内核进行配置

在配置前首先输入sudo -s，并输入密码，登录超级用户，之后再次输入命令vim/etc/profile检查交叉编译路径是否正确。之后退出，输入命令source /etc/profile，使环境变量生效。

进入内核目录，执行命令为cd kernel-for-mceb。如果不是第一次编译内核，先运行make mrproper清除以前的配置，huidaomoren 配置。然后继续进行内核配置，执行命令为make menuconfig。出现窗口如下图所示：

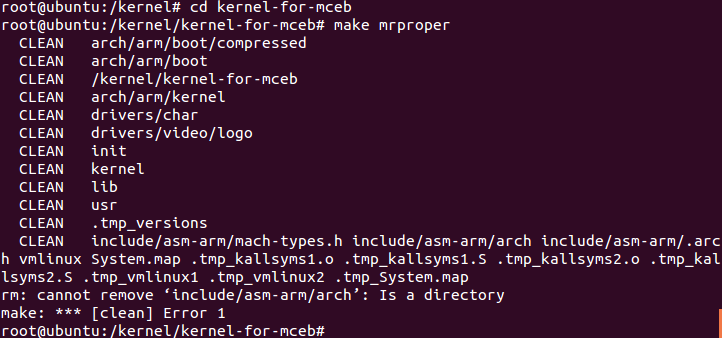


图6-2 回到默认配置

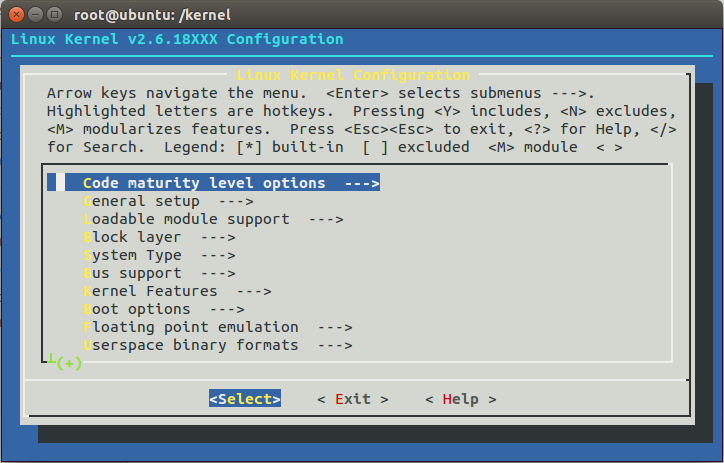


图6-3 配置内核

编译内核时往往根据自己的需要来编译自己所需要的。下面是一些常见的驱动选项。

|  |  |
| --- | --- |
| 选择 | 说明 |
| MemoryTechnologyDevices(MTD) | 配置存储设备，需要该选项使Linuxkeyiduqu闪存卡（Flash、Card）存储器 |
| Parallel port support | 配置并口。如果不使用，可不选 |
| Block devices | 块设备支持 |
| ATA/ATAPI/MFM/RLL support | IDE 硬盘和 ATAPI 光驱，纯 SCSI 系统且不使用这些接口，可以不选 |
| SCSI device support | SCSI 仿真设备支持 |
| Multi-devicesupport(RAID and LVM) | 多设备支持（RAID 和 LAM） |
| Fusion MPT device support | MPT设备支持 |
| IEEE 1394(FireWire)support | IEEE 1394（火线） |
| I2O device support | I2O 设备支持。如果有 I2O 界面，必须选中。是由于智能I/O系统的标准接口 |
| Network device support | 内核在没有网络支持选项的情况下甚至无法编译。是必选项。 |
| ISDN subsystem | 综合数字业务网 |
| Input device support | 输入设备，包括鼠标、键盘等 |
| Character devices | 字符设备，包括虚拟终端、串口、并口等设备 |
| I2C support | 用于监控电压、风扇转速、温度等。 |
| Hardware Monitoring support | 需要I2C的支持 |
| Misc devices | 杂项设备 |
| Multimedia Capabilities Port drivers | 多媒体功能接口驱动 |
| Multimedia devices | 多媒体设备 |
| Graphics support | 图形设备/显卡支持 |
| Sound | 声卡 |
| USB support | USB接口支持配置 |
| MMC/SD Card support | MMC/SD卡支持 |

**步骤3：**配置选择内核定制，选择自己需要的功能。按键盘空格键进行选择\*表示选定直接编译进内核，M表示选定模块编译为动态加载模块。在这里，以让内核支持ntfs文件系统为例。

1. 在make menuconfig命令打开的窗口中选择Files systems。如下图：

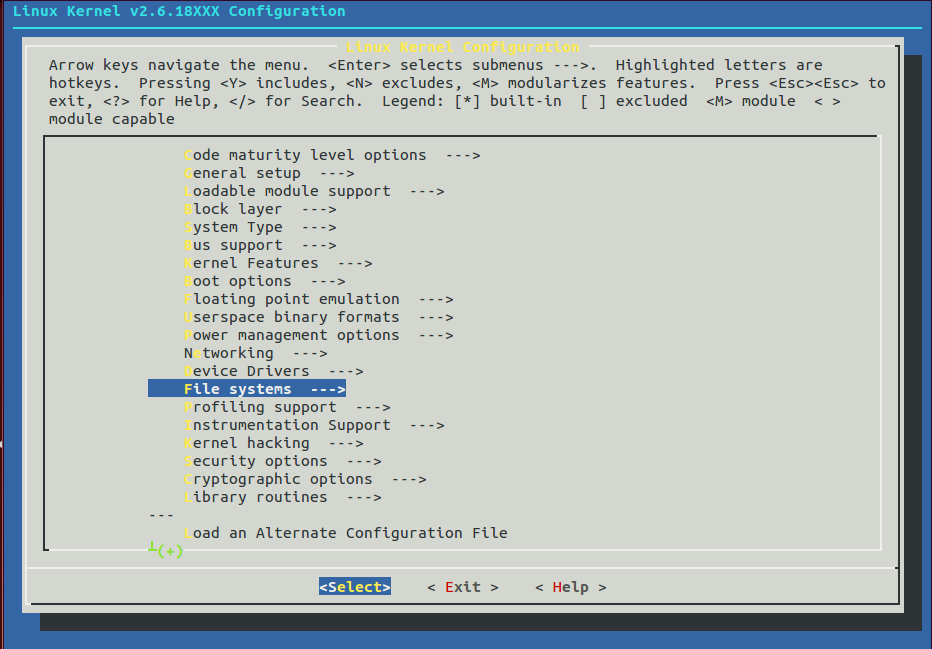


图6-4 文件系统配置

1. 敲回车后，继续选择能支持ntfs选择系统类型的选项。如下图：

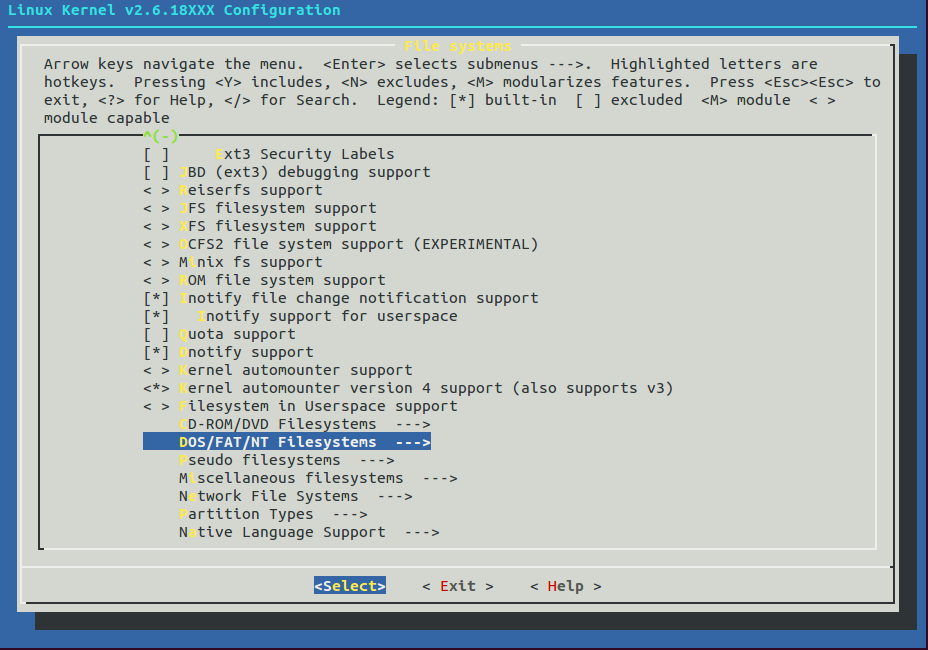


图6-5 文件系统选项

1. 继续回车键，最后选择我们需要的ntfs文件系统类型。如下图：

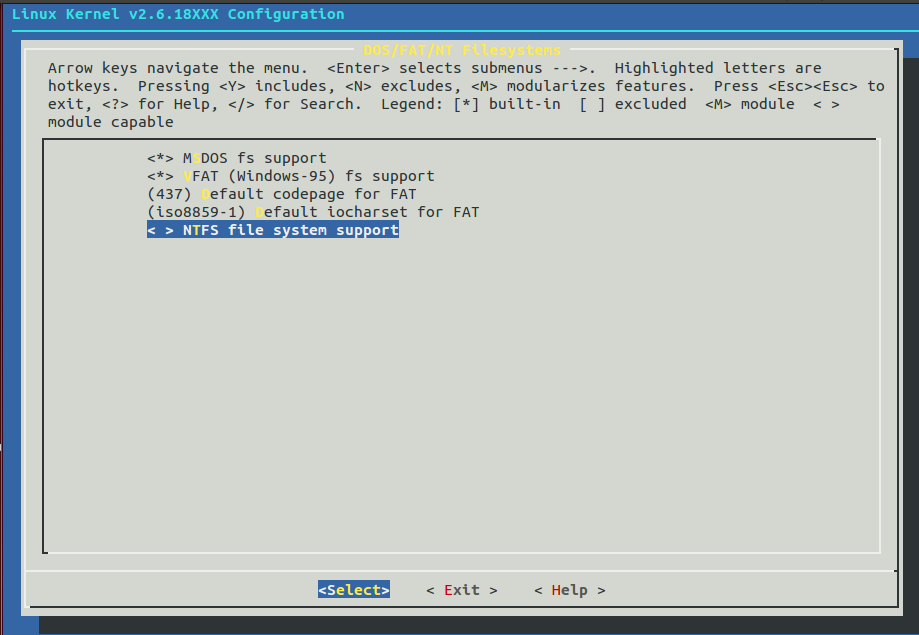


图6-6 NTFS文件系统选项

1. 按空格键选择编译进内核。并在键盘上左右键移动光标退出键按钮，按回车键不断退出。如下图所示：

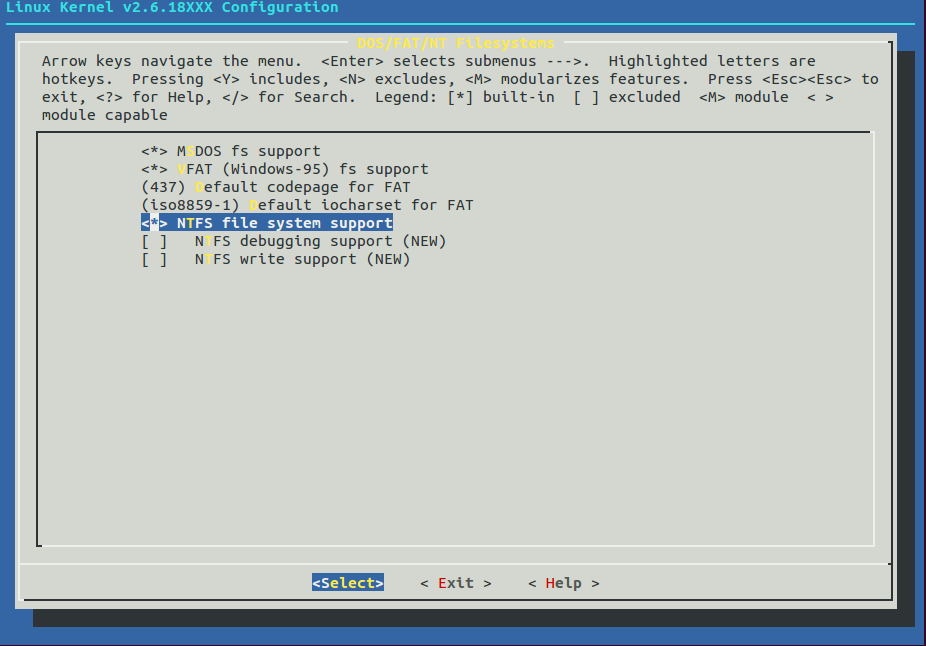


图6-7 配置为编译进内核

1. 不断回车后，出现是否保存界面，选择yes保存配置，回车键退出。

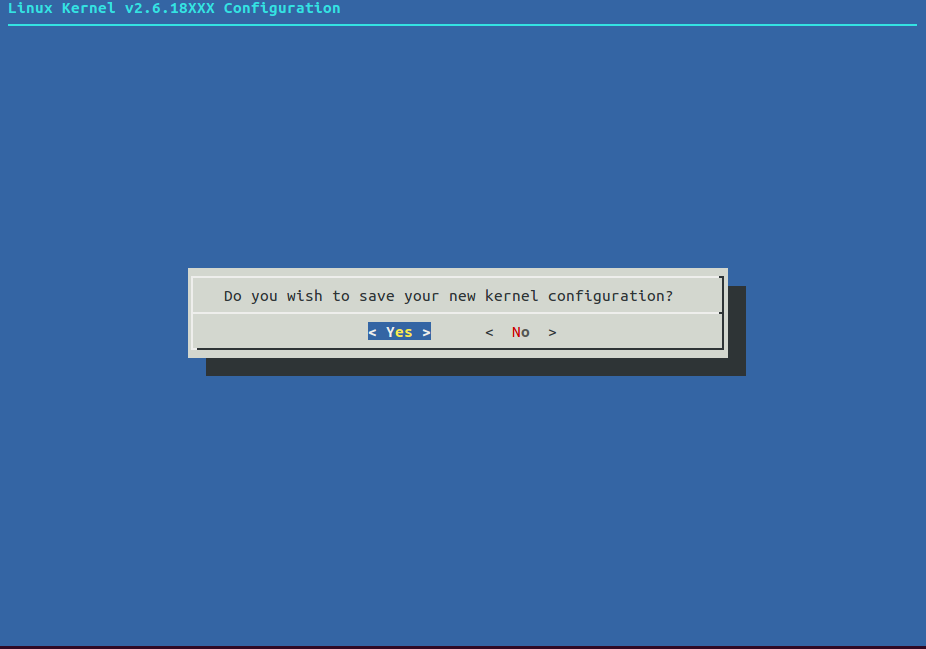


图6-8 配置退出保存界面

**步骤4：**编译内核镜像

编译内核时候一般需要root权限，对特定用户加sudo命令即可。在内核目录下使用命令make uImage。回车键后内核开始编译，等待出现 Image arch/arm/boot/uImage is ready表示编译结束。编译好后再目录arch/arm/boot/下生成一个uImage二进制文件。如图所示：

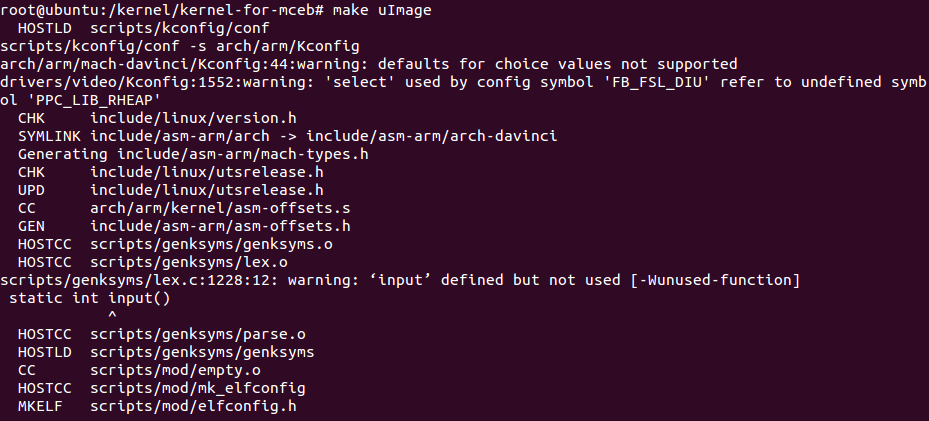


图6-9 开始内核编译

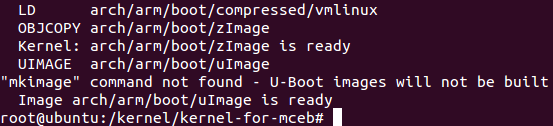


图6-9 编译生成的内核镜像

这就是利用编译进内核的方法编译生成的内核镜像，可以移植到实验板子 上。对于一般的要求，利用编译进内核的方法就足够了。不过对于许多实验和工程的要求，为了减轻内核的负担，往往是需要利用动态加载的方法。下面继续实 验来熟悉动态模块编译的方法。

**步骤5：**将NTFS文件系统配置成模块方式

按步骤3钟的方法，将NTFS file system support嵌满\*改成为M，即将ntfs文件系统配置成为模块加载形式。如下图所示：

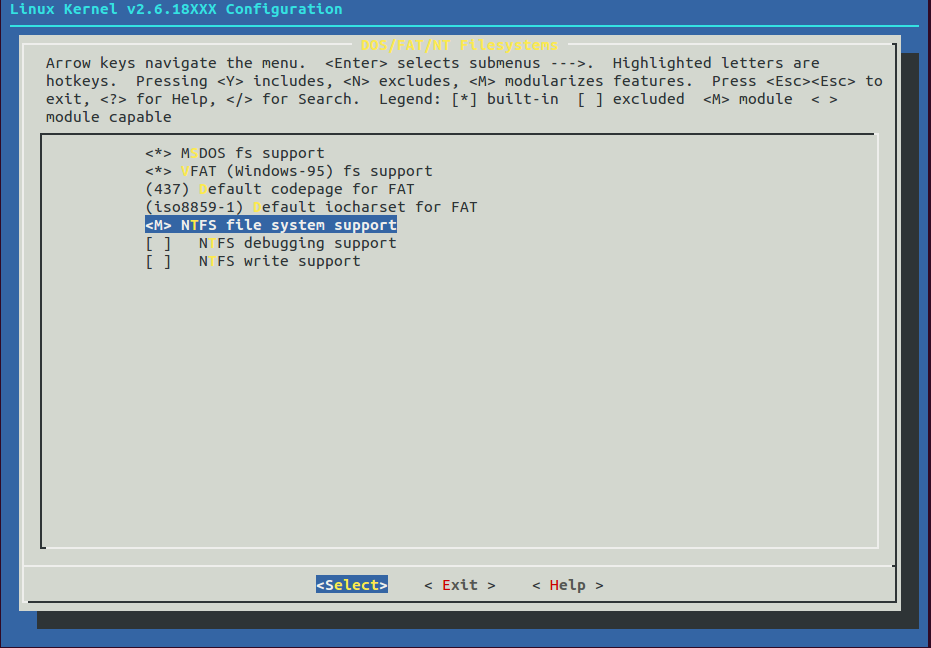


图6-10 内核配置为模块方式

**步骤6：**重新编译内核镜像由于编译生成的模块需要被加载到内核中才能使用， 而开始编译生成的内核是一个可以支持ntfs文件系统的内核。可以先执行make clean，清除刚才生成的镜像，然后执行make uImage生成一个新内核镜像。即新生成的内核不支持ntfs文件系统，从而可以将ntfs文件模块添加进去，使内核可以支持ntfs文件系统。

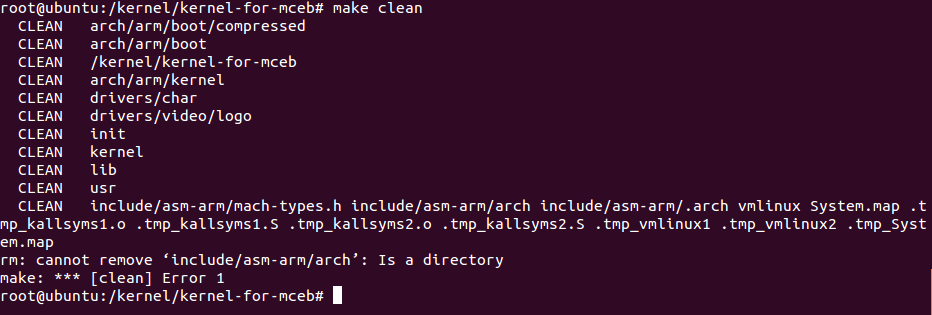


图6-11 清除刚才生成的镜像

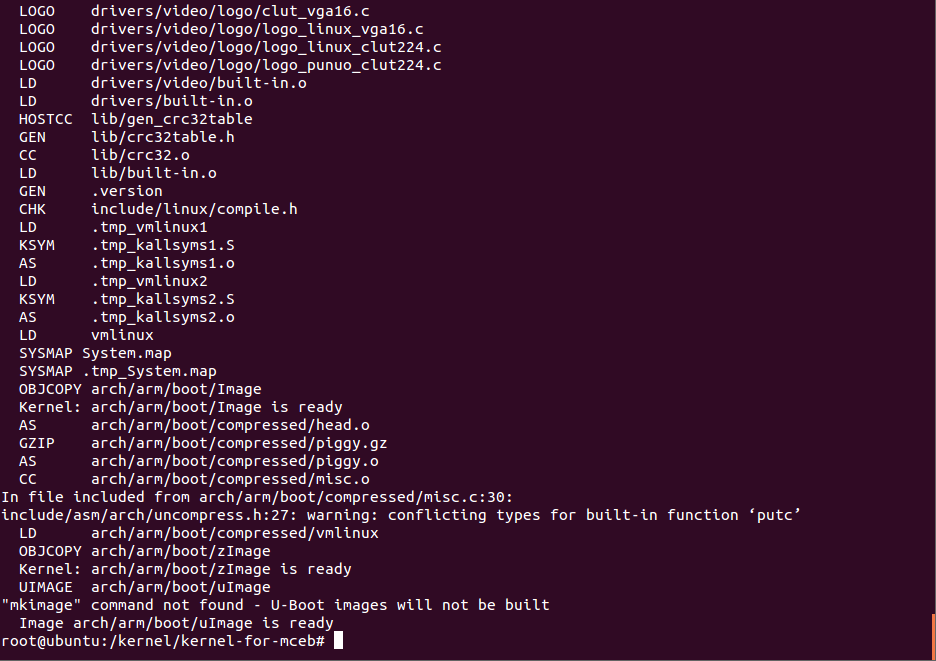


图6-12 模块编译结束

1. 心得与体会

通过这次内核编译实验，对ARM的内核结构有了更为深入的理解，并能够学会正确配置内核，编译内核镜像，对于其中重要的参数也有更为全面的认识和理解。