

# 嵌入式系统设计方法—— Lab5

PB18111697 王章瀚

## 编译安装 busybox

- busybox 版本: 1.22.0

## 遇到问题及解决方案

### RLIMIT\_FSIZE undeclared

如下图所示:

```
loginutils/passwd.c: In function 'passwd_main':
loginutils/passwd.c:93:16: error: storage size of 'rlimit_fsize' isn't known
  93 | struct rlimit rlimit_fsize;
      | ^~~~~~
loginutils/passwd.c:172:2: warning: implicit declaration of function 'setrlimit' [-Wimplicit-function-declaration]
  172 | setrlimit(RLIMIT_FSIZE, &rlimit_fsize);
      | ^~~~~~
loginutils/passwd.c:172:12: error: 'RLIMIT_FSIZE' undeclared (first use in this function)
  172 | setrlimit(RLIMIT_FSIZE, &rlimit_fsize);
      | ^~~~~~
```

### 解决方案

就是在 libbb.h 加一句

```
#include "sys/resource.h"
```

```
from networking/ifenslave.c:116:
/usr/include/linux/sysinfo.h:8:8: error: redefinition of 'struct sysinfo'
  8 | struct sysinfo {
      | ^~~~~~
In file included from networking/ifenslave.c:101:
```

2.

### 解决方案

3. rdate

```
/usr/bin/ld: debianutils/lib.a(mktemp.o): in function 'mktemp_main':
mktemp.c:(.text.mktemp_main+0xa6): 警告: the use of 'mktemp' is dangerous, better use 'mkstemp' or 'mkdtemp'
/usr/bin/ld: util-linux/lib.a(rdate.o): in function 'rdate_main':
rdate.c:(.text.rdate_main+0x117): undefined reference to 'stime'
/usr/bin/ld: coreutils/lib.a(date.o): in function 'date_main':
date.c:(.text.date_main+0x27e): undefined reference to 'stime'
collect2: error: ld returned 1 exit status
make: *** [Makefile:716: busybox_unstripped] 错误 1
```

解决方案: 干脆不要勾选:

- Coreutils/date:

```
[*] cat
[*] date
[*] hostid
```

- Linux System Utilities/rdate

```
[*] pivot_root
[*] rdate
[*] rdev
[*] readprofile
```

至此 busybox 编译安装完成

```
./_install//usr/sbin/setlogcons -> ../../bin/busybox
./_install//usr/sbin/svlogd -> ../../bin/busybox
./_install//usr/sbin/telnetd -> ../../bin/busybox
./_install//usr/sbin/tftpd -> ../../bin/busybox
./_install//usr/sbin/ubiattach -> ../../bin/busybox
./_install//usr/sbin/ubidetach -> ../../bin/busybox
./_install//usr/sbin/ubimkvol -> ../../bin/busybox
./_install//usr/sbin/ubirmvol -> ../../bin/busybox
./_install//usr/sbin/ubirsvol -> ../../bin/busybox
./_install//usr/sbin/ubiupdatevol -> ../../bin/busybox
./_install//usr/sbin/udhcpd -> ../../bin/busybox
```

```
-----
You will probably need to make your busybox binary
setuid root to ensure all configured applets will
work properly.
-----
```

## 构建根文件系统

1. 创建所需目录:

```
mkdir dev etc home lib mnt proc root sys tmp var -p
```

2. 在创建的根文件系统的 etc目录下创建inittab文件:

```
touch etc/inittab
vim etc/inittab
```

写入

```
#this is run first except when booting in single-user mode.
::sysinit:/etc/init.d/rcS
# /bin/sh invocations on selected ttys
::respawn:/bin/sh
# Start an "askfirst" shell on the console (whatever that may be)
::askfirst:/bin/sh
# Stuff to do when restarting the init process
::restart:/sbin/init
# Stuff to do before rebooting
::ctrlaltdel:/sbin/reboot
::shutdown:/sbin/swapoff -a
```

### 3. 构建 init.d 及其内容

```
mkdir etc/init.d/ -p
touch etc/init.d/rcS
vim etc/init.d/rcS
```

写入:

```
#!/bin/sh
#This is the first script called by init process
/bin/mount -a
echo /sbin/mdev>/proc/sys/kernel/hotplug
mdev -s
```

### 4. 构建 fstab 及其内容

```
touch etc/fstab
vim etc/fstab
```

#device	mount-point	type	options	dump	fsck order
proc	/proc	proc	defaults	0	0
tmpfs	/tmp	tmpfs	defaults	0	0
sysfs	/sys	sysfs	defaults	0	0
tmpfs	/dev	tmpfs	defaults	0	0

### 5. 创建 profile 文件

```
touch etc/profile
vim etc/profile
```

```
#!/bin/sh
export HOSTNAME=pb18111697
export USER=root
export HOME=root
export PS1="[$USER@$HOSTNAME \w]\# "
#export PS1="[ \033[01;32m\]$USER@[\033[00m\][\033[01;34m\]$HOSTNAME[\033[00m\ \
PATH=/bin:/sbin:/usr/bin:/usr/sbin
LD_LIBRARY_PATH=/lib:/usr/lib:$LD_LIBRARY_PATH
export PATH LD_LIBRARY_PATH
```

## 6. 添加动态库支持

查找位置:

```
armlinuxgccpath=$(which arm-none-linux-gnueabi-gcc)
armlinuxgccpath=${armlinuxgccpath%/*}/..
cp $armlinuxgccpath/arm-none-linux-gnueabi/libc/lib lib -ra
```

```
(torch1.4) rabbit@rabbit:~/embed/lab5/rootfs$ armlinuxgccpath=$(which arm-none-linux-gnueabi-gcc)
(torch1.4) rabbit@rabbit:~/embed/lab5/rootfs$ armlinuxgccpath=${armlinuxgccpath%/*}/..
(torch1.4) rabbit@rabbit:~/embed/lab5/rootfs$ cp $armlinuxgccpath/arm-none-linux-gnueabi/libc/lib lib -ra
(torch1.4) rabbit@rabbit:~/embed/lab5/rootfs$ ls
bin dev etc home lib linuxrc mnt proc root sbin sys tmp usr var
(torch1.4) rabbit@rabbit:~/embed/lab5/rootfs$ cd lib/
(torch1.4) rabbit@rabbit:~/embed/lab5/rootfs/lib$ ls
ld-2.8.so          libgcc_s.so.1      libnss_nisplus-2.8.so
ld-linux.so.3      libm-2.8.so         libnss_nisplus.so.2
libanl-2.8.so       libmemusage.so      libnss_nis.so.2
libanl.so.1         libm.so.6           libpcprofile.so
libBrokenLocale-2.8.so libnsl-2.8.so        libpthread-2.8.so
libBrokenLocale.so.1 libnsl.so.1          libpthread.so.0
libc-2.8.so         libnss_compat-2.8.so libresolv-2.8.so
libcldn-2.8.so      libnss_compat.so.2  libresolv.so.2
libcldn.so.1        libnss_dns-2.8.so   librt-2.8.so
libcrypt-2.8.so     libnss_dns.so.2     librt.so.1
libcrypt.so.1       libnss_files-2.8.so libSegFault.so
libc.so.6           libnss_files.so.2   libthread_db-1.0.so
libdl-2.8.so        libnss_hesiod-2.8.so libthread_db.so.1
libdl.so.2          libnss_hesiod.so.2  libutil-2.8.so
libgcc_s.so         libnss_nis-2.8.so   libutil.so.1
```

## 创建镜像 image 文件

使用飞凌官方提供的 `mkyaffs2image-nand2g` 来生成. 见如下命令:

```
mkyaffs2image-nand2g rootfs rootfs.yaffs2
```

其中 `rootfs` 是刚才所生成的根文件系统的路径, `rootfs.yaffs2` 是生成的文件名.

## 烧写

烧写过程和第一次实验无异,

1. 开启 sd 卡启动模式
2. 烧写
3. 开启 nand 启动模式
4. 打开设备

## 启动系统并测试

### 遇到问题及解决方案

遇到了一个问题: modprobe 的地方有点问题, 如下:

```
request_module: runaway loop modprobe binfmt-464c
request_module: runaway loop modprobe binfmt-464c
request_module: runaway loop modprobe binfmt-464c
request_module: runaway loop modprobe binfmt-464c
request_module: runaway loop modprobe binfmt-464c
```

我尝试取消勾选 modprobe , 但毫无效果.

查阅了 [参考资料](#), 在 busybox 中配置交叉编译的 prefix 后才成功.

```
(arm-linux-) Cross Compiler prefix
() Path to sysroot
```

## 测试

成功启动后,

1. 首先可以看到主机名是我设置了的 pb18111697
2. 然后执行我提前放入的 HelloWorld , 结果如期.

```
fixmap : 0xffff0000 - 0xffffe000 ( 896 kB)
DMA : 0xff600000 - 0xffe00000 ( 8 MB)
vmalloc : 0xd0800000 - 0xf4000000 ( 568 MB)
lowmem : 0xc0000000 - 0xd0000000 ( 256 MB)
pkmap : 0xbfe00000 - 0xc0000000 ( 2 MB)
modules : 0xbf000000 - 0xbfe00000 ( 14 MB)
 .init : 0xc0008000 - 0xc0036000 ( 184 kB)
 .text : 0xc0036000 - 0xc07d4434 (7802 kB)
 .data : 0xc07d6000 - 0xc082eaf0 ( 355 kB)
 .bss : 0xc082eb14 - 0xc08b1124 ( 522 kB)
SCSI subsystem initialized
-----[ cut here ]-----
WARNING: at drivers/gpio/gpiolib.c:101 gpio_ensure_requested+0x58/0x124()
---[ end trace da227214a82491b7 ]---
type=2000 audit(0.190:1): initialized
Hello, world
Creating 4 MTD partitions on "NAND 2GiB 1.8V 8-bit":
```

```
0x00000000000000-0x00000002000000 : "Bootloader"
0x00000005000000-0x0000001e000000 : "Kernel"
0x0000001e000000-0x000000e6000000 : "File System"
0x000000e6000000-0x00000040000000 : "User"
mcp2515 spi1.0: MCP251x didn't enter in conf mode after reset
mcp2515 spi1.0: probe failed
dm9000 dm9000.0: read wrong id 0x01010101
s3c_g2d_probe called
s3c_g2d_probe Success
Registering the dns_resolver key type
can't run '/etc/init.d/rcS': Permission denied

Please press Enter to activate this console.
[root@pb18111697 ]# ls
HelloWorld  etc          linuxrc      proc         sys          var
bin         home        lost+found   root         tmp
dev         lib         mnt          sbin         usr
[root@pb18111697 ]# ./HelloWorld
Hello, World!
[root@pb18111697 ]#
```

## 实验总结

本次实验让我学会了如何构建, 生成, 并烧写一个根文件系统, 收获颇丰.