修改 Linux 的开机 LOGO

LOGO 原理

嵌入式 Linux 是直接在 FrameBuffer 的基础上。直接显示一个 ppm 格式的图象，它由 kernel/drivers/video/fbcon.c 中的 fbcon_show_logo()完成，最大颜色支持 224 色而不常见的 255 色。默认的 logo 文件是 drivers/video/logo/logo_linux_clut224.ppm。

LOGO 制作

因为 LINUX LOGO 格式需要 ppm 格式来显示。这种格式是一种用 ASCII 来描述图像数据一种格式。一般只有少数软件能识这种这种格式。 因此假设你有一张做好的 LOGO 图片（JPG 或 PNG 格式，最好是后者），接下来就可以制作 LOGO PPM 格式.，首先你要保证你的 LOGO 的尺寸不能超过你的屏幕尺寸，在各种情况能显示。

FTP：//wikiusers@ftp.arm9board.net/Kernel_logo （密码：Arm9board）

改变像素大小

```
$ convert -resize 240x240! Mylogo.png Mylogo1.png
```

图片格式转换工具：netpbm

```
$ sudo apt-get install netpbm
```

格式转换指令：

```
$ pngtopnm mylogo.png > mylogo.pnm        png 图片转成 pnm
```

将 pnm 图片的颜色数限制在 224

```
$ pnmquant 224 mylogo.pnm > mylogo_224.pnm
```

将 pnm 图片转换成我们需要的 ppm

```
$ pnmtoplainpnm mylogo_224.pnm > logo_linux_clut224.ppm
```

将 pnm 图片转换成我们需要的 ASCII 模式：

```
$ pnmnoraw logo_linux_clut224.ppm> logo_linux_custom_ascii_224.ppm
```

LOGO 内核添加

$ git clone https://github.com/raspberrypi/tools.git

$ git clone https://github.com/raspberrypi/linux.git

方案一：LOGO 替换

将制作好的 ppm 文件替换源码中的 logo

```
$ cp logo_linux_custom_ascii_224.ppm  drivers/video/logo
$ mv logo_linux_clut224.ppm logo_linux_clut224bk.ppm(图标备份)
$ mv logo_linux_custom_ascii_224.ppm logo_linux_clut224.ppm
```

方案二：LOGO 添加

编辑必要的文件

移动到徽标目录：

```
$ cd Linux
```

编辑 Makefile

```
$ gedit Makefile
```

在第 19 行附加下列行：

```
obj-$(CONFIG_LOGO_ARMADEUS_CLUT224)+=logo_custom_clut224.o
```

编辑 logo.c 文件

```
$ gedit logo.c
```

在所有_CLUT224 ifdef 例程之间添加以下行：

```
#ifdef CONFIG_LOGO_ARMADEUS_CLUT224
    /* Armadeus Linux logo */
    logo = &logo_custom_clut224;#endif
```

编辑 Kconfig 文件

```
$ gedit Kconfig
```

并添加以下行添加您自己的启动徽标到 menuconfig，当然你可以将文本更改为任何你想要的：

```
config LOGO_CUSTOM_CLUT224
 bool "Your own boot logo here"
 default y
```

编辑头文件

移动到 linux 内核的 include 目录：

```
$ cd Linux/include
$ gedit linux_logo.h
```

在第 50 行附近，添加以下行：

```
extern const struct linux_logo logo_custom_clut224;
```

编译：

```
$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- menuconfig

Device Drivers  --->
    Graphics support  --->
       [*] Bootup logo  --->
           [*] Your own boot logo here
```

```
 设备驱动程序--->
       图形支持--->
          [*] Bootup logo --->
             [*]这里你自己的开机标志
```

LOGO 内核编译

```
For Pi 1 or Compute Module:
$ cd linux
$ KERNEL=kernel
$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- bcmrpi_defconfig

For Pi 2/3:
cd linux
KERNEL=kernel7
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- bcm2709_defconfig

for both:
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- zImage modules dtbs
```

编译源码时，在编译内核使用 make menuconfig 时出现如下错误：

```
HOSTLD  scripts/kconfig/mconf
scripts/kconfig/mconf.o: In function `show_help':
mconf.c:(.text+0x811): undefined reference to `stdscr'
```

安装 libncurses5-dev 可以解决

```
$ sudo apt-get install libncurses5-dev
```

LOGO 内核移植

使用"lsblk"查看 Linux 硬盘
$ lsblk

文件挂载

sdb1 为 the FAT (boot) partition， sdb2 为 the ext4 filesystem (root) partition.

```
$ mkdir /mnt/fat32
$ mkdir /mnt/ext4
$ sudo mount /dev/sdb1 /mnt/fat32
$ sudo mount /dev/sdb2 /mnt/ext4
```

安装模块

```
$ sudo make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- INSTALL_MOD_PATH=/mnt/ext4 modules_install
```

最后，复制内核到 SD 卡，确保备份旧的内核：

```
$ sudo cp /mnt/fat32/$KERNEL.img /mnt/fat32/$KERNEL-backup.img
$ sudo scripts/mkknlimg arch/arm/boot/zImage /mnt/fat32/$KERNEL.img
$ sudo cp arch/arm/boot/dts/*.dtb /mnt/fat32/
$ sudo cp arch/arm/boot/dts/overlays/*.dtb* /mnt/fat32/overlays/
$ sudo cp arch/arm/boot/dts/overlays/README /mnt/fat32/overlays/
$ sudo umount /mnt/fat32
$ sudo umount /mnt/ext4
```

另一个选择是复制内核在同一个地方，用不同的文件名，例如，kernel-myconfig.img 而不是覆盖 kernel.img 文件。然后你可以编辑 config.txt 文件选择内核，PI 将引导到：

```
kernel=kernel-myconfig.img
```

这有保持您的内核从系统管理的内核映像和任何自动更新工具分开的优势，并且允许您在内核无法启动的情况下轻松地恢复到一个库存内核。

番外篇：LOGO 属性设置

logo 居中：
1、设置 logo 图片在屏幕中的位置

```
$ vi drivers/video/fbmem.c
```

找到"fb_show_logo_line"函数，把
image.dx = 0;
image.dy = y;
改为
image.dx = (info->var.xres/2) - (610/2);
image.dy = (info->var.yres/2) - (206/2);
[注：
info->var.xres 和 info->var.yres 是分辨率大小
610 和 206 是 logo 图片的大小

2、
$ vi drivers/video/console/fbcon.c
找到"fbcon_prepare_logo"函数，在
logo_height = fb_prepare_logo(info, ops->rotate);
后面加上
logo_height += (info->var.yres/2) - (206/2);

3. logo 全屏
搞定了 在配置的时候 在 Console drivers --->Frame-buffer support --->
把 Select compiled-in fonts 去掉,就能成功的做全屏的 logo 了