目录

前言

> 本教程主要用于指导大家制作可引导的SD卡,从SD卡启动easyarm_imx280a开发板,并挂载rootfs文件系统.
> 交流学习请联系: **whjwnavy@163.com** or **805400349.com**

u-boot修改适配

> 开发板资料光盘中所给的uboot源码编译得到的 **imx28_ivt_uboot.sb** (***ivt*** *表示是经过签名的固件*)镜像文件,只要通过 **cfimager.exe** 工具烧录到SD卡中,然后选择SD卡启动,上电就可以直接进入uboot了. 烧录脚本如下[*根据原来的* ***TF烧写固件（uboot启动）.bat*** *脚本修改而成*]:

```
@echo off
echo.
echo EasyARM-iMX28x 制作SD卡启动程序
echo.
echo 请输入SD卡盘符:
set /p diskpath=
set cmdpath=%~dp0
echo 注意:
echo 文件会被烧写在 %diskpath% 盘
echo.
%cmdpath%cfimager.exe -a -f %cmdpath%imx28_ivt_uboot.sb -d %diskpath%
echo 烧写完毕，按键退出
echo.
pause>nul
```

> 但是这样直接编译得到的uboot镜像是 **不支持** 保存环境变量到SD卡中的,输入saveenv命令仍然会把环境变量保存到NAND FLASH中,这样是没有意义的,所以必须要修改uboot源码,以支持保存环境变量到SD卡中.
> 这里只需要修改一个宏定义就可以了,修改uboot源码中 **bootloader/u-boot-2009.08/include/configs/mx28_evk.h** 文件的第269行:
>
> 把:

```
//#define CONFIG_FSL_ENV_IN_MMC
#undef CONFIG_FSL_ENV_IN_MMC
#define CONFIG_FSL_ENV_IN_NAND
```

> 改为:

```
#define CONFIG_FSL_ENV_IN_MMC
//#undef CONFIG_FSL_ENV_IN_MMC
//#define CONFIG_FSL_ENV_IN_NAND
```

> 然后重新编译uboot,生成新的imx28_ivt_uboot.sb文件:

```
# make ARCH=arm CROSS_COMPILE=arm-fsl-linux-gnueabi- distclean
# make ARCH=arm CROSS_COMPILE=arm-fsl-linux-gnueabi- mx28_evk_config
# make ARCH=arm CROSS_COMPILE=arm-fsl-linux-gnueabi-
# cp -av u-boot ../imx-bootlets-src-10.12.01/
# cd ../imx-bootlets-src-10.12.01/
# make CROSS_COMPILE=arm-fsl-linux-gnueabi- BOARD=iMX28_EVK
# cp -av ./imx28_ivt_uboot.sb $(TFTPBOOT) -fr
```

> 这样修改后编译出来的uboot就支持保存环境变量到SD卡中了.

制作rootfs分区镜像

> 既然是通过SD卡启动,当然需要把rootfs文件系统也挂载到SD卡中,这里为了便于使用 **cfimager.exe** 工具烧写文件系统到SD卡中,需要把rootfs文件系统所在分区提取成镜像(*类似于Ghost中的备份分区成GHO文件*):

> 把SD卡插入到Linux系统主机中,用分区工具(*比如* ***fdisk*** *命令*)给SD卡分一个ext2格式的分区(imx28开发板可能会不支持ext3格式),然后挂载该分区, 假设sd卡在系统中的设备节点为 **/dev/sdc**.

```
# fdisk /dev/sdc
    Command (m for help): [n回车]
    Select (default p): [回车]
    Partition number (1-4, default 1): [回车]
    First sector (2048-2097151, default 2048): [回车]
    Command (m for help): [w回车]
# mkfs.ext2 -q /dev/sdc0
# mkdir /media/rootfs
# mount /dev/sdc0 /media/rootfs
```

> 假设SD卡上的rootfs分区挂载在系统的 **/media/rootfs** 目录下,然后解压 **rootfs.tar.bz2** 到该分区,最后通过 **dd** 命令备份该分区为rootfs分区镜像:

```
# tar -xjf rootfs.tar.bz2 -C /media/rootfs
# dd if=/dev/sdc0 of=rootfs.ext2.img
```

> 只需要制作一次,如果文件系统没有变动的话以后就可以一直使用这次制作好的rootfs分区镜像.

制作全0文件镜像

> 通过dd命令制作一个127Kb大小的内容全为0的镜像文件,用于填充uboot环境变量区域:

```
# dd if=/dev/zero of=zero.raw bs=1K count=127
```

> 关于环境变量区的大小,在uboot源码中 **bootloader/u-boot-2009.08/include/configs/mx28_evk.h** 文件的第286行:

```
#elif defined(CONFIG_FSL_ENV_IN_MMC)
    #define CONFIG_ENV_IS_IN_MMC    1
    /* Assoiated with the MMC layout defined in mmcops.c */
    #define CONFIG_ENV_OFFSET               (0x400) /* 1 KB, 环境变量分区的偏移地址 */
    #define CONFIG_ENV_SIZE                 (0x20000 - 0x400) /* 127 KB,  环境变量分区的大小*/
#else
```

烧录镜像到SD卡

> 我们已经有了 **imx28_ivt_uboot.sb**, **rootfs.ext2.img**, **zero.raw** 这三个文件,在烧录之前你当然还需要一个编译好的Linux内核镜像文件,即 **uImage**文件.
> 这里主要借助了**cfimager.exe** 这个工具, 有了上面的准备工作之后,烧录就变得很简单了,只需要执行下面这个烧录脚本:

```
@echo off
echo.
echo EasyARM-iMX28x 制作SD卡启动程序
echo.
echo 请输入SD卡盘符:
set /p diskpath=
set cmdpath=%~dp0
echo 注意:
echo 文件会被烧写在 %diskpath% 盘
echo.
%cmdpath%cfimager.exe -a -f %cmdpath%imx28_ivt_uboot.sb -e %cmdpath%rootfs.ext2.img -d %diskpath%
%cmdpath%cfimager.exe -raw -offset 0x100000 -f %cmdpath%uImage -d %diskpath%
%cmdpath%cfimager.exe -raw -offset 0x400 -f %cmdpath%zero.raw -d %diskpath%
echo 烧写完毕，按键退出
echo.
pause>nul
```

> 这里需要说明以下两点:

1. **zero.raw文件的烧录地址**：从uboot源码中可知(*见上面的代码注释*),uboot环境变量分区在SD卡从0开始的第 ***00x400*** 地址处,并且大小为127K, 所以zero.raw文件文件要烧录在 ***0x400*** 地址处.
2. **uImage文件的烧录地址** :为了给uboot环境变量保留足够的空间,我们把uImage文件烧录在 ***0x100000(1048576)*** 地址处,即第 ***0x800(2048)*** 个扇区处(每个扇区512字节, ***2048=1048576/512***),给uboot环境变量预留1M的大小.

设置uboot环境变量

> 把制作好的SD卡插入开发板SD卡插槽中,选择从SD卡启动,上电按任意键进入uboot,设置uboot的环境变量:

```
setenv bootargs 'gpmi=g console=ttyAM0,115200n8 root=/dev/mmcblk0p3 rw rootwait rootfstype=ext2 init=/sbin/init fec_mac= ethact mem=64M'
setenv sdcard_boot 'mmc read 0 $(loadaddr) 0x800 0x3000;bootm'
setenv bootcmd 'run sdcard_boot'
```

> 这里需要说明以下两点:
>
> 1. bootargs内核启动参数中的 **root=/dev/mmcblk0p3** 表示rootfs文件系统在内存卡中的哪个分区,这里是第3个分区,如果启动不了,可以把SD卡插入Linux电脑上,通过df命令查看sd卡的分区,如果查得rootfs在 **/dev/sdc3** 中,则这里就要设置为 **/dev/mmcblk0p3** .
> 2. mmc read命令用法:

```
mmc read <device_no> load_addr blk_no cnt
    device_no      mmc设备号, 0x00
    load_addr      把uImage文件读取到内存中指定的地址处(十六进制,0x41600000)
    blk_no         读取的起始扇区号(十六进制,0x800,注意这里是扇区号,而不是偏移地址)
    cnt            读取的扇区数(十六进制,0x3000=6M)
```

完