# 一、NXP官方Uboot编译与测试

 1、将NXP提供的uboot拷贝到ubuntu中。

 一个开发板也好运行uboot，DDR或者叫DRAM，串口，SD、EMMC、NAND。板子能工作。

 测似结果：

 1、uboot能正常启动

 2、LCD驱动要根据所使用的屏幕修改。

 3、NET初始化失败。

# 二、移植NXP官方uboot到ALPHA开发板

## 2.1 添加板子默认配置文件

 借鉴NXP官方6ULL EVK开发板，默认配置文件也用他的，

## 2.2添加板子对应的头文件

 不同的板子，有一些需要配置的信息，一般是在一个头文件里面配置，每个板子有一个。对于NXP官方的6ULL EVK板子，这个头文件就是

## 2.3 添加板子对应的板级文件夹

 每个板子都有特有的文件，也叫做板级文件。这里我们将6ULL EVK的板级文件直接拷贝过来。

## 2.4 修改uboot的配置界面

## 2.5 使用新添加的板子配置并编译Uboot

## 2.6 LCD驱动修改

 1、确定LCD IO初始化正确，mx6ull\_alientek\_emmc.c中的lcd\_pads。

 2、LCD参数，mx6ull\_alientek\_emmc.c中的displays。fb\_videomode表示RGB LCD参数。

 MX6ULL\_LCDIF1\_BASE\_ADDR (AIPS2\_OFF\_BASE\_ADDR + 0x48000)

 AIPS2\_OFF\_BASE\_ADDR (ATZ2\_BASE\_ADDR + 0x80000)

 ATZ2\_BASE\_ADDR AIPS2\_ARB\_BASE\_ADDR

AIPS2\_ARB\_BASE\_ADDR 0x02100000

 MX6ULL\_LCDIF1\_BASE\_ADDR **= 0**x02100000 + 0x80000 + 0x48000 = 0x21c8000

 Panel环境变量表示LCD ID。

## 2.7 网络驱动修改

 6ULL网络方案采用内部MAC+外部PHY，6ULL官方开发板使用的PHY芯片就是KSZ8081。正点原子的ALPHA开发板没有使用KSZ8081，我们使用的LAN8720A。因此要修改驱动。

 LAN872有一个管理接口，叫做MDIO，两根线，MDIO和MDC，一个MDIO接口可以管理32个PHY芯片。MIDO通过PHY ADDR来确定访问那个PHY芯片。ALPHA开发板ENET1的PHY ADDR是0x0，ENET2的PHY ADDR是0X1.

 每个LAN8720都有一个复位引脚，ENET1是SNVS\_TAMPER7，ENET2是SNVS\_TAMPER8。

 LAN8720驱动，因为所有的PHY，其前32个寄存器一模一样，因此uboot里面已经写好了通用PHY驱动，所以理论上不需要修改。

 驱动修改

 1、修改PHY ADDR

 2、删除原有的74LV595相关代码。

 3、添加ALPHA开发板的网络复位IO

## 2.8 其他需要修改的地方

# 三、bootcmd和bootargs环境变量

## 3.1 bootcmd环境变量

 宏CONFIG\_BOOTCOMMAND也可以设置bootcmd的值。

 "run findfdt;" \

 "mmc dev ${mmcdev};" \

 "mmc dev ${mmcdev}; if mmc rescan; then " \

 "if run loadbootscript; then " \

 "run bootscript; " \

 "else " \

 "if run loadimage; then " \

 "run mmcboot; " \

 "else run netboot; " \

 "fi; " \

 "fi; " \

 "else run netboot; fi"

 Findfdt 设置fdt\_file环境变量，也就是dtb文件名字。

mmc dev 1 //切换到emmc

fatload mmc 1:1 80800000 zImage

fatload mmc 1:1 83000000 imx6ull-14x14-evk.dtb

booz 80800000 - 83000000

 "loadbootscript=" \

 "fatload mmc ${mmcdev}:${mmcpart} ${loadaddr} ${script};\0" \

 展开以后：fatload mmc 1:1 80800000 boot.scr

loadimage=fatload mmc ${mmcdev}:${mmcpart} ${loadaddr} ${image}\0" \

 展开：fatload mmc 1:1 80800000 zImage

"loadfdt=fatload mmc ${mmcdev}:${mmcpart} ${fdt\_addr} ${fdt\_file}\0" \

 展开

fatload mmc 1:1 83000000 imx6ull-14x14-evk.dtb

bootz ${loadaddr} - ${fdt\_addr};

booz 80800000 - 83000000

## 3.2 bootargs环境变量

 宏CONFIG\_BOOTARGS也可以设置bootargs的值

mmcargs=setenv bootargs console=${console},${baudrate} " \

 CONFIG\_BOOTARGS\_CMA\_SIZE \

 "root=${mmcroot}\0" \

展开以后就是：

bootargs console= ttymxc0,115200 root=/dev/mmcblk1p2

Bootargs是会传递给Linux内核，设置了一些东西

Bootargs环境变量也叫做命令行参数。

# 四、uboot启动Linux测试

## 4.1 从EMMC启动

 1、首先查看EMMC里面是否有系统，linux镜像zImage和.dtb文件。先将当前设备切换到EMMC：

mmc dev 1 //切换到EMMC

fatls mmc 1:1 //查看EMMC分区1里面的文件

fatload mmc 1:1 80800000 zImage //将zimage下载到DDR的0x80800000处

fatload mmc 1:1 83000000 imx6ull-14x14-emmc-7-1024x600-c.dtb //将dtb读取到0X83000000

bootz 80800000 – 83000000 //启动内核

 如果内核启动成功，说明uboot支持emmc启动，验证成功。

## 4.2 从网络启动

 tftp 80800000 zImage //从tftp服务器下载zimage

 tftp 83000000 imx6ull-14x14-emmc-7-1024x600-c.dtb //从tftp服务器下载.dtb

 bootz 80800000 – 83000000 //启动系统

# 五、uboot DDR初始化

##  1、裸 机

 imxdownload软件下载，会在bin文件头部添加IVT DCD数据，

##  2、uboot

 uboot编译生成u-boot.imx。u-boot.imx已经包含了IVT DCD数据。

 u-boot.imx的头部信息是怎么添加的？

 u-boot.imx的DCD中的DDR初始化代码该怎么修改。

 uboot编译会输出

 ./tools/mkimage -n board/freescale/mx6ull\_alientek\_emmc/imximage.cfg.cfgtmp -T imximage -e 0x87800000 -d u-boot.bin u-boot.imx

 可以看出uboot使用/tools/mkimage工具，向u-boot.bin添加board/freescale/mx6ull\_alientek\_emmc/imximage.cfg.cfgtmp文件信息，从而得到u-boot.imx。

 默认只有imximage.cfg文件，imximage.cfg里面保存的就是DCD数据。DDR初始化也此文件里面。

 我们要修改DDR初始化代码，就需要修改imximage.cfg文件。此文件默认拷贝的NXP给IMX6ULL EVK开发板写的，默认是给512MB DDR3L写的。