这个 u-boot 老版本和新版本有很大的不同，我只接触过 2012.10 版的，你跟踪一下代码从 start.S 看最后跳到了什么位置。下面我说下 arm 的 logo 显示：

start.S --> /arch/arm/lib/board.c

在 board.c 这个文件里关注 board_init_r，也就是当代码在 RAM 中的初始化函数，它调用 stdio_init 初始化输入输出设备。当然，屏的初始化也会放在这里……

board.c --> /common/stdio.c

这个文件直接就看 stdio_init 函数，这里关于屏的有两个调用：一是“drv_lcd_init”,这个没研究过;二是"drv_video_init"，这里可实现 logo，控制台的屏幕输出等。

stdio.c --> /drivers/video/cfb_console.c

进入 drv_video_init 这个函数后，会调用本文件的 video_init 函数。

video_init 函数会调用 video_hw_init 函数，这个函数需要自己在 farmbuffer 驱动中实现。然后，跟着往下看时有这样一个编译控制宏“CONFIG_VIDEO_LOGO”,如果定义会使本文件的 video_logo 函数被调用，这样你的 logo 就会出现了。

在 video_logo 中，宏“CONFIG_SPLASH_SCREEN_ALIGN”是控制显示位置的，默认情况下 logo 是显示在屏的左上角，你可以坐标形式给出 logo 的位置，也可以简单的使用参数“m”，让 logo 显示在屏的中央。

在 video_logo 中，宏“CONFIG_SPLASH_SCREEN”，其控制之下有个函数调用“video_display_bitmap”，这个函数的第一个参数是从 uboot 环境变量中取出的 logo 图片所在的内存地址。其默认值为“/tools/easylogo/linux_logo.tga”。详细情况，请参考 u-boot README。

另外说下配置宏：

```
#define CONFIG_VIDEO                    //必需，开启 video 设备
#define CONFIG_VIDEO_S3C64X0            //必需，需要自己实现，你 LCD 设备的 video 驱动
#define LCD_VIDEO_ADDR 0x56000000  //非必需，这里是因为我的 video 驱动中使用了这个宏
#define CONFIG_SYS_VIDEO_VCLOCK_HZ (133000000) //非必需，这里因为我的 video 驱动中使用了这个宏
/* for logo bmp */
#define CONFIG_CMD_BMP                  //非必需，看你的源图咯
#define CONFIG_VIDEO_LOGO               //必需，logo 显示与否的开关
#define CONFIG_SPLASH_SCREEN            //必需，真正的作图函数在这个宏控制之下
```

另外，如果要将 u-boot 控制台输出放到屏上的话可以加以下控制宏

```
/* for cfb_console */
#define CONFIG_CFB_CONSOLE              //必需，开启 LCD 作为控制台输出
#define CONFIG_VIDEO_SW_CURSOR          //非必需，是否显示控制台光标
#define VIDEO_FB_16BPP_WORD_SWAP        //非必需，是否开启字符显示的半字交换
```

最后，如果为 arm 架构，由于 4 字节对齐，所以对于 8bbp, 16bbp, 24bbp 需要跟据需要开启半字换或字交换，否则显示图片模糊或有虚影。

如果楼上实在无解的话，我给你个最简单，但不建议的方法：

首先，你肯定是要有 LCD 驱动的。然后将你的 logo 做成与屏分辨率一样大，再转换成数组形式。最后，找到这个驱动文件，找一下它的 farmbuffer 的地址(肯定有的)，在它显存地址设置后一句 memcpy 将数组复制到显存就好了。对于支持虚拟屏的，手动调整一下其实际显示的位置为你 memcpy 的那一段。

这样做最大的不好就是，你的 bootloader 将变得很大，难于更换 logo。另外也不科学，所以不推荐。