BeagleBone Black eMMC 烧写全记录(基于 AM335x SDK06)

eMMC 存储介质目前越来越广泛的应用在嵌入式系统中，AM335x 的用户也越来越多的使用 EMMC 作为系统的主要存储介质。目前 AM335x 的几款官方 demo 板中，只有 BeagleBone Black 上加入了对 eMMC 芯片的支持，很多用户也是参考 BeagleBone Black 进行自己 AM335x 系统的 eMMC 设计。笔者最近分别通过 TI Uniflash 和 SD 卡完成了 BeagleBone Black 上 eMMC 芯片的烧写验证工作，软件基于 AM335x Linux SDK06，总结出来供大家参考。

1. 使用 TI UniFlash 工具通过 USB RNDIS 烧写

1.1 TI Uniflash 简介

Uniflash 是 TI 开发的存储器烧写工具，可以支持 AM335x 系统的 NAND Flash，NOR Flash, SPI Flash，eMMC 烧写。可以参考 wiki 上的 guide：http://processors.wiki.ti.com/index.php/Sitara_Uniflash_Quick_Start_Guide，在 Windows 宿主机上下载并安装 Uniflash，并按照其中 3.3 节所述在 Windows 宿主机上安装 USB RNDIS 驱动。

1.2 eMMC 烧写原理

本文介绍的验证方法是使用 Uniflash 工具通过 USB 对 BeagleBone Black 上的 eMMC 进行烧写，原理是通过 Romcode，SPL 和 u-boot 三个阶段，将一个专门用于 eMMC 烧写的 Linux 操作系统在 BeagleBone Black 板上运行起来，并自动运行脚本进行烧写。

第一阶段，通过设置 AM335x 的 sysboot 管脚，使 AM335x 的启动项包含 USB0 启动。BeagleBone Black 的默认启动设置为：MMC1->MMC0->UART0->USB0，MMC1 和 MMC0 分别连接了 eMMC 和 SD 卡，如果 eMMC 为空，并且不插 SD 卡，芯片上电后执行的 Romcode 就会执行 USB 启动。Romcode 会初始化 USB RNDIS 以太网络通信协议，通过 Windows 宿主机上 Uniflash 自带的 DHCP 服务器进程拿到分配的 IP 地址，然后再通过 Windows 宿主机上 Uniflash 自带的 TFTP 服务器进程将 Uniflash 设定的 tftp 目录下用于 eMMC 烧写的特殊的 SPL 下载到 AM335x 的内部 ram 中并运行。

第二阶段，用于 eMMC 烧写的特殊的 SPL 执行之后，会初始化 USB RNDIS 以太网络通信协议，并通过 Windows 宿主机上 Uniflash 自带的 DHCP 服务器程序拿到分配的 IP 地址，然后再通过宿主机上 Uniflash 自带的 TFTP 服务器程序将 Uniflash 设定的 tftp 目录下用于 eMMC 烧写的特殊的 U-boot.img 下载到板子上的 DDR3 中并运行。

第三阶段，用于 eMMC 烧写的特殊的 U-boot.img 执行之后，通过执行 u-boot 中的 DHCP 和 TFTP 命令，将 Windows 宿主机上 Uniflash 设定的 tftp 目录下的用于烧录 eMMC 的 Linux uImage 下载到板子的 DDR3 中并运行。这个 Linux uImage 会通过执行启动脚本，

通过 tftp 的方式，将 Windows 宿主机上 Uniflash 设定的 tftp 目录下名为 debrick.sh 脚本下载并执行，debrick.sh 可实现 eMMC 的擦除和烧写。

1.3 eMMC 烧写流程

1.3.1 制作 u-boot-spl-restore.bin 和 u-boot-restore.img

用于烧录 eMMC 的特殊的 SPL 和 u-boot.img 可以通过指定特定的编译参数编译 SDK06 的 U-boot 源码获得，在编译之前需要加几个 patch 如下：

1) 延长 USB RNDIS 连接等待时间的 patch:
   修改/drivers/usb/gadget/ether.c 文件：

```
index de880ff..926c6f2 100644
--- a/drivers/usb/gadget/ether.c
+++ b/drivers/usb/gadget/ether.c
@@ -108,7 +108,11 @@ static const char driver_desc[] = DRIVER_DESC;
                                |USB_CDC_PACKET_TYPE_PROMISCUOUS \
                                |USB_CDC_PACKET_TYPE_DIRECTED)

+#if defined(CONFIG_RESTORE_FLASH) && defined(CONFIG_SPL_USBETH_SUPPORT)
+#define USB_CONNECT_TIMEOUT (15 * CONFIG_SYS_HZ)
+#else
 #define USB_CONNECT_TIMEOUT (3 * CONFIG_SYS_HZ)
+#endif
```

2) 使能 U-boot 中 cache 用于加快传输时间的 patch:
   修改/include/configs/am335x_evm.h 文件：

```
index 90e35ee..efa1e90 100644
--- a/include/configs/am335x_evm.h
+++ b/include/configs/am335x_evm.h
@@ -236,6 +236,7 @@

 #define CONFIG_CMD_FAT
 #define CONFIG_FAT_WRITE
 #define CONFIG_CMD_EXT2
+#define CONFIG_CMD_CACHE
```

3) 修改bootcmd启动参数，在/include/configs/am335x_evm.h 中182行修改如下：

```
#ifdef CONFIG_SPL_USBETH_SUPPORT
#define CONFIG_BOOTCOMMAND \
"dcache on; " \
"setenv autoload no; " \
"setenv ethact usb_ether; " \
"dhcp; "            \
"if tftp 81000000 uImage; then "            \
```

```
"bootm 81000000; "      \
"fi"
#else
```

加入以上 3 个 patch 之后，在 u-boot 源码目录下输入以下编译指令：

```
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- O=flash-restore
am335x_evm_restore_flash_usbspl
```

在 U-boot 源码目录下的 boards.cfg 文件中，可以看到 am335x_evm_restore_flash_usbspl 这个编译选项设定了宏 RESTORE_FLASH，SPL_USBETH_SUPPORT。

生成的 u-boot.img 和 u-boot-spl.bin 分别在在 flash-restore\和 flash-restore\spl 目录下，要将其改为 u-boot-restore.img 和 u-boot-spl-restore.bin 并放置 Windows 宿主机上 Uniflash 设定的 tftp 目录下。

1.3.2 制作 uImage

用于烧录 eMMC 的 Linux uImage，以 Initcramf 的方式包含了一个用于烧录的 Ramdisk 文件系统，下面是具体配置和生成的步骤。

在 SDK06 的 Linux 源码目录下，执行：

```
make CROSS_COMPILE=arm-linux-gnueabihf- ARCH=arm tisdk_am335x-evm_defconfig

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

1) 配置内核所包含的文件系统目录。

将用于烧录的 Ramdisk 文件系统解压到虚拟机的某个目录下，如 /home/zhoujian/Flasher。

进入 general setup 目录，按图 1-1 配置 Initcramfs 所在目录

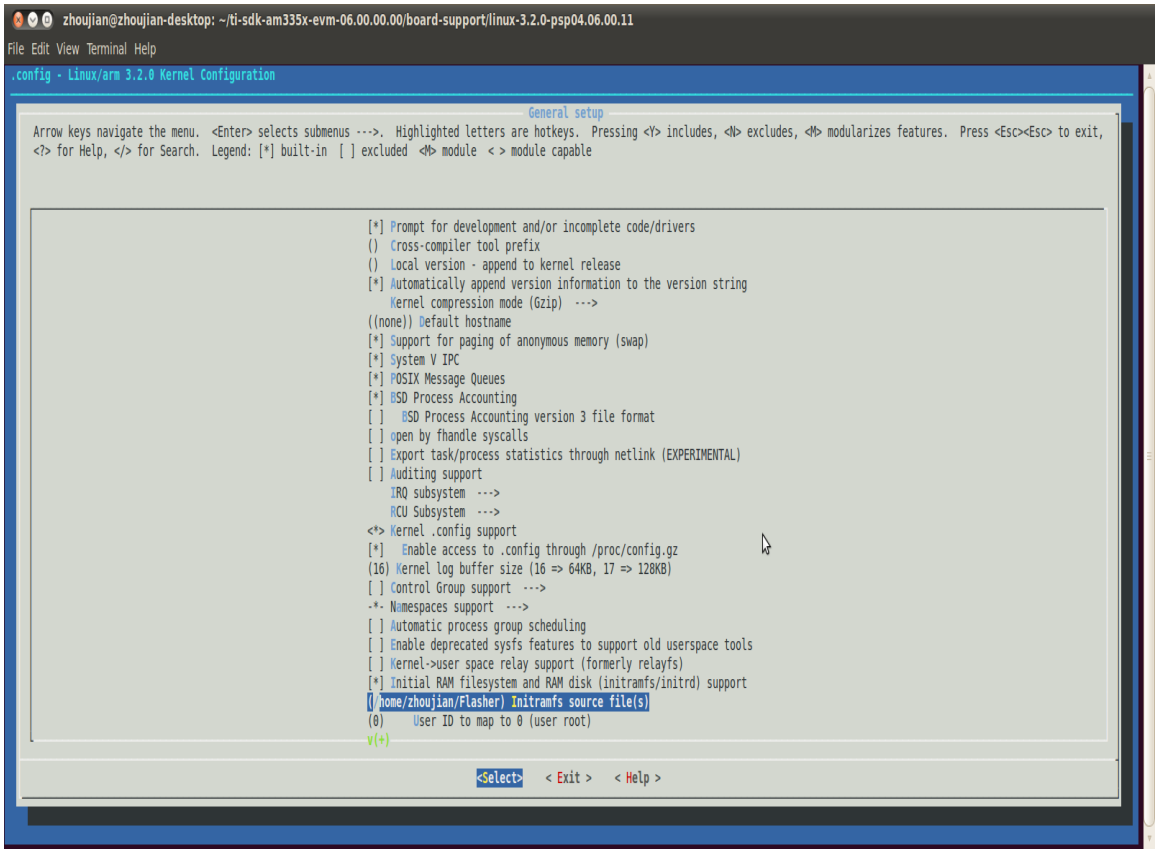

图 1-1

2) 进入 Device Drivers，按照图 1-2，图 1-3，图 1-4 和图 1-5 配置 kernel 对 USB RNDIS 的支持

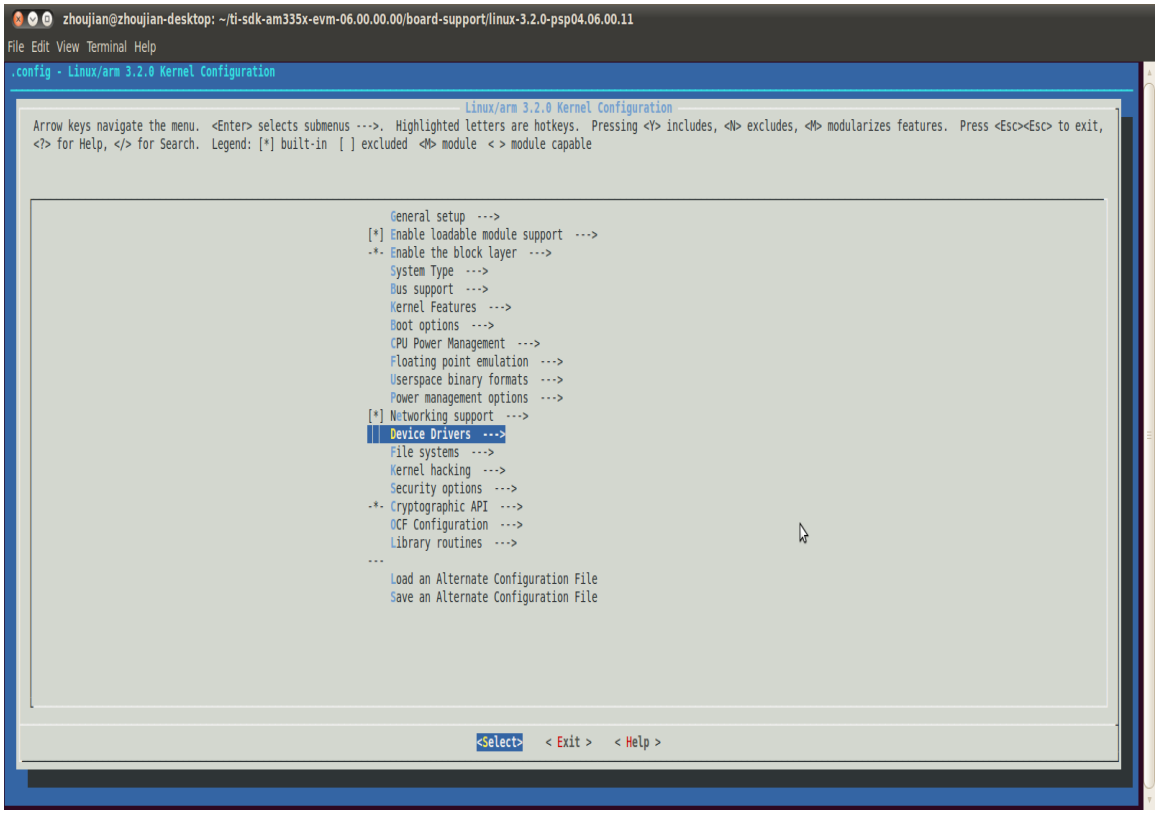

图 1-2 选择进入 Device Drivers

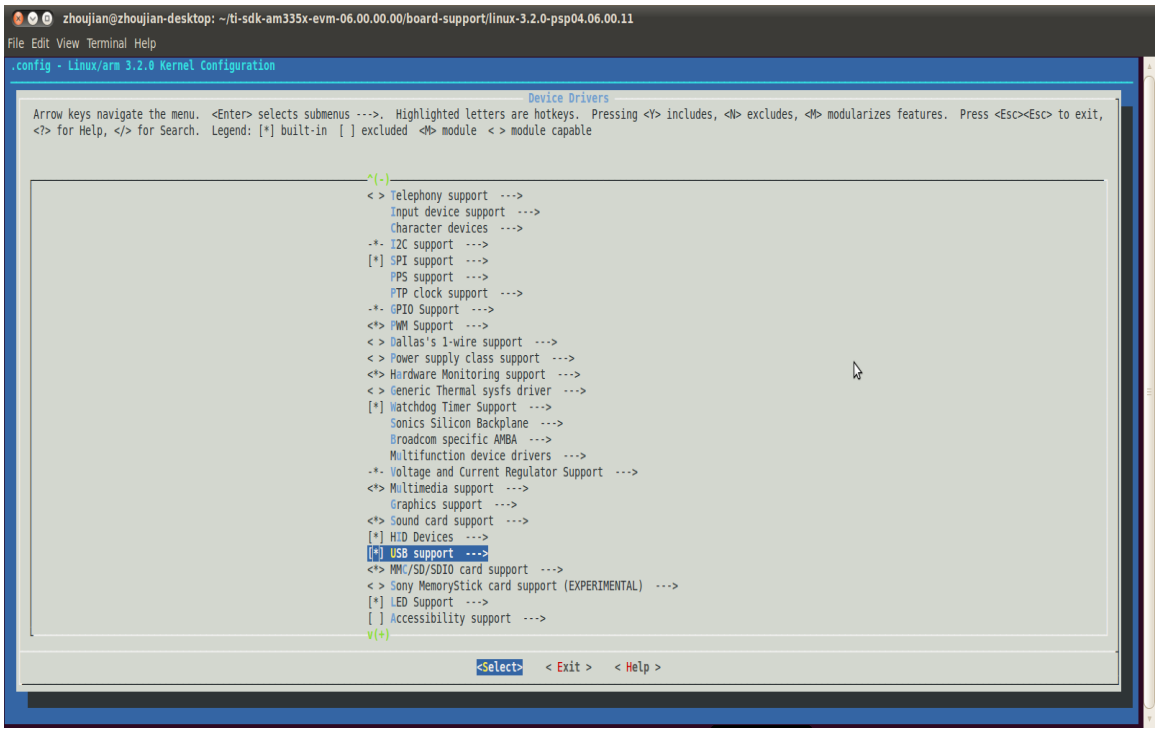

图 1-3 选择进入 USB support

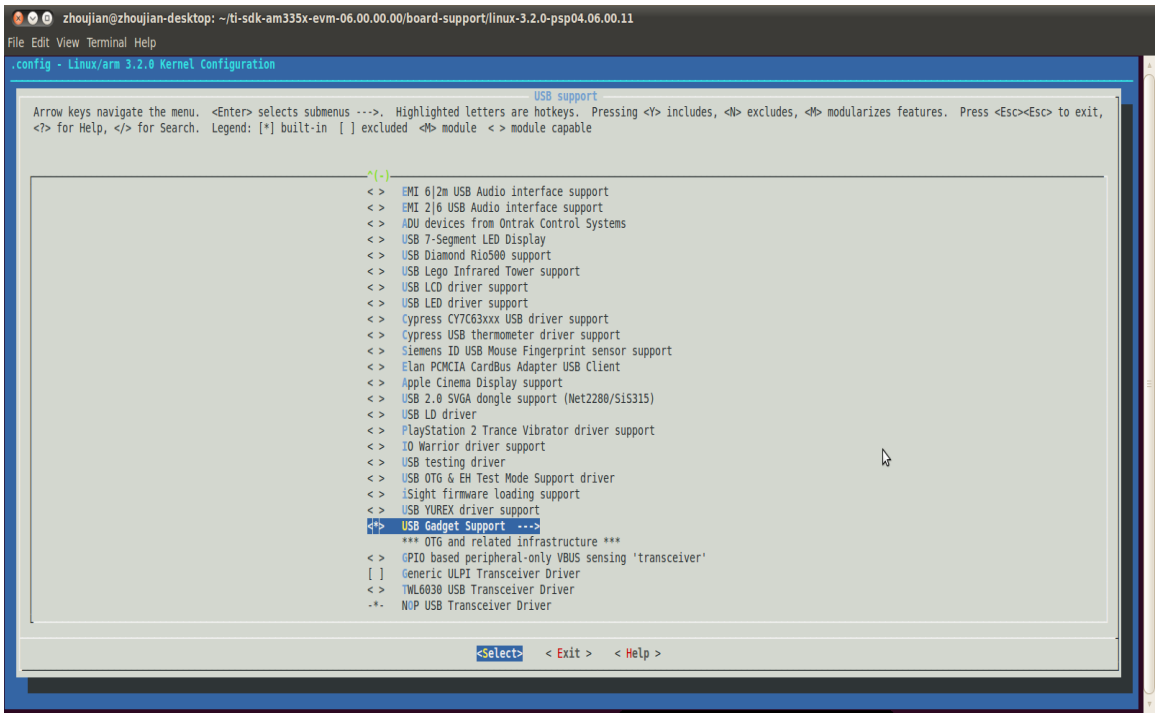

图 1-4 选择进入 USB Gadget support

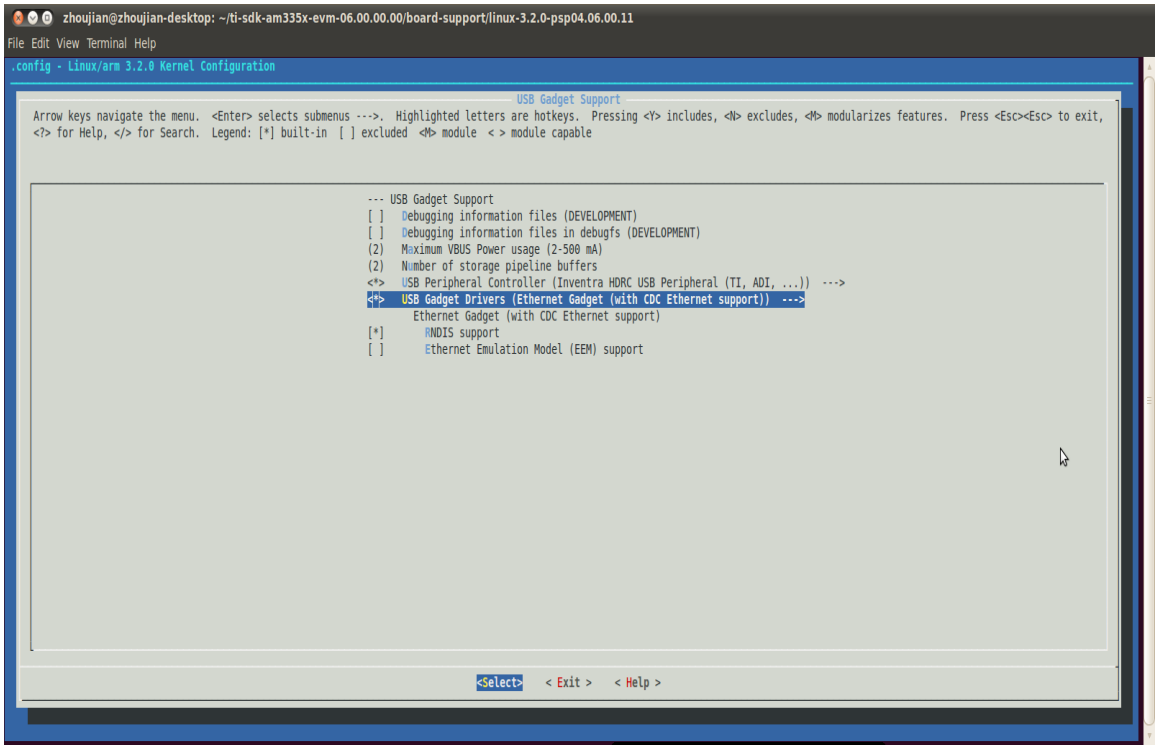

图 1-5 在 USB Gadget Drivers 中，只保留 Ethernet Gadget(with CDC Ethernet support)和 RNDIS support

保存退出后，执行：

make CROSS_COMPILE=arm-linux-gnueabihf- ARCH=arm uImage

将生成的\arch\arm\boot\uImage 放置 Windows 宿主机上 Uniflash 设定的 tftp 目录下。

如果对速度有要求，可在 menuconfig 的界面中，选择[*Device Drivers*]->[*Network device support*]->[*Ethernet driver support*]，如图 1-6 所示，将 kernel 中的 CPSW 驱动去掉，从而停掉系统启动后的以太网 DHCP 脚本的执行，以节省时间。

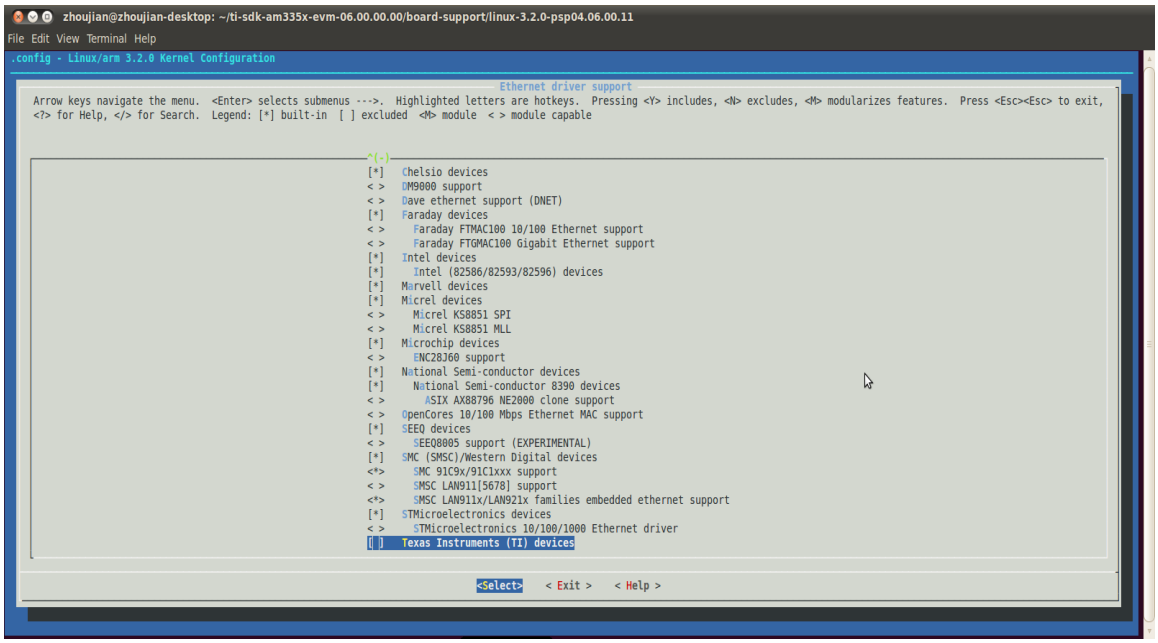

图 1-6

1.3.3 进行烧写

将 eMMC 烧录脚本 debrick.sh 放置 Windows 宿主机上 Uniflash 设定的 tftp 目录下。

将要烧写到 eMMC 中的 MLO，u-boot.img 和 uImage 压缩成 debrick.sh 脚本指定的 boot_partition.tar.gz：

tar –cvzf boot_partition.tar.gz MLO u-boot.img uImage

将要烧写到 eMMC 中的文件系统压缩文件名改为 rootfs_partition.tar.gz，和 boot_partition.tar.gz 一起放置 Windows 宿主机上 Uniflash 设定的 tftp 目录下。

在 Windows 上运行 Uniflash，可以按图 1-7 所示指定 tftp 目录为 C:\AM335x_Flashtool\images

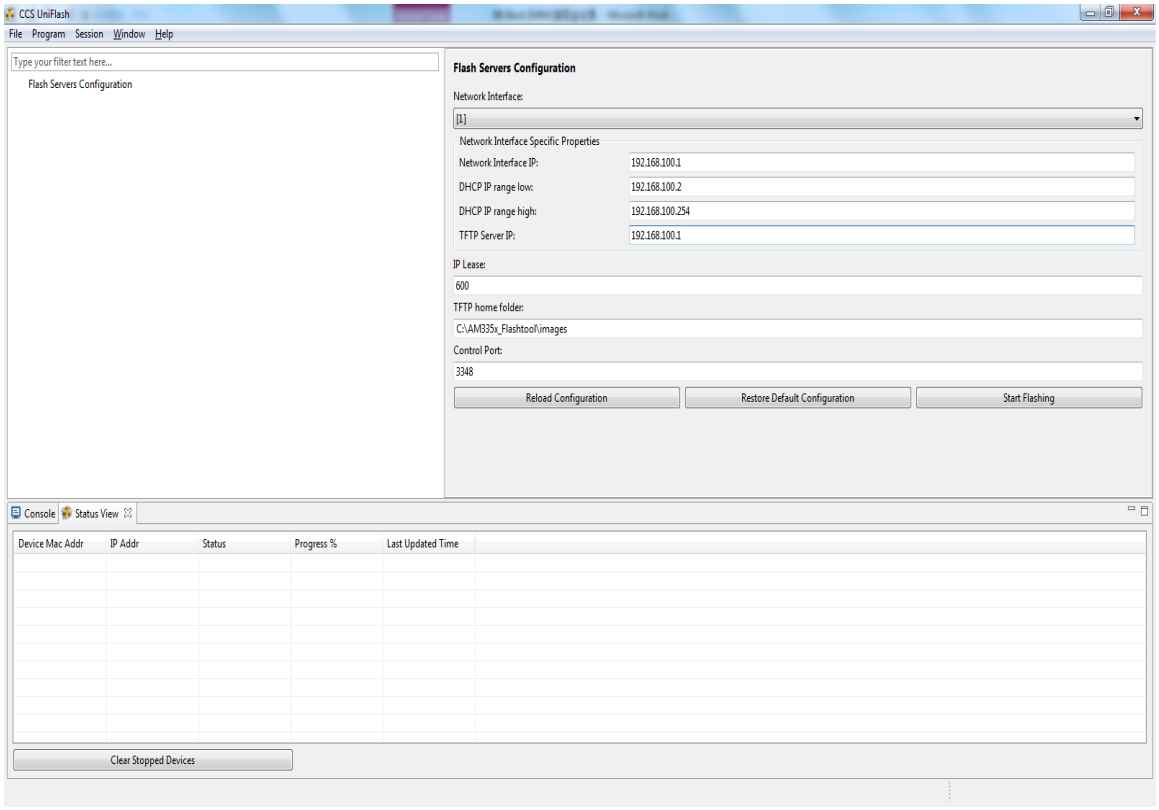

图 1-7

点击 Start Flashing 按钮，通过 USB 线连接 BeagleBone Black 和宿主机，eMMC 的烧写可自动完成。

2. 使用 SD 卡烧写

BeagleBone Black 上带有 SD 卡，也可以通过 SD 卡进行 eMMC 的烧写，可将 SDK06 自带的 prebuild image 中的 MLO，u-boot.img，uImage 以及上一章提到的用于烧写 eMMC 的文件系统烧写到 SD 卡中，然后将 SD 卡中的内容通过 Ubuntu 虚拟机显示出来，并对此文件系统中的内容做如下改动。

1) 将 SD 卡上文件系统中的/etc/init.d/fetcher.sh 改为：

```
DEBRICK_SCRIPT="/home/root/debrick.sh"
chmod 777 ${DEBRICK_SCRIPT}

echo ""
echo "****************************************"
echo "Sitara Flash Fetcher is complete. Executing ${DEBRICK_SCRIPT}."
echo ""

## Execute script
```

./${DEBRICK_SCRIPT}

2) 将 debrick.sh 脚本拷贝到 SD 卡上文件系统中的/home/root/目录下，并将其中 tftp 传输的部分去掉：

```
##echo "Getting files from server: ${SERVER_IP}"
##time tftp -b 4096 -g -r ${BOOT_PARTITION} ${SERVER_IP} &
##boot_pid=$!
## time tftp -b 4096 -g -r ${ROOTFS_PARTITION} ${SERVER_IP} &
## rootfs_pid=$!
```

3) 将要烧写到 eMMC 中的 MLO，u-boot.img 和 uImage 压缩成 debrick.sh 脚本指定的 boot_partition.tar.gz：

```
tar –cvzf  boot_partition.tar.gz MLO u-boot.img uImage
```

将要烧写到 eMMC 中的文件系统压缩文件名改为 rootfs_partition.tar.gz，和 boot_partition.tar.gz 一起拷贝到 SD 卡上文件系统中的/home/root/目录下

在给 Beaglebone Black 上电之前，插上制作好内容的 SD 卡，按住板子的 S2 开关，然后通过 USB 线连接 BeagleBone Black 和宿主机给板子上电，会自动执行 eMMC 的烧写。

3. 总结和参考文档

Linux 系统从 eMMC 上启动，需要修改 u-boot 的 mmcdev 参数，将其由默认的 0 改为 1，也可以直接修改/include/configs/am335x_evm.h 中的 73 行为"mmcdev=1\0" \，然后重新编译。

第一次烧写，可以从板子的 J1 插针引出 UART0 信号到主机超级终端查看烧写流程。

如果用户是自己做的板子，那么也可以在针对自己板子移植好的 U-boot 和 Linux 源代码的基础上，按照上面所讲的加入对 eMMC 烧写的支持。

对于 Linux SDK 07，操作步骤基本类似，未来会继续验证。

本文主要参考下述文档：

http://processors.wiki.ti.com/index.php/Sitara_Linux_AM335x_Flash_Programming_Linux_Development

http://processors.wiki.ti.com/index.php/Sitara_Linux_Program_the_eMMC_on_Beaglebone_Black

http://www.deyisupport.com/question_answer/dsp_arm/sitara_arm/f/25/t/52381.aspx