

# 使用 JLink 间接烧写 S3C2410、S3C2440 开发板 Nor、Nand Flash 的方法

## 1. 简要说明

JLink 的调试功能、烧写 Flash 的功能都很强大，但是对于 S3C2410、S3C2440 的 Flash 操作有些麻烦：烧写 Nor Flash 时需要设置 SDRAM，否则速率很慢；烧写 Nand Flash 只是从理论上能够达到，但是还没有人直接实现这点。

本文使用一个间接的方法来实现对 S3C2410、S3C2440 开发板的 Nor、Nand Flash 的烧写。原理为：JLink 可以很方便地读写内存、启动程序，那么可以把一个特制的程序下载到开发板上的 SDRAM 去，并运行它，然后使用这个程序来烧写。

## 2. 操作步骤

### 2.1 连接硬件

对于大多数的 S3C2410、S3C2440 开发板而言，它们所用的 JTAG 接口一般有 3 种(如图 1 所示)，其中前两种用得比较多。



图 1. 三种常用的 JTAG 接口

但是市面上的 JLink，大多只支持第 3 种 JTAG 接口，所以需要用到转接板。或者直接使用 JLink 的变种，如图 2 所示的两种改进版 JLink：

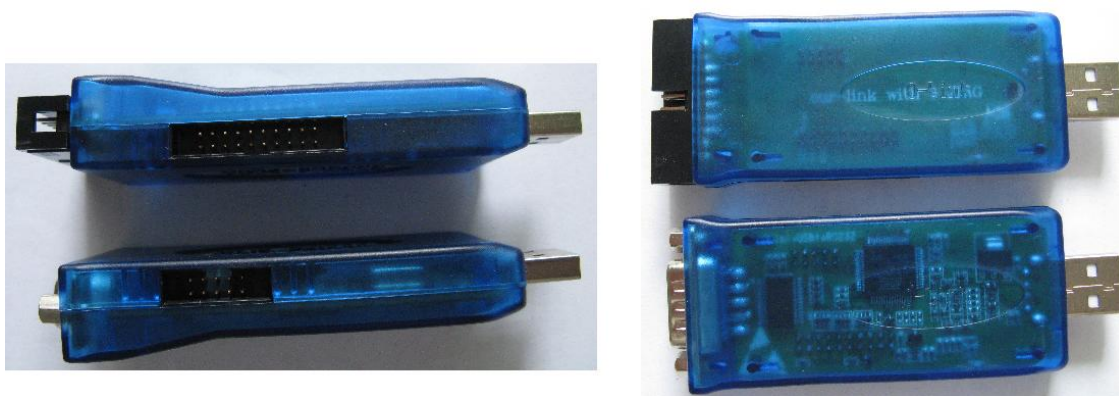


图 2. 两种改进版 JLink

以 mini2440 为例，如图 3 接好 JTAG 线。

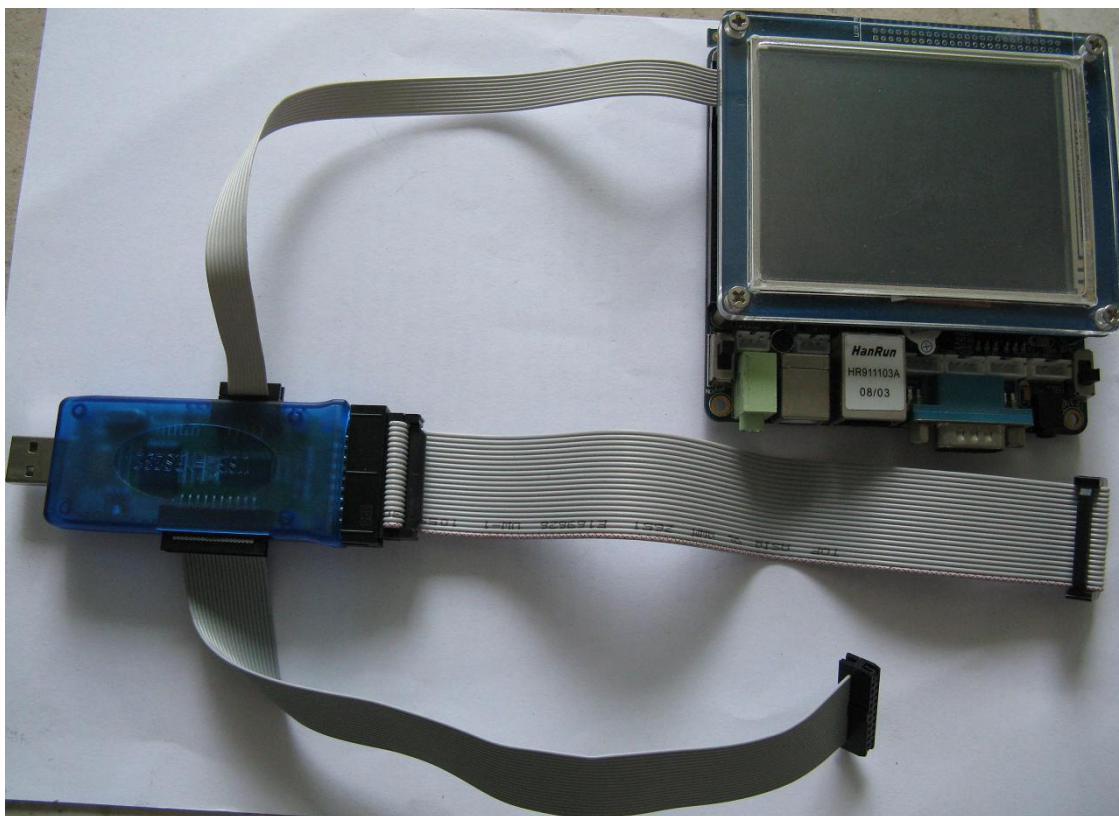


图 3. 改进版 JLink 与 mini2440 的连接

## 2.2 运行 J-Link commander

J-Link commander 启动界面如图 4 所示，(如果没有发现检测到 CPU，就在里面执行 usb 命令连接 JLink，再执行 r 命令识别处理器)。



图 4. J-Link commander 启动界面

### 2.3 下载运行特制的程序

对于 S3C2410、S3C2440 处理器，它们内部有 4K 的 SRAM，当使用 Nor Flash 启动时，地址为 0x40000000；当使用 Nand Flash 启动时，地址为 0。

对于 S3C2410、S3C2440 开发板，一般都外接 64M 的 SDRAM。SDRAM 能被使用之前，需要经过初始化。

所以，先把一个 init.bin 下载到内部 SRAM 去运行，它执行 SDRAM 的初始化；然后再下载一个比较大的程序，比如 u-boot 到 SDRAM 去运行，它将实现对 Nor、Nand Flash 的操作。

以下是在 J-Link commander 里的命令，假设 init.bin、u-boot.bin 在 e:盘下。

1. speed 12000 //设置 TCK 为 12M，下载程序时会很快

2. 下载并运行 init.bin，这是用来初始化 SDRAM 的

2.1 如果是 NAND 启动：

```
loadbin e:\init.bin 0
```

```
setpc 0
```

```
g
```

2.2 如果是 Nor 启动：

```
loadbin e:\init.bin 0x40000000
```

```
setpc 0x40000000
```

```
g
```

3. 下载特制的 uboot：

```
h
```

```
loadbin e:\u-boot.bin 0x33f80000
```

```
setpc 0x33f80000
```

```
g
```

上述操作命令可以参考图 5。

```

快捷方式 到 快捷方式 到 J-Link Commander
JTAG speed: 5 kHz
J-Link>speed 12000
JTAG speed: 12000 kHz
J-Link>loadbin e:\init.bin 0
Loading binary file... [e:\init.bin]
Writing bin data into target memory @ 0x00000000.
J-Link>setpc 0
J-Link>g
J-Link>h
Info: J-Link: ARM9 CP15 Settings changed: 0xC000007A from 0x00000078, MMU Off, I
Cache Off, DCache Off
PC: <R15> = 000000B8, CPSR = 200000D3 <SUC mode, ARM FIQ dis, IRQ dis.>
R0 = 0007C608, R1 = C000007A, R2 = 56000010, R3 = FFFFFFFF
R4 = 00000002, R5 = 56000014, R6 = EFFFFFFF, R7 = E3A00000
USR: R8 =EAFFFFC, R9 =E2401061, R10=E3510019, R11 =8A000001, R12 =E3A00001
R13=E1A0F00E, R14=E2400041
FIQ: R8 =EFCABD17, R9 =48B97FDB, R10=D4EBA086, R11 =63CDF7E4, R12 =D7FEBBF3
R13=BF776DFF, R14=B4F7DBE7, SPSR=40000010
SUC: R13=33F4DAA4, R14=000000A0, SPSR=F00000DF
ABT: R13=77DB6B97, R14=9F445F75, SPSR=0000003C
IRQ: R13=33F4FF7C, R14=33F8B978, SPSR=80000053
UND: R13=DDEC3FDE, R14=DFDDEF2, SPSR=00000010
J-Link>loadbin e:\u-boot.bin 0x33f80000
Loading binary file... [e:\u-boot.bin]
Writing bin data into target memory @ 0x33f80000.
J-Link>setpc 0x33f80000
J-Link>g
J-Link>_

```

图 5. 使用 jlink 下载运行 u-boot

现在，u-boot 已经启动了，在串口工具上可以看到如图 6 所示界面，以后就可以通过网络、串口下载文件，然后使用 u-boot 里的命令进行烧写。

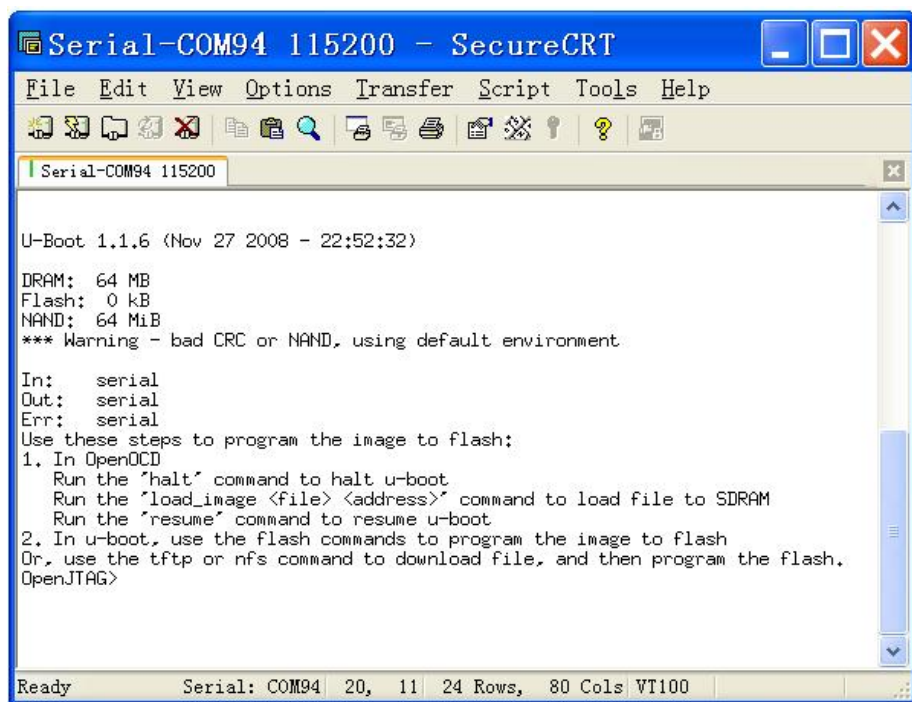


图 6. u-boot 启动界面

当然，如果没有网络，也不想忍受串口的速率，也可以通过 jlink commander 下载，比如：

h

loadbin your\_file.bin 0x30000000

g

这时，你的文件已经被下载到 SDRAM 0x30000000 去了。

后面的操作就是 u-boot 的命令了。

#### 2.4 使用 u-boot 烧写 Flash

以例子为例，假设需要烧写一个名为 leds.bin 的程序到 Nor、Nand Flash，那么请参考：

##### (1). 通过 Jlink 下载：

在 J-Link commander 里执行：

h

loadbin e:\leds.bin 0x30000000

g

注意 leds.bin 的大小

##### (2). 通过 u-boot 烧写到 Nor Flash：

在 u-boot 里执行：

---

```
protect off all      // 解锁
erase 0 2fff         // 擦除从 0 地址开始的大小为 0x30000 的 NOR Flash 扇区(大小为可
擦除块的整数倍, 可以运行 flash info 命令查看)
cp.b 0x30000000 0 30000 // 把前面下载到 0x30000000 的程序烧写到 NOR 去
```

(3). 通过 u-boot 烧写到 Nand Flash:

在 u-boot 里执行:

```
nand erase 0 30000      // 擦除从 0 地址开始的大小为 0x30000 的 Nnad Flash 扇区
nand write.jffs2 30000000 0 30000 // 把前面下载到 0x30000000 的程序烧写到 Nand 去
```

注意, 上面用的 2fff、30000 等数字是 192K, 如果你的程序比较小, 请自行设置。