【详解】嵌入式开发中固件的烧录方式

版本：v1.2

Crifan Li

摘要

本文主要介绍了嵌入式开发过程中，将固件从PC端下载到开发板中的各种方式，主要包括NFS挂载，Nand Flash和Nor Flash，USB，RS232，网卡NIC等方式。

本文提供多种格式供：

在线阅读	HTML [1]	HTMLs [2]	PDF [3]	CHM [4]	TXT [5]	RTF [6]	WEBHELP [7]
下载（7zip压缩包）	HTML [8]	HTMLs [9]	PDF [10]	CHM [11]	TXT [12]	RTF [13]	WEBHELP [14]

HTML版本的在线地址为：

http://www.crifan.com/files/doc/docbook/firmware_download/release/html/firmware_download.html

有任何意见，建议，提交bug等，都欢迎去讨论组发帖讨论：

http://www.crifan.com/bbs/categories/firmware_download/

修订历史		
修订 1.2	2013-12-15	crl
1. 通过Docbook发布 2. 添加了xml:id，添加了一些细节提示		
修订 1.0	2011-08-25	crl
1. 解释了嵌入式开发中的固件下载方式		

[1] http://www.crifan.com/files/doc/docbook/firmware_download/release/html/firmware_download.html
[2] http://www.crifan.com/files/doc/docbook/firmware_download/release/htmls/index.html
[3] http://www.crifan.com/files/doc/docbook/firmware_download/release/pdf/firmware_download.pdf
[4] http://www.crifan.com/files/doc/docbook/firmware_download/release/chm/firmware_download.chm
[5] http://www.crifan.com/files/doc/docbook/firmware_download/release/txt/firmware_download.txt
[6] http://www.crifan.com/files/doc/docbook/firmware_download/release/rtf/firmware_download.rtf
[7] http://www.crifan.com/files/doc/docbook/firmware_download/release/webhelp/index.html
[8] http://www.crifan.com/files/doc/docbook/firmware_download/release/html/firmware_download.html.7z
[9] http://www.crifan.com/files/doc/docbook/firmware_download/release/htmls/index.html.7z
[10] http://www.crifan.com/files/doc/docbook/firmware_download/release/pdf/firmware_download.pdf.7z
[11] http://www.crifan.com/files/doc/docbook/firmware_download/release/chm/firmware_download.chm.7z
[12] http://www.crifan.com/files/doc/docbook/firmware_download/release/txt/firmware_download.txt.7z
[13] http://www.crifan.com/files/doc/docbook/firmware_download/release/rtf/firmware_download.rtf.7z
[14] http://www.crifan.com/files/doc/docbook/firmware_download/release/webhelp/firmware_download.webhelp.7z

【详解】嵌入式开发中固件的烧录方式:

Crifan Li

版本：v1.2

出版日期 2013-12-15

[15] http://www.crifan.com/files/doc/docbook/soft_dev_basic/release/html/soft_dev_basic.html#cc_by_nc

目录

表格清单

第 1 章 背景介绍

目前在嵌入式开发中，经常要实现将对应的固件，烧写到开发板中，然后开发板才能运行我们的程序。

嵌入式开发，很多用的是Linux系统，也有用WinCE和其他系统，但此文只介绍Linux系统下面的情况。

Linux系统中，多数为bootloader+ kernel + rootfs的模式。

其中，所用的bootloader，多为uboot。负责初始化硬件和设置好软件环境，

然后加载kernel，运行kernel，kernel运行后，再去加载rootfs，之后就是你所看到的运行的Linux了。

其中，在开发过程中，常常会遇到，需要把某个文件，比如U-boot.bin，uImage，rootfs等文件

从PC上，下载到Uboot的SDRAM，即内存中，

然后再用对应命令或工具，将数据写入到某存储介质中。

其中，有时候也需要在发布产品之后，在系统运行的情况下，动态升级整个系统的固件的。

此文就是主要探讨，此嵌入式Linux中，开发过程中和产品发布后，相关的固件烧写方式。

第 2 章 名词解释

2.1. 固件

固件，firmware

所谓固件，就是文件，固化在存储介质上的文件，而文件，其实就是数据。

嵌入式开发中，尤其是Linux开发，常见的方式是，从板子（个人用的是arm的板子）上启动，会允许Uboot，然后Uboot去加载kernel内核，个人常用的kernel是uImage，然后Linux运行后，去加载根文件系统rootfs，个人常用到的yaffs2文件系统。

前后系统运行所需要的文件，总的来说，就是这三个：

- Uboot，比如u-boot.bin
- Kernel，比如uImage
- Rootfs，比如yaffs2.rootfs.arm

而大家一直说的固件，在系统是Linux系统的情况下，常常就是指的是这些文件。

2.2. 烧写

所谓烧写，就是写数据，把文件（固件/数据）写到存储介质（Nand Flash, Nor Flash等）上。

而对于烧写这个词，说法很多，常见的有：

烧写=烧录=flash=编程=program=programming

下面另外提到的，更新固件，其实也指的是将新的固件烧写进去，即所谓更新固件，更新系统。

2.3. 存储介质

存储介质，此处主要是指，嵌入式中存放firmware的地方，多数是Nor Flash加上Nand Flash的组合。其他的，也有单独是Nor Flash，单独是Nand Flash，单独是SD/MMC卡等方式。

2.4. USB Host和USB Device

USB Host，即对于开发板来说，USB是Host端，所以，此时可以去插上一个U盘，对应的是USB Mass Storage的用法，所以，可以理解为：

开发板是USB Host = USB Mass Storage =开发板可以外接 U盘

而USB Device，即对于开发板来说，自己是作为USB的Device端。

而USB Device端，相对于USB Host来说，也叫USB Slave端。

此时的UBS Host就是PC端了，然后PC端连出一根USB线，接上开发板，然后开发板就是USB Device=USB Slave端了，就可以当做U盘用了。

即：

开发板是USB Slave = USB Device = 开发板自己是PC上的U盘。

更多关于USB方面的基本概念和逻辑，可以参考：

USB基础知识概论 [1]

2.5. NIC

NIC，Network Interface Card，网络接口卡，即网卡。

NIC这个叫法，是之前在学习网络方面的知识的时候，遇到的，觉得虽然有点拗口，但是意义表述很明确，所以此处才用此NIC来表示网卡的。

[1] http://www.crifan.com/files/doc/docbook/usb_basic/release/htmls/

第 3 章 固件烧写方式

固件开发方式，这里讨论的主要有两种。

一种是开发过程中，产品发布之前，用到的一些方式。

另外一种是，产品发布之后，产品已经运行了系统了，此时，如何在线动态地更新固件，实现系统升级的功能。

先列出不同的分类：

表 3.1. 嵌入式开发中固件烧录的方式

开发过程中，固件烧写方式	不需要烧写kernel和rootfs的方式	即直接通过tftp,nfs等方式挂在kernel和rootfs的方式
	将kernel，rootfs，uboot等烧写到存储介质上的方式	一步到位的方式（直接通过工具烧写文件到对应存储介质上）
		两步到位的方式（先将数据先写入到Uboot中，再用uboot中的命令把数据写到存储介质上）
产品发布后，动态升级系统的方式	即，通过读写对应的Linux下的设备节点，实现更新固件	

下面就详细讨论这两种过程中所用到的固件升级方式。

3.1. 开发前期或者开发过程中，固件的烧写方式

此处介绍的是，在开发过程中，如何实现固件更新，开发调试，根据是否一定要将新版的固件，烧写到存储介质上，可以分两种：

- 一种是不需要烧写kernel和rootfs的方式；
- 另外一种是，需要把新版本的固件，即uboot，kernel，rootfs，烧写到存储介质上的。

3.1.1. 不需要烧写kernel和rootfs的方式

此种做法，在实际开发中，还是有一些人会用到的。

其背景是，嵌入式开发中，相对普通上层软件开发，每次新编译出一个版本的软件，都要很麻烦地烧录到对应的存储介质，比如Nor Flash上，然后给开发板上电，继续开始调试开发，而不能像开发上层PC端软件，在IDE中，编译一下，点击运行，即可看到最新结果。

所以，嵌入式开发中，开发的效率显得很低，其中一个方法，可以先对避开此问题，避免每次都要重新烧写新编译的程序的问题，那就是，对于新版本的kernel和rootfs，分别通过tftp或NFS挂在kernel，通过NFS挂在rootfs，的方式，重新编译一个新版本的kernel或者是rootfs时，每次都不用重新烧写，只需要把对应的文件，放到对应的tftp或者NFS的文件夹下面即可。

此法详细做法相关的部分内容，下面会涉及，故此处不做太多探讨。而且真的详细讨论的话，超出了此文的范畴。

此处，只是对于此法进行概要说明：

1. 目标
   实现kernel通过tftp挂载，rootfs通过nfs挂载的方式，实现高效率的嵌入式开发
2. 前提
   a. 硬件

i. 开发板上有网卡

ii. 网卡已连接到一个路由或交换机，并且PC端，即提供tftp和nfs的服务器端，也连到此网络，开发板和PC端，同属于一个局域网段。

b. 软件

i. PC端运行了tftp服务，新编译的kernel文件，放在tftp的根目录下

ii. PC端运行了nfs服务，所用的Linux内涵，也设置并启用了对应的nfs服务，编译好的rootfs，放在nfs服务的根目录下。

3. 如何操作

a. uboot中，通过tftp mem_addr kernel_file的方式去加载内核

b. 内核运行起来后，通过NFS去挂在rootfs

c. 正常加载rootfs后，就可以像普通的Linux开发一样，通过串口，输入命令操作Linux了

4. 优缺点

- 优点
  免去了每次新编译的kernel和rootfs，都要重新烧写这一麻烦的事情
- 缺点
  - 很明显，如果开发中，涉及到对应的网络驱动的调试等，内核的NFS服务的调试等，即本身所用到的网络功能都是要调试的对象，那就不能用此法了
  - 另外，网络加载文件的速度，一般都是不错的，但是也不排除，有时候会受其他PC端某个网络资源占用太多的程序的影响
  - 而网络加载文件的稳定性，不同的环境，差异很大。多数情况下，都是很稳定的，但是也有人遇到各种原因，导致不稳定的，所以此时此法即使可用，但用起来也会很郁闷

3.1.2. 将kernel，rootfs，uboot等烧写到存储介质上的方式

需要将对应的文件，烧写到存储介质上，此时，有两种方法：

- 一种是一步到位的方式，即直接通过某工具将文件写入到存储介质上。
- 另外一种是两步到位的方式，先通过某种方式把文件下载到Uboot中，再通过Uboot中的命令，去把数据写入到存储介质上。

3.1.2.1. 一步到位：直接通过工具烧写文件到对应存储介质上

目前常见的存储介质，主要有Nor Flash和Nand Flash，所以下面主要讲解如何烧写Nor Flash还是Nand Flash。

另外，还有一些存储介质是SD/MMC卡等，其烧写数据，我用过的烧写数据方式是，一种是在Uboot中，把下载到内存中的数据，写入到SD/MMC卡中，或者在板子已经跑起来了Linux的环境下，把数据写入到SD/MMC卡中。

3.1.2.1.1. Nor Flash

由于Nor Flash接口比较常见和通用，而且有专门的规范定义了对应的操作命令，所以，目前有很多工具，只要你板子上的Nor Flash是常见的Nor Flash，那么这些工具，多数都可以直接拿过来用，直接将文件烧写到Nor Flash中。

1. 目标
   通过某些工具，连接上开发板或直接接上对应的硬件芯片Nor　Flash，直接通过工具烧写文件到目标存储介质（即对应的硬件芯片）上。即不需要开发板上面运行Uboot或者Linux系统。用工具直接操作即可。

2. 前提
   - 硬件
     - 你所使用的Nor Flash，如果是那种通用的（其实大多数都是通用的），工具所支持的
     - 开发板具有对应的硬件接口，比如JTAG接口
     - 你自己有对应的硬件工具，比如JLink硬件
   - 软件
     - 对应的软件工具支持对应的Nor Flash芯片，比如J-Flash，支持很多种常见Nor Flash的烧写

3. 如何操作
   个人接触比较多的是，Jlink硬件 + 软件工具J-Flash ARM。其如何操作，参见：[4]

3.1.2.1.2. Nand Flash

由于Nand　Flash没有一个统一的规范，和本身操作起来就比较复杂，所以，虽然存在一些工具，可以支持直接烧写Nand Flash，但是相对比较少，使用起来所要求的限制条件也比较多。

此外，是有专门的Nand Flash的烧录器的，一般叫做Nand Flash Programmer，直接将对应的文件，烧写到对应的Nand Flash上的，不过个人没怎么用过，不多解释。

3.1.2.2. 两步到位：先写到内存，再用uboot的命令写到存储介质上

3.1.2.2.1. 如何把文件或数据下载到内存中

下表简单总结了，如何将数据通过硬件接口+相关软件，下载到Uboot中的方式：

表 3.2. 将文件下载到Uboot中的方式

开发板上的硬件接口	软件协议	相关软件或Uboot中的命令	说明
USB	USB Host – USB Mass Storage	Fatls usb 0 fatload usb addr file	USB cable Board has USB Host
	USB Slave/Device	DNW	USB cable Board has USB Slave
RS232	Kermit/Ymodem	loadb/loady	RS232 Cable
NIC	Tftp	tftp file	network interface card Server has tftp service
	NFS	NFS	network interface card Server has NFS service Network cable switch
SD/MMC	SD/MMC	Fatls mmc 0 Fatload mmc 0 addr file	
JTAG	JTAG	IDE tool	Hardware debug tool

开发板上的硬件接口	软件协议	相关软件或Uboot中的命令	说明
			IDE support Load file into Memory

下面，对每一种方式进行详细的阐述：

3.1.2.2.1.1. USB

关于USB Host和USB Device，上面已经名词解释过了，此处不再赘述。

现在很多开发板上，都有USB的Host和USB的Device的接口。

所以，对应着，可以实现，外接U盘到开发板上，或者将开发板作为U盘连到PC上

然后操作U盘，把文件拷贝到U盘里，实现对应的把文件数据传输到开发板上这一功能。

1. USB Host = USB Mass Storage = U Disk

1. 目标
   把插在开发板上的U盘中的文件，拷贝Uboot的内存中
2. 前提
   - 硬件
     - 开发板上有USB Host芯片和接口
       以我这里的TQ2440的板子为例，用的CPU是三星的S3C2440，其中包含了一个OHCI的USB Host主控制器。

       板子上也有USB Host接口。
     - 自己有U盘
   - 软件
     - Uboot中已经实现了USB Host Controller的驱动
       如果Uboot中没有你的板子上的USB Host Controller的驱动的话，需要自己移植，甚至从头实现的话，这个工作量和难度，还是不小的。

       以此处的S3C2440的驱动为例，此处已经把新的版本的Uboot中的相关代码，移植到了TQ2440的1.1.6的uboot中，实现了对应的S3C2440的OHCI的驱动。

       相关过程和源码，参考：[5]
     - U盘的文件系统是FAT格式的
       如果你的U盘是NTFS等其他格式，那么要重新格式化为FAT16/FAT32格式。

       当然，如果是其他的文件系统，比如ext2等，也是可以的，下面对应的命令就是ext2ls和ext2load了。
3. 如何操作
   在Uboot中使用对应命令来操作U盘：
   a. usb rescan

      去初始化usb host。关于usb 子系统更多的相关的命令，可以通过

      ```
      help usb
      ```

      看到更多的帮助信息。

b. fatls usb 0

将你U盘的FAT文件系统中的文件列出来，以确保USB现在可以正常工作，和知道你当前U盘里面有哪些文件，此时应该可以看到你所要拷贝的文件，如果你是把文件放在根目录的话。(一般都是把u-boot.bin等文件，放到U盘根目录的)

c. fatload usb 0 mem_addr file_name

去将U盘中的文件file_name载入到内存中mem_addr的位置。

2. USB Slave = USB Device

1. 目标
将PC端的文件，通过USB线，传输到作为USB Device端的开发板上的Uboot的内存中
2. 前提
   - 硬件
     - 板子上有对应的USB Device功能的controller和对应的USB Device接口
   - 软件
     - PC端已经安装了对应的USB相关驱动
     - PC端需要有对应的DNW软件
     - Uboot中实现了对应的命令
       以TQ2440为例，其中已经有了usb slave 相关功能和命令
3. 如何操作
具体的操作，相对比较麻烦，此处只列出主要步骤：
   a. 去Uboot端执行对应的usb slave命令
   以等待PC主机端传输文件
   b. 去PC端用DNW去传输文件
   USB Port -> Transmit -> 选择要传输的文件

   然后对应的文件就可以传输到对应的Uboot中的内存中去了。

   更多的细节，如何操作，请参看TQ2440的手册：[1]

   在此，免费为天嵌的TQ2440宣传一句，其资料和相关文档，做的是蛮不错的，东西很全，很详细，尤其适合初学者。

3.1.2.2.1.2. RS232

RS232的连接方式，是最常见的。

即，开发板上有串口接口，然后接了根RS232线，连到PC端，然后PC端用一个串口终端程序，连接开发板，比如常见的Windows XP系统自带的超级终端Hyper Terminal，功能强大的SecureCRT，以及Putty等等，都是不错的串口工具。

其中关于如何在Win7下面使用超级终端（Hyper Terminal），不了解的可以去参考：[11]

1. Kermit

Kermit是一种协议，广泛使用的协议，用来传输文件和数据的协议，很早之前就有了此协议，所以现在很多地方都已实现和支持此协议。

关于Kermit和Ymodem的详情，去看我转的帖子：[2]

而关于Kermit，XModem，Ymodem和Zmodem之间的区别和联系，可以去看：[6]

1. 目标
   通过Kermit协议，将文件通过RS232接口传送到Uboot的内存中
2. 前提
   - 硬件
     - 开发板中有RS232接口，并且已连接到PC端
   - 软件
     - Uboot中已经实现kermit协议的loadb命令
       这个，一般的uboot中都已实现。

       此处说一个诡异的事情，之前遇到过，即使help中没有看到loadb的命令，但是实际也是支持loadb的，估计是uboot开发者，把此命令注释掉了，但是实际kermit协议用途太广泛，而uboot本身程序中早已经实现了，所以loadb还是已经在uboot中的了。
3. 如何操作
   a. 在uboot中，输入loadb
   b. 在PC端使用串口程序去传送文件
      以windows XP下的串口工具超级终端为例：

      选择Transfer ⇒ Send File ⇒ Protocol选择Kermit，FileName选择你所要传送的文件->点击确定即可。然后就是慢慢传送文件了。

   至于文件数据传输后，放在uboot的内存中的哪个位置，是由你uboot中的环境变量loadaddr决定，我这里的是loadaddr=0x800000。

   当然，你也可以在执行loady的时候，后面加上你要的地址，比如：

   loadb 0x1000000

   Kermit协议，数据传输速度比较慢，我这里传输了个8MB的文件，大概要40分钟左右的。

2. Ymodem

关于Ymodem协议，是从之前的Xmodem协议演化出来的，之后还有Zmodem。

简单的说就是，一个数据包大小为1KB的数据传输协议。

更多的解释，参见上面已经提到的[6]

1. 目标
   通过Ymodem协议，将文件通过RS232接口传送到Uboot的内存中
2. 前提
   - 硬件
     - 开发板中有RS232接口，并且已连接到PC端
   - 软件
     - Uboot中已经实现Ymodem协议的loady命令

3. 如何操作

   a. 在uboot中，输入loady

   b. 在PC端使用串口程序去传送文件

   以windows XP下的串口工具超级终端为例：

   选择Transfer ⇒ Send File ⇒ Protocol选择Ymodem，FileName选择你所要传送的文件->点击确定即可。然后就是慢慢传送文件了。

4. 示例

```
Bootldr> loady
## Ready for binary (ymodem) download to 0x00800000 at 115200 bps...
CCCxyzModem - CRC mode, 2(SOH)/8192(STX)/0(CAN) packets, 5 retries
## Total Size      = 0x00800000 = 8388608 Bytes
```

3.1.2.2.1.3. NIC/network

多数开发板上，也都带有网卡接口，然后通过网线，连接到一个路由或者交换机上，另外一个PC也连接到此路由或交换机上，然后通过网线，将PC上的文件数据，传输到板子上。

1. tftp

1. 目标
   将文件通过tftp方式，从PC端，下载到Uboot的内存中

2. 前提

   - 硬件
     - 硬件板子上有网卡
     - 板子通过网线连到路由或交换机上，PC也连到该路由或交换机上，共处同一个网段
   - 软件
     - PC端设置好tftp服务
       关于PC端安装了tftp服务（TFTP service），详情可以参考：[7]
     - 安装好tftp服务后，把对应的u-boot.bin等文件，放到tftp的根目录下
     - Uboot中，首先肯定是已经实现了网卡驱动，以及添加了对应的tftp命令
       此两个前提，一般开发板都已经具有此条件

3. 如何操作
   在Uboot中，执行命令
   tftp mem_addr file_name

   就可以将文件file_name传送到Uboot的内存地址mem_addr中了。

4. 示例

```
EmbedSky> tftp 0x30010000 u-boot.bin
dm9000 i/o: 0x20000300, id: 0x90000a46
MAC: 0a:1b:2c:3d:4e:5f
```

```
TFTP from server 192.168.1.101; our IP address is 192.168.1.120
Filename 'u-boot.bin'.
Load address: 0x30010000
Loading: T ###############
done
Bytes transferred = 207396 (32a24 hex)
```

2. NFS

1. 目标
   将文件通过NFS命令，从PC端，通过网络，传送到Uboot的内存中去
2. 前提
   - 硬件
     - 硬件板子上有网卡
     - 板子通过网线连到路由或交换机上，PC也连到该路由或交换机上，共处同一个网段
   - 软件
     - PC端设置好NFS服务
     - Uboot中实现了网卡驱动和nfs命令
     - Uboot中设置好了ip地址，ip掩码mask，网关gateway
3. 如何操作
   Uboot中执行：
   nfs mem_addr IP:path/file
4. 示例

```
nfs 0x30008000 192.168.0.3:/home/nfs/uImage
```

3.1.2.2.1.4. SD/MMC

1. tftp

1. 目标
   将文件从SD/MMC卡中，拷贝到Uboot的内存中
2. 前提
   - 硬件
     - 开发板有SD/MMC的controller，有对应的SD/MMC插槽
     - 自己有SD或MMC卡
   - 软件
     - Uboot中实现了对应的SD/MMC驱动及对应的命令
       关于uboot中，把新版本的mmc驱动，移植到旧的上，可以参考：[8]

- SD/MMC卡是FAT文件系统
  当然，如果是其他的文件系统，比如ext2等，也是可以的，下面对应的命令就是ext2ls和ext2load了。

3. 如何操作

   a. mmcinit或mmc rescan

      即初始化mmc，旧版本的uboot的是mmcinit，新版本的uboot是mmc rescan

   b. fatls mmc 0

      将mmc卡中的文件列出来，确保mmc卡工作正常和知道里面有哪些文件

   c. fatload mmc 0 mem_addr file_name

      将mmc卡中的file_name文件拷贝到内存mem_addr处。

4. 示例

```
EmbedSky> mmcinit
mmc: Probing for SDHC ...
mmc: SD 2.0 or later card found
trying to detect SD Card...
Manufacturer: 0x02, OEM "TM"
Product name: "SA04G", revision 0.5
Serial number: 2621440179
Manufacturing date: 7/2010
CRC: 0x73, b0 = 1
READ_BL_LEN=15, C_SIZE_MULT=0, C_SIZE=365
size = 0
SD Card detected RCA: 0x1234 type: SDHC
EmbedSky> md 30000000
30000000: 00000000 00000000 00000000 00000000 ................
30000010: 00000000 00000000 00000000 00000000 ................
。。。
300000f0: 00000000 00000000 00000000 00000000 ................
EmbedSky> fatls mmc 0
512 nikon001.dsc
misc/
dcim/
194 error.html

2 file(s), 2 dir(s)

EmbedSky> help fatload
fatload <interface> <dev[:part]> <addr> <filename> [bytes]
- load binary file 'filename' from 'dev' on 'interface'
to address 'addr' from dos filesystem

EmbedSky> fatload mmc 0 30000000 error.html
reading error.html

194 bytes read
EmbedSky> md 30000000
30000000: 4d54483c 423c3e4c 3e59444f 6e6f7257 <HTML><BODY>Wron
30000010: 50492067 7263733c 3e747069 646e6977 g IP<script>wind
。。。
```

```
300000f0: 00000000 00000000 00000000 00000000 ................
EmbedSky>
```

上述md（memory display）命令，只是为了显示内存中的内容，用以表示，拷贝文件前后内存中数据的变化。

3.1.2.2.1.5. JTAG == debug tool

1. tftp

1. 目标
   在开发板运行程序的情况下，比如Uboot中，通过debug工具，将文件下载到Uboot的内存中
2. 前提
   - 硬件
     - 开发板上有JTAG等debug接口，连接上对应的Jlink等硬件
   - 软件
     - 对应的debug 工具支持载入文件到内存的功能
       一般debug工具，都是IDE集成开发环境，对应的IDE里面会有对应的功能。

       比如ARM的RVDS，里面就有对应的load file到memory的功能。
3. 如何操作
   在IDE工具中，找到对应的功能选项，然后把文件load载入到开发板的内存中，即可。

3.1.2.2.2. 如何把内存中的数据，写入到对应存储设备上

前面的操作，是把数据从外部传输到Uboot的内存中，接下来，就要把对应的数据，写入到对应的存储介质中去。

常见的存储介质以及Uboot中相关的命令，分类如下：

表 3.3. 如何把Uboot的内存中数据写入到存储设备上

存储介质	Uboot中相关命令	说明
Nand Flash	nand erase nand write	先擦除才能再写入数据
Nor Flash	erase cp.b	先擦除才能再写入数据
USB	usb write	
SD/MMC	mmc write	

下面分别介绍，在Uboot中，对于每种存储设备，如何用相关的命令，把数据写入到对应存储设备中。

3.1.2.2.2.1. Nand Flash

1. 目标
   把Uboot中内存中数据，写入到Nand Flash中去
2. 前提
   - 软件

- Uboot中，已经实现了nand erase和nand write命令了

3. 如何操作
   - nand erase

     需要先用nand的erase命令，去擦出对应的区域
   - nand write

     然后再用nand write，把内存中的数据，写入到nand 中。

3.1.2.2.2.2. Nor Flash

关于Nor Flash，需要额外说明一些事情。

本身Flash这个名词，在存储领域方面，包括了Nand Flash和Nor Flash。

而由于Nor Flash出现最早，应用很广泛，所以Uboot中，对于单独说Flash这个词，是指的是Nor Flash。

所以，会有对应的命令：

flinfo = Flash Info = Nor Flash Info

而又由于Nor Flash的很多操作，很像SDRAM等设备，可以直接读，（写操作需要特定的命令），但是可以把Nor Flash的操作，兼容统一到cp拷贝这个命令中去。

所以，很多时候，你会发现，好像没有单独的Nor Flash的读写的命令，其实是包含在了cp这个命令中了。

另外，对于cp命令本身，其有三种方式：

- cp.b单位为b=byte=字节的方式，去拷贝数据
- cp.w单位为w=word=字的方式，去拷贝数据
- cp.l单位为l=long=长整型的方式，去拷贝数据

不过，对于eeprom，也有单独的一套eeprom的命令的，比如eeprom write，用于将数据写入到eeprom中去。

1. 目标
   把Uboot中内存中数据，写入到Nor Flash中去
2. 前提
   - 软件
     - Uboot中，已经实现了Nor flash 相关的命令了，包括erase和cp命令支持了Nor Flash了
3. 如何操作
   在Uboot中，执行下列命令：
   - protect off

     只有在你当前需要重新写入新数据的Nor Flash的Block是已经被写保护的情况下，才需要此步骤去解除锁定。

     一般情况下，都不需要此步骤的。
   - erase

去擦出Nor Flash中的数据

- cp或eeprom write

  将内存中的数据，写入到Nor Flash中。

3.1.2.2.2.3. USB

1. 目标
   将Uboot中的内存中的数据，写入到USB设备中
2. 前提
   - 软件
     - Uboot中已经实现了对应的usb write命令
3. 如何操作
   在Uboot中，执行下列命令：
   - usb write

     将对应的内存中的数据，写入到Usb设备中。

3.1.2.2.2.4. SD/MMC

1. 目标
   将Uboot中的内存中的数据，写入到SD/MMC设备中。
2. 前提
   - 软件
     - Uboot中已经实现了对应的mmc write命令
3. 如何操作
   在Uboot中，执行下列命令：
   - mmc write

     将对应的内存中的数据，写入到SD/MMC设备中。

3.2. 产品发布后：通过操作设备节点实现动态升级固件

除了开发过程中，去烧写固件之外，在发布产品后，很多厂商，希望在系统运行的情况下，实时地，可以去更新对应的固件，比如kernel的uImage或者rootfs等，此时，多数系统，往往是不太容易这样去升级的，不过还是有可能实现这样的在线升级系统的。

基本的思路是，在运行的Linux中，通过操作对应的设备节点，比如：

1. 对于Nand Flash或者Nor Flash，通过MTD的工具，即mtd-util中的nandwrite等，操作/dev/mtdN将新的固件写入进去
   具体实现方法，参见另外一篇文章：[9]
2. 对于SD/MMC，通过操作/dev/mmc设备，将新的固件写入进去
   以此实现在线升级固件。

   相关实现方式，参考：[10]

参考书目

[1] TQ2440开发板使用手册[1]

[2] 【转】xmodem与kermit协议[2]

[3] uboot 用 nfs 挂载 linux kernel 和 fs[3]

[4] 使用破解版的JLink实现对开发板上的外部Nor Flash的烧写 + JLink V4.08k 下载地址[4]

[5] 【记录】在TQ2440的uboot中添加SD/MMC支持 + 添加USB Mass Storage支持 + 解决fatls乱码问题[5]

[6] 【整理】Kermit Xmodem Xmodem-1K Ymodem Ymodem-G Ymodem-1K Zmodem[6]

[7] 【已成功】Ubuntu 10.10下安装TFTP的步骤 tftp-hpa版本[7]

[8] 【记录】将Uboot 2011.06中mmc驱动移植到uboot 1.1.6的过程[8]

[9] 在Linux运行期间升级Linux系统（Uboot+kernel+Rootfs）[9]

[10] 【经验记录】Linux驱动中如何给SD/MMC卡加多个分区 How to add multi partition for SD/MMC card[10]

[11] 【整理】如何在Win7中安装使用超级终端Hyper Terminal[11]

[1] http://soft.embedsky.net/files/TQ2440开发板使用手册V2.5_20100611.rar
[2] http://www.crifan.com/switch_xmodem_and_kermit_protocol/
[3] http://f.wiseleung.com/?p=154
[4] http://www.crifan.com/use_the_cracked_version_of_jlink_to_achieve_the_development_of_the_external_board_of_nor_flash_programming__jlink_v408k_download/
[5] http://www.crifan.com/recorded_in_the_uboot_tq2440_add_sd__mmc_usb_mass_storage_support__added_support__solution_fatls_garbled/
[6] http://www.crifan.com/order_kermit_xmodem_xmodem-1k_ymodem_ymodem-g_ymodem-1k_zmodem/
[7] http://www.crifan.com/_has_been_successfully_installed_under_ubuntu_1010_tftp_tftp-hpa_version_of_the_steps/
[8] http://www.crifan.com/records_in_the_uboot_201106_mmc_drive_the_process_of_porting_to_uboot_116/
[9] http://www.crifan.com/files/doc/docbook/runtime_upgrade_linux/release/html/runtime_upgrade_linux.html
[10] http://www.crifan.com/records_linux_driver_experience_how_to_sd__mmc_card_with_multiple_partitions_how_to_add_multi_partition_for_sd__mmc_card/
[11] http://www.crifan.com/order_how_to_install_in_win7_using_hyperterminal_hyper_terminal/