|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 文件编号 | |  | 密 级 |  | |
| 文件类型 | |  | 作 者 | 金益彬 | |
| 项目代码 | |  | 适用范围 |  | |
| **飞控计算机1-Vybrid嵌入式软件开发说明** | | | | | |
| 关联文档 | | | | | |
| **文件编号** | **文件名称** | | | | **版本** |
|  | 多平台飞行控制计算机设计说明 | | | | 0.1 |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |
|  |  | | | |  |

**修订记录**

| **版本** | **修订内容说明** | **修订人** | **修订日期** |
| --- | --- | --- | --- |
| 1.0 | 新建 | 金益彬 | 2015/1/18 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 程序烧写、启动及调试

调试只能通过UART0来进行，下一版本需要加入JTAG，方便在线跟踪调试。

CPU程序从Flash加载。程序可以先通过USB0或者UART0烧录到Flash。CPU的启动有三种模式：串口（UART和USB），FUSE以及RCON。RCON模式适合在研发阶段使用，通过配置管脚的设置，可以切换不同的程序加载介质（例如从SD卡启动）；到生产阶段，启动配置参数就固化在CPU内部的FUSE当中，一旦CPU内部的FUSE熔断，RCON模式就无法使用。此时一旦上一次烧录的程序无法启动，就必须进入串口启动模式进行重新烧写。

飞思卡尔提供了USB烧写工具VYBRID\_MFG\_TOOL\_PRG（注：从飞思卡尔官网上下载的工具包里的MFG u-boot源代码有bug，bug详情参考链接<https://community.freescale.com/thread/355140>，其他源代码改动参考压缩包里的README-First）。该工具会先将烧写u-boot（u-boot-mfg）下载到CPU内部ram当中，然后启动该u-boot，设置USB为gadget模式，最后将工作u-boot（u-boot-host）烧入Flash。这种方式的烧写，目前还显得有点复杂，不利于生产。因为需要分多个步骤才能完成最终的程序烧写：

1. 烧写u-boot、kernel
2. Fuse启动参数固化为Flash启动模式（进入u-boot-mfg，运行命令vybridfuse）
3. 启动烧写好的kernel
4. 烧写最终的rootfs

另外还有第三方的烧写工具imx\_loader-master，该工具提供USB以及串口两种烧写方式，只能在linux下运行。

目前的程序烧写过程如下：

1. 直接运行MFGTools，烧写包里的镜像
2. 烧写成功后，断电，进入intramfs启动
3. 烧写ubifs

### Flash空间分配

|  |  |  |
| --- | --- | --- |
| 地址 | 数据 | 说明 |
| 0x000000 | FCB |  |
| 0x040000 | FCB | 备份FCB |
| 0x060000 | Device Tree |  |
| 0x080000 | UBOOT |  |
| 0x0E0000 | Uboot 环境变量 |  |
| 0x100000 | UBOOT | 备份Uboot |
| 0x180000 | Kernel(rootfs) | Rootfs启动 |
| 0x500000 | Kernel(initramfs) | Initramfs启动，用来维护修复系统 |
| 0xa00000 | Rootfs | Ubifs |

### Initramfs启动

在uboot模式下，可以通过修改bootcmd，进入initramfs启动模式。启动模式如下：

=> setenv bootcmd 'nand read 81000000 60000 4000;nand read 80010000 500000 400000;bootm 80010000 - 81000000'

=> boot

### 烧写UBI文件系统

用MFGTool烧写完Uboot和Kernel之后，通过intramfs启动系统，插入优盘，优盘里面事先已经存放有需要烧写的UBI文件系统镜像文件，以及相关（MTD和UBI）操作工具文件。然后执行以下命令：

1. 查看分区容量

#cat sys/class/mtd/mtd5/size

251396096

1. 格式化分区，分区大小根据容量计算：251396098/131072（131072为block大小）

# ./media/sda1/test/mtd/flash\_erase /dev/mtd5 0 1918

1. 卸载之前挂载过的ubi0设备

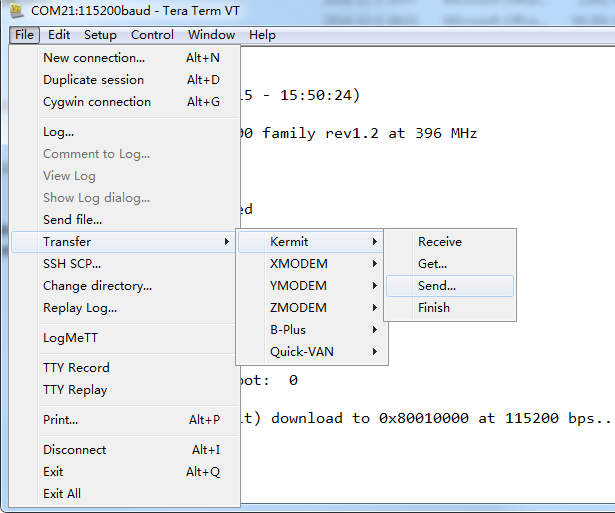
# ubidetach -d 0 /dev/ubi\_ctrl

1. 格式化烧写UBI镜像

# ./media/sda1/test/ubi/ubiformat /dev/mtd5 -f /media/sda1/test/ubi/rootfs.ubi -s 2048 -O 2048

### 烧写DeviceTree

1. 进入uboot
2. 输入命令loadb
3. 从pc控制台软件选择烧写的devicetree镜像



1. 擦除device tree的nand 分区

= > nand erase 60000 20000

1. 烧入device tree

= > nand write 80010000 60000 3e8b（device tree的size）

### GPIO操作



# 软件调试

软件调试需要参考网站[www.timesys.com](http://www.timesys.com)

登陆账号：yibin\_jin 密码：011751abcd

## 搭建调试环境

1. 参考Desktop Factory Host Requirements搭建主机环境<https://linuxlink.timesys.com/docs/wiki/factory/FactoryHostRequirements>
2. 参考Desktop Factory Getting Started Guide搭建编译环境

<https://linuxlink.timesys.com/docs/wiki/factory/FactoryGSG>

1. 参考https://linuxlink.timesys.com/ide安装timestorm调试软件
2. 参考以下链接使用TWR开发板

https://linuxlink.timesys.com/docs/gsg/twr\_vf600

## 交叉编译MTD-utils

从Flash启动kernel需要MTD-utils支持。在kernel从initramfs启动之后，需要MTD-utils来给Flash进行擦除，分区，读写之类的操作，自动静态交叉编译的shell脚本如下：



## Ubifs启动

1. 制作及启动ubifs参考 链接<https://linuxlink.timesys.com/docs/wiki/engineering/HOWTO_Use_UBIFS>

<http://processors.wiki.ti.com/index.php/UBIFS_Support>

1. UBI文件系统的理论及使用注意事项参考linux-mtd网站

<http://www.linux-mtd.infradead.org/doc/ubifs.html>

1. Ubifs出现第二次启动失败，参考linux-mtd网站

<http://www.linux-mtd.infradead.org/doc/ubi.html>

<http://www.linux-mtd.infradead.org/faq/ubifs.html#L_free_space_fixup>

## 更改console输出串口

1. devicetree 设置uart0
2. kernel配置 menuconfig—>kernel hacking --> vybrid debug uart port selection

选择0

## eclipse编译phread出错

Project->Properties->C/C++ Build->Settings->tool settings—>GCC C++ Linker->Libraries  
在Libraries(-l)中添加pthread  
在Libraries search path(-L)中添加crypto

## GitHub

软件以及fpga代码均在<https://github.com/jinyibin>上

### 创建ssh key

1. 打开终端shell，执行，一路回车。最后在主目录下会有.ssh目录，里面有id\_rsa和id\_rsa.pub两个文件

$ ssh-keygen -t rsa -C "jinyibin@gmail.com"

1. 登陆github，在setting—SSH Keys下，add SSH key，填写任意title，粘贴id\_rsa.pub的内容
2. 点击add key

### 常用命令

1. git status ：查看git状态
2. git add file ：添加文件
3. git commit –m “fix bug” :把添加的文件载入git库，并加入说明
4. git push origin master :同步git hub上的master库