软件更新说明

在原来的基础上做了如下的改动：

1，在InsideFlashWR()函数中计算有多少个整数2k块和余数的后面加上一个延时SysCtlDelay(500000);，这里主要是为了应对当用较小的app升级efm32时，不能写入到efm32指定的flash位置。在对efm32内部flash的擦除和写入后，增加了状态监测，本工程中采用while来查询当执行完相关操作后是否成功来决定是否继续。

2，在对bootloader升级的时候，在最终往efm32内部flash写入时执行了4次重新写入的机制。

备注：该工程中的”生成文件-1.bin”就是本bootloader加上i4-app-0428.cfw.

2014-5-12 17:22:24

3，在用旧的bootloader测试第二批I4实验板时，发现有个app升级完后，还一直在升级，后来发现吧loader中堆栈改成1000就可以了。而本工程中的loader的堆栈大小就是1000.本次修改的地方是在写efm32的内部flash时进行了重复3次的操作。

4，本次继续将bootloader的loader的中堆栈指针改到2000，另外生成了新的bootloader的自身的cfw文件，本工程在大板子上进行了efm32.ble,bootloader自身的升级测试。第二批的小板子上也做了测试。

2014年6月20日8:21:26

5，在升级新的BLE的APP时，发现一直在写并没有升级成功，调试发现，程序停在发送开始指令的地方等待BLE的响应。我的更改是在这个地方加上重发机制，加延时。然后，加上重发机制的还有：AppToBoot(),以及发送结束升级的命令部分。做成了新的bootloader：bootloader-self-0621-2.cfw.经过了几次测试，没有发现问题。

2014年6月21日17:35:24

本工程中的“生成文件.bin”是有本bootloader和I4-APP-1.5.cfw生成。最终的bootloader自身的文件是bootloader-self-2.cfw

另外修改的是增加了bootloader的版本号，实现的方法是：我在boot中加入了一个

#pragma location = "ConstSection1"

\_\_root const char abc1[4] = {0,0,BOOT\_VER\_M ,BOOT\_VER\_S};

在globaldata.h中定义了上面2个宏，另外在bootld.icf文件中加入

place at address mem: 0x000097FC { readonly section ConstSection1 };

这个是在release版本中才会生效。

2014年6月25日18:54:38

在boot中更改了欠压闪烁的状态去掉了快速，只用慢闪烁。生成bootloader1.1.cfw。2014年7月7日10:48:04

2014年8月22日12:56:08进行的修改有：

1，在发送开始升级，结束升级命令后等响应时都引入了重发，超时机制，当超时后直接跳转到MCU的APP，避免死机。

2，由于新版的BLE硬件复位后，会首先发送 3c 05 07 01 3e的信息，第二个帧才会发送器件地址和状态信息。而我要等待的信息是第二个帧的，所以在中断中，当收到第二个帧时我才把BLE\_ONLINE置位，然后在分析BLE状态，然后动作。在判断BLE\_ONLINE时，也采取了超时和硬件复位的操作，如果3次硬件操作都没有数据返回给MCU，则直接跳转到MCU的APP。经过测试的BLE版本是1.53，1.54，1.56,1.57，1.63

2014年8月30日16:11:37

本次修订中加入不同设备的FW的判断。

2014年9月12日7:23:11

本次修订的地方：

1），在BLE 的串口中断中，增加了对收到包的内容进行分析，然后在决定，BLE所处的状态，一定要先查看到命令的类型。

2），在往BLE的flash中写数据时，增加了对BLEbootloader的版本的判断，根据不同的版本，在写的时候进行了不同的处理，在新的bootloader中，它的应答信息包含了，该数据是写在那个页的。我这里需要判断，如果写的页不同，那么就直接退出到MCU的APP中。

这次修改后生产1.3版本。