FEE速度的优化方法

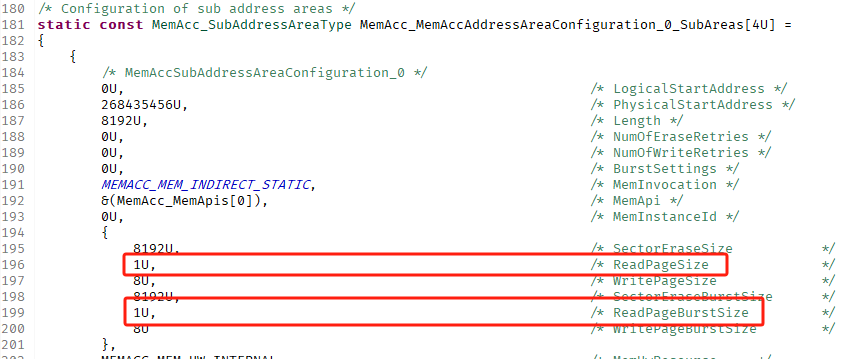
Peter Chen（[Peter\_Chen@wtmec.com](mailto:Peter_Chen@wtmec.com)）

# 现状

S32K3芯片内并不带有EEPROM模拟硬件。为解决客户对于EEPROM数据存储的需求，S32K3使用AUTOSAR的FEE模块，使用PFLASH或者DFLASH来模拟EEPROM。由于S32K3的PFLASH的写入寿命只能保证1000次，而DFLASH的写入寿命可以达到10万次，所以推荐使用DFLASH的Sector来模拟EEPROM。FEE模块是BSW（基础软件）中的一部分，客户会向第三方AUTOSAR软件供应商购买。为了能让客户在不购买BSW的前提下也能有相应的功能，NXP在RTD中提供了一个免费的实现。客户在使用过程中反馈FEE在初始化阶段的耗时太长，导致无法在规定的时间内完成初始化。为了研究FEE的代码，我将FEE样例程序移植到了Windows平台，请参考<https://github.com/chenzch/Fee_On_Windows>。运行了以后发现速度慢的原因主要是默认的配置对于Flash的读取是按照1字节来配置的，初始化的时候如果需要读取60字节的头部数据，就需要连续进入60次MainFunction周期。本文以S32DS for S32 Platform 3.5 + SW32K3\_S32M27x\_RTD\_R21-11\_4.0.0\_HF02\_D2407作为演示，来提升FEE的初始速度。

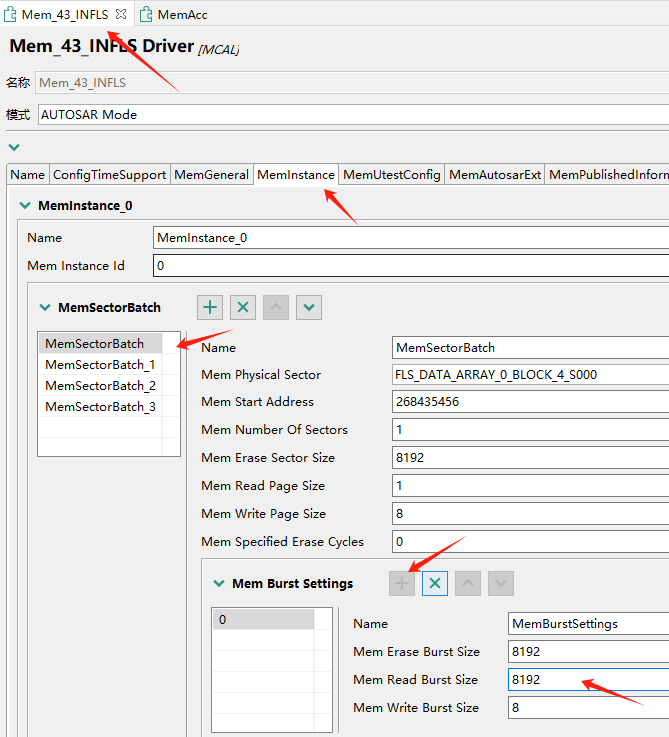
# 修改配置

演示的样例使用RTD自带的Fee\_Example\_S32K344。MemAcc\_PBcfg.c文件中列出了MemAcc模块的配置信息，样例中4个Sector的ReadPageSize和ReadPageBurstSize都被设置成了1，如下图。这样FEE每次最多只能从Flash读取1个字节，初始化的时候就需要进入多次读取周期才能将头部信息读完。

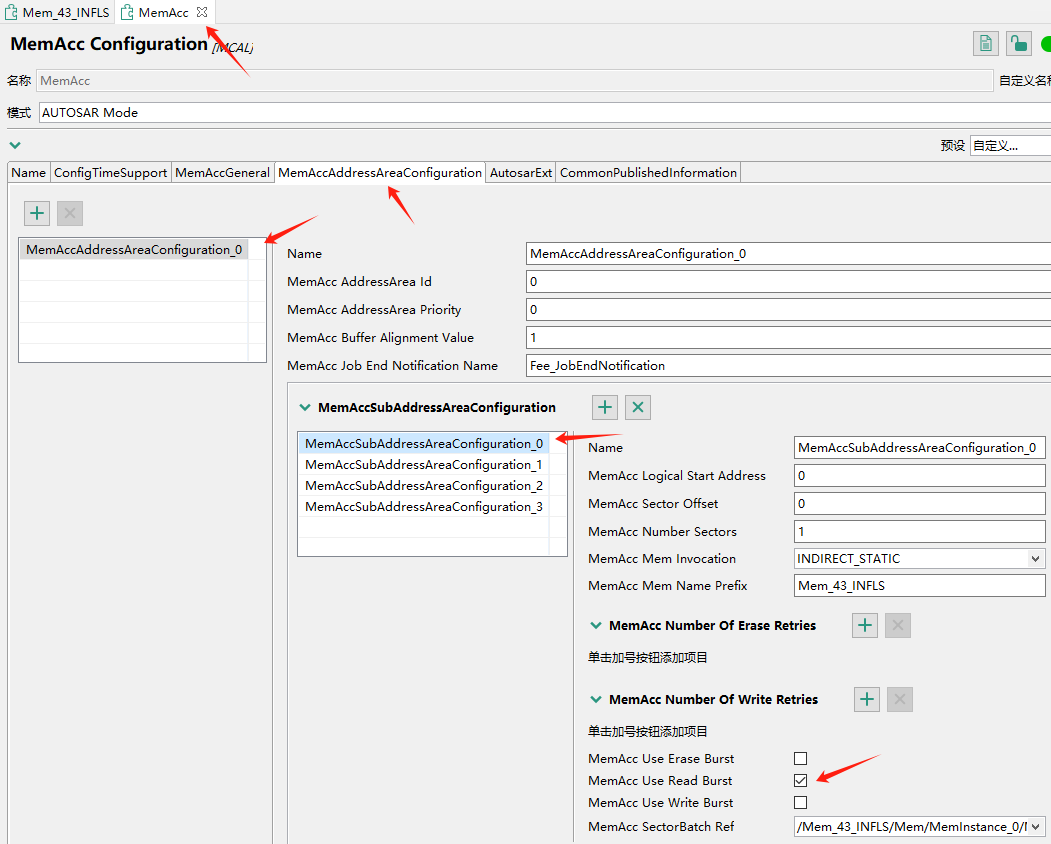


修改方法是将ReadPageBurstSize改成8192，然后在MemAcc中打开Burst读取。由于EEPROM在访问的时候是会有单字节访问的情况出现的，所以ReadPageSize只能设置为1。如果PageSize设置成了其他值，那么对于这个Sector的读取就必须按照这个PageSize进行对齐。其实当你打开Burst读取的时候，这个PageSize就只有验证对齐的功能，对于读取速度没有任何影响，所以可以保留为1。

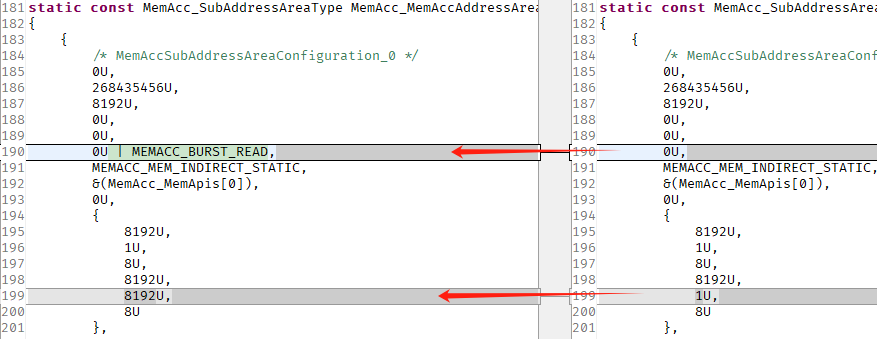
第一步是要将Mem\_43\_INFLS中MemInstance里的每一个MemSector都添加Burst设置，并将Read Burst Size设置成8192。



第二步是将MemAcc组件中MemAccAddressAreaConfiguration中的每一个MemAccSubAddressAreaConfiguration的Use Read Burst打开。



这样生成的代码中就会打开BurstRead功能，每次最大读取量就变成了8192字节。



# 修改驱动

按照以上方法操作以后，初始化速度应该会有4倍左右的提升，如果速度还不能满足需求，就要按照以下方法来修改C40的驱动代码。MemAcc对于Flash的读、写、擦除操作都是通过C40这个驱动来完成的。SW32K3\_S32M27x\_RTD\_R21-11\_4.0.0\_HF02\_D2407这个版本中的C40驱动（C40\_Ip.c : 799行）只实现了1、2、4字节的读取功能。也就是说，我们将Burst读取设置了8192个字节，那么在一个MainFunction周期内最多可以读取8192个字节，但是是拆分成很多个4字节的读取操作来完成的，对于S32K3来说，最经济的读取操作是按照8字节对齐进行读取。这样就需要修改C40的驱动代码，使得最小的读取单位可以扩展成8个字节。当然也可以扩展成无限的字节数，但是一个MainFunction中的多次调用读取函数的开销，在没有设置回调函数的情况下与在一次读取函数中多次实际访问差别并不大。考虑到FEE实际对于FLASH的访问强度在初始化以外的情况下并不会太大，所以没有必要冒风险增加任意字节的访问方法。一共需要修改C40\_Ip\_Types.h和C40\_Ip.c这2个文件，修改的内容请参考以下2个链接。

<https://github.com/chenzch/Fee_On_Windows/blob/main/C40_Ip_Types.h>

<https://github.com/chenzch/Fee_On_Windows/blob/main/C40_Ip.c>