

目录

目录.....	1
1 库体介绍.....	2
2 API 介绍.....	2
2.1 IAP_CodeProgramByteOption	2
2.2 IAP_CodeSectorEraseOption	2
2.3 IAP_EEPROMProgramByteOption.....	3
2.2 IAP_EEPROMSectorEraseOption	3
3 IAP_LIB 导入工程.....	3
4 IAP 操作说明.....	5
5 更改记录.....	7
6 声明.....	7

1 库体介绍

该库适用于赛元 SC95FXX1XB 和 SC95FXX6X 系列芯片。SC95FXX1XB 和 SC95FXX6X 系列芯片在进行 IAP 擦除和写入操作时，由于操作较为复杂，将其封装成库体，并提供 SC95FXXXX_IAP_lib_VXXX.lib，IAP 操作过程中需使用库体提供的 API 接口。

2 API 介绍

2.1 IAP_CODEPROGRAMBYTEOPTION

函数名	IAP_CodeProgramByteOption
函数原型	void IAP_CodeProgramByteOption (unsigned long Add,unsigned char Data);
功能描述	单 Byte 写入
输入参数	Add :需要写入的地址; Data:要写入的 Byte; 操作对象 Code 区
返回值	无
使用说明	1. 需要注意 IAP range 范围 2. LDR0M 不可操作 3. 需要先擦除后写入

2.2 IAP_CODESECTORERASEOPTION

函数名	IAP_CodeSectorEraseOption
函数原型	void IAP_CodeSectorEraseOption(unsigned long Add);
功能描述	扇区擦除，扇区大小 512B
输入参数	Add :需要擦除的地址; 操作对象 Code 区
返回值	无
使用说明	1. 需要注意 IAP range 范围 2. LDR0M 不可操作 3. 固定擦除 512B 4. Add 地址向 512 取整，Add 设置 0X100 进行擦除，擦除的范围为 0X00~0X200 ， Add 设置 0X3000 进行擦除，擦除的范围为 0X3000~0X3200

2.3 IAP_EEPROMPROGRAMBYTEOPTION

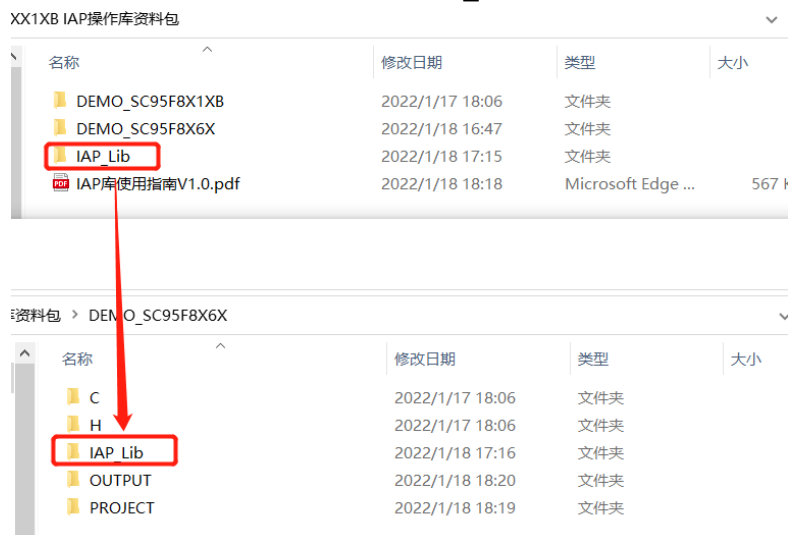
函数名	IAP_EEPROMProgramByteOption
函数原型	void IAP_EEPROMProgramByteOption (unsigned int Add,unsigned char Data);
功能描述	单 Byte 写入
输入参数	Add :需要写入的地址; Data:要写入的 Byte; 操作对象 EEPROM 区
返回值	无
使用说明	1. 需要注意 IAP range 范围 2. LDROM 不可操作 3. 需要先擦除后写入

2.4 IAP_EEPROMSECTORERASEOPTION

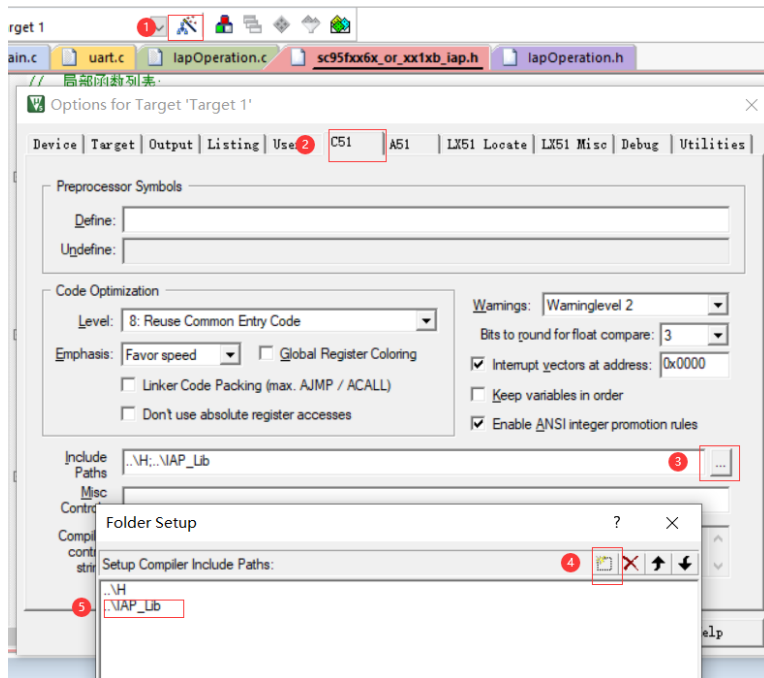
函数名	IAP_EEPROMSectorEraseOption
函数原型	void IAP_EEPROMSectorEraseOption(unsigned int Add);
功能描述	扇区擦除，扇区大小 512B
输入参数	Add :需要擦除的地址; 操作对象 EEPROM 区
返回值	无
使用说明	1. 需要注意 IAP range 范围 2. LDROM 不可操作 3. 固定擦除 512B 4. Add 地址向 512 取整，Add 设置 0X100 进行擦除，擦除的范围为 0X00~0X200 ， Add 设置 0X3000 进行擦除，擦除的范围为 0X3000~0X3200

3 IAP_LIB 导入工程

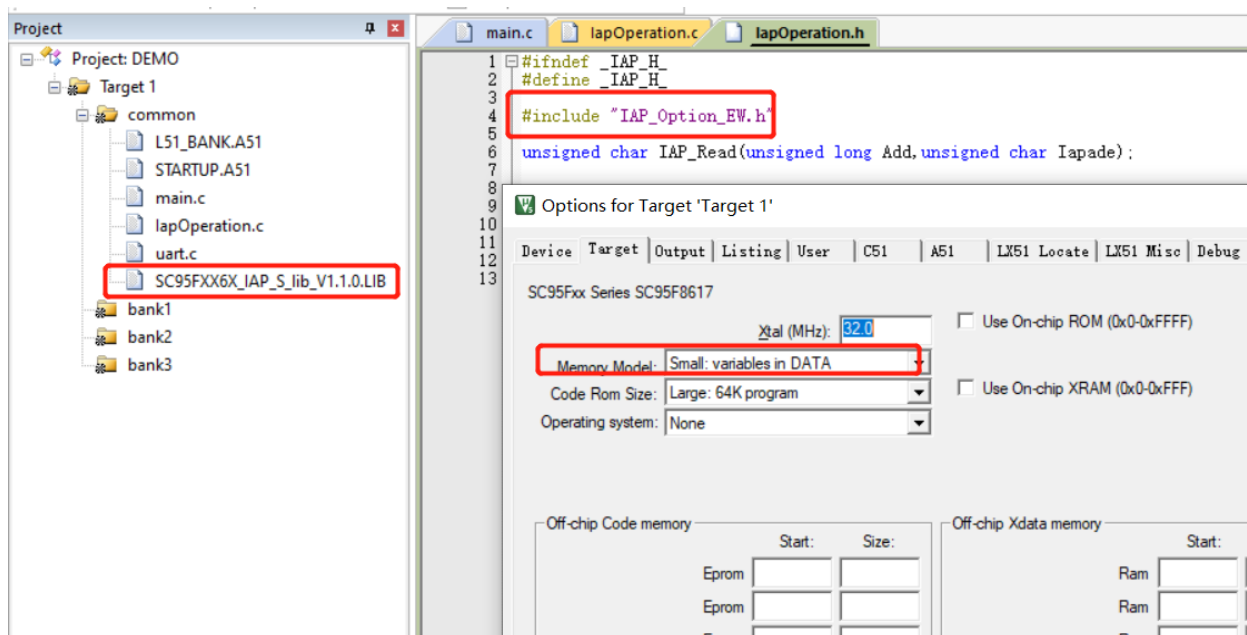
1. 打开 IAP 操作库资料包，将其中的 IAP_LIB 文件拷贝到新建工程中



2. 打开工程，在 Options for Target->C51->Include Paths 中添加 IAP_LIB 文件路径



3. 根据 MCU 型号和 Memory Model (Small 选择带 S 的, Large 选择带 L 的) 选择对应的 lib 文件, 若文件用到库中的 API 则引入头文件 #include "IAP_Option_EW.h" 即可



4 IAP 操作说明

使用注意：

1. 在修改 ROMBNK、IAPADE 进行操作读、写、擦时需要关闭中断，避免 ROMBNK、IAPADE 修改导致 MOVC 指向异常，在进行操作后在恢复中断前需要将 ROMBNK、IAPADE 寄存器值恢复；
2. 在 ROMBNK 进行读值操作时，建议定义一个 xdata 变量；

对 ROM 大小为 64K 及以下 MCU：

不需要进行 ROMBNK 的更改，只需对 IAPADE 进行更改，在更改前需要将中断关闭，避免更改 IAPADE 后产生中断导致 MOVC 指向异常，IAPADE 修改后在操作结束时需要将其恢复，IAP 读函数如下(写操作函数和擦除函数接口 API 具体使用方法见例程)；

```
/******  
*函数名称: unsigned char IAP_Read(unsigned long Add,unsigned char ADER)  
*函数功能: 单Byte读取  
*入口参数: Add : 地址  
*          ADER: 操作对象 APROM为00, EEPROM为02  
*出口参数: void  
*****/  
unsigned char IAP_Read(unsigned long Add,unsigned char Iapade)  
{  
    unsigned char xdata IAP_IapData;  
    unsigned char code *point = 0;  
    //保存IAPADE、EA  
    unsigned char tempADER = IAPADE;  
    unsigned char tempEA = EA;  
  
    EA = 0; //关闭中断  
  
    IAPADE = Iapade;  
    IAP_IapData = *(point+Add);  
  
    //恢复IAPADE、EA，避免MOVC位置出错  
    IAPADE = tempADER;  
    EA = tempEA;  
    return IAP_IapData;  
}
```

对 ROM 64K 以上 MCU:

在对 64K 以上 MCU 进行 IAP 操作时就要将 Flash 物理地址转 MCU 逻辑地址，并且通过设置 ROMBNK 切换 IAP 操作的 Bank 区域具体见如下函数（修改 ROMBNK 的动作和 IAP 读写擦的动作需要在一个函数中，因为在分 Bank 工程中不同 Bank 的函数切换会修改 ROMBNK 中的值，从而导致在读写擦生效时 ROMBNK 的值已经发生变动，出现操作地址不正确的问题），在更改 ROMBNK 或 IAPADE 前需要将中断关闭避免中断，避免更改 ROMBNK 或 IAPADE 后产生中断导致 MOV C 指向异常，ROMBNK 或 IAPADE 修改后在操作结束时需要将其恢复。IAP 写操作函数，擦除函数均已加入该做法，IAP 读函数如下（写操作函数和擦除函数接口 API 具体使用方法见例程）。

```
/******  
*函数名称: unsigned char IAP_Read(unsigned long Add, unsigned char ADER)  
*函数功能: 单Byte读取  
*入口参数: Add : 地址  
*          ADER: 操作对象 APROM为00, EEPROM为02  
*出口参数: void  
*****/  
unsigned char IAP_Read(unsigned long Add, unsigned char Iapade)  
{  
    unsigned char IAP_IapData;  
    //保存ROMBNK、IAPADE、EA  
    unsigned char tempADER = IAPADE;  
    unsigned char xdata tempROMBNK = ROMBNK;  
    unsigned char tempEA = EA;  
    unsigned char code* point = 0;  
    EA = 0; //关闭中断  
    IAPADE = Iapade;  
    if(Add >= 0x10000) //如果超过64K则需要操作Bank寄存器指向目标地址  
    {  
        if(Add < 0x18000)  
        {  
            ROMBNK = (ROMBNK & 0xCF) | 0x20; //切换为Bank0和Bank2  
            Add = (Add - 0x8000); //进行地址转换  
        }  
        else if(Add < 0x20000)  
        {  
            ROMBNK = (ROMBNK & 0xCF) | 0x30; //切换为Bank0和Bank3  
            Add = (Add - 0x10000); //进行地址转换  
        }  
    }  
    else  
    {  
        ROMBNK = (ROMBNK & 0xCF) | 0x10; //地址不超过64K, 切换为Bank0和Bank1  
    }  
  
    IAP_IapData = *(point + Add);  
  
    //操作结束恢复ROMBNK、IAPADE、EA, 避免MOV C位置出错  
    IAPADE = tempADER;  
    ROMBNK = tempROMBNK;  
    EA = tempEA;  
    return IAP_IapData;  
}
```

5 更改记录

版本	记录	日期
V1.0	初版	2022 年 1 月
V1.1	加入了关于 ROMBNK 的注意事项	2022 年 5 月
V1.2	更新关于 ROMBNK 的注意事项，修改接口说明	2022 年 5 月
V1.3	更新 IAP 操作区域，区分 CODE/EEPROM，修改接口说明	2022 年 6 月
V1.4	更新关于 ROMBNK 读值操作的注意事项	2023 年 8 月

6 声明

深圳市赛元微电子股份有限公司（以下简称赛元）保留随时对赛元产品、文档或服务进行变更、更正、增强、修改和改进的权利，恕不另行通知。赛元认为提供的信息是准确可信的。本文档信息于 2022 年 1 月开始使用。在实际进行生产设计时，请参阅各产品最新的数据手册等相关资料。