

一、使用环境及适合的单片机机型

1. 本 STC 库基于 Keil uvision5 如果使用的是其他版本的 keil ,请使用 uvision5 版本
2. 本库集成封装了高级定时器，串口缓冲等高级功能，只支持 1T 单片机，部分老系列的单片机及 12T 单片机暂不获得支持。目前已知支持类型为 STC12 系列，STC15W/F 大部分机型

二、库文件布局

1. 工程文件夹布局



example	2015/6/12 23:26
STClib_Project	2015/6/12 23:26
README.txt	2015/6/12 23:41
更新日志.txt	2015/6/12 23:43
使用手册.docx	2015/6/12 23:56

>.example 存放范例的文件夹，含 ADC、定时器、EEPROM 等范例

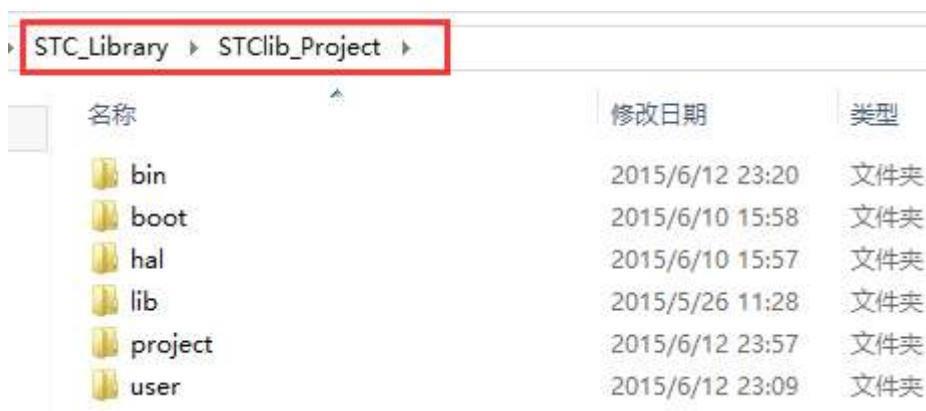
>.STClib_Project 工程模板，使用时将该文件复制到工作路径即可

>.README.TXT 简单说明文件

>.更新日志.TXT 历史更新说明

2. 库文件布局

打开工程模块，文件布局如下：



名称	修改日期	类型
bin	2015/6/12 23:20	文件夹
boot	2015/6/10 15:58	文件夹
hal	2015/6/10 15:57	文件夹
lib	2015/5/26 11:28	文件夹
project	2015/6/12 23:57	文件夹
user	2015/6/12 23:09	文件夹

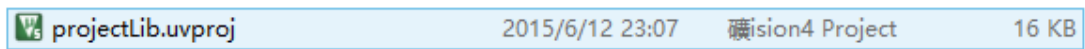
>.bin/ 存放 Hex 文件的文件夹，工程编译完成 Hex 文件在该文件夹生成，格式为 app.hex；另外，若果编译完成时该文件夹存在 app.hex，则先将 app.hex 备份为 app_bak.hex，再重新生成 app.hex

>.boot/ 存放启动文件，该启动文件请勿修改

>.hal/ 存放硬件相关的.C 文件

>.lib/ 存放库文件，C 文件位于 src/文件夹，H 文件位于 inc/

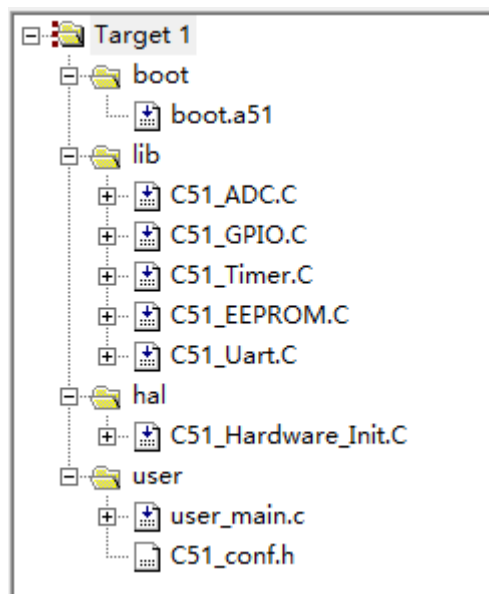
>.project/ 存放工程文件以及编译中间文件，直接双击打开工程



>.user/ 存放用于应用文件

3. 库工程布局

打开工程，布局如下：



>.boot/ 存放启动文件，请勿修改

>.lib/ 存放库文件

>.hal/ 存放硬件相关的文件

>.user/ 存放用户应用文件，已包含 user_main.c

三、如何使用

1.工程配置

工程配置比较简单，只需要修改一个文件“C51_conf.h”即可。（直接在工程

的 user 文件下打开）修改分为三步：

第一步：修改时钟频率，对应目标板的晶振频率修改

第二步：修改使用的单片机机型，保持只有一个宏定义被允许，其他注销

第三步，修改使用的库，不使用的库直接注销

2.使用方法

打开“user_main.c” ,该文件有两个函数“user_init()”以及“user_application()”前者存放初始化相关代码 ,启动文件会调用 ;后者存放应用程序 ,直接在“while (true)” 循环中编写即可(为了保持良好的风格，尽量使用函数接口，而不是编程长长的应用代码)

3.说明

用户新建 C 文件时，只需包含"C51_Lib.h"以及用户使用的其他 H 文件即可

四、库接口及示例

1.系统级接口

4.1.1 delay_ms

原型：void delay_ms(uint16 ms)

实现：C51_Timer.c

功能：毫秒级延时，无需初始化，直接使用即可

输入参数：uint16 ms -- 需要延时的时间

返回：无

4.1.2 user_init

原型: void user_init()

实现: user_main.c

功能: 用户初始化函数; 存放用户初始化功能代码

输入参数: 无

返回: 无

4.1.2 user_application

原型: void user_application()

实现: user_main.c

功能: 用户程序入口; 存放用户代码

输入参数: 无

返回: 无

2. 定时器接口

4.2.1 timer_disarm

原型: void timer_disarm(timer0_t *timer)

实现: C51_Timer.c

功能: 取消定时器的定时功能

输入参数: timer0_t *timer --定时器指针

返回: 无

4.2.2 timer_setfn

原型: void timer_setfn(timer0_t *timer, sysTimer0Func *timerfn, uint8 timer arg)

实现: C51_Timer.c

功能: 设置定时器回调函数, 用于定时器回调

输入参数: timer0_t *timer --定时器指针

*timerfn 回调函数指针

arg 回调函数参数

返回: 无

示例:

```
timer0_t Timer;
```

```
void func(uint8 arg){
```

```
}
```

```
timer_setfn(&Timer, (sysTimer0Func) func, 1);
```

4.2.3 timer_arm

原型: void timer_arm(timer0_t *timer, uint32 period, Timer_Status_Def Timer_Status)

实现: C51_Timer.c

功能: 设置定时器的定时功能

输入参数: timer0_t *timer 定时器指针 period 定时周期

Timer_Status 定时类型, 可为单次定时或者重复定时

返回: 无

3.ADC 接口

4.3.1 GetADC_Result

原型: uint16 GetADC_Result(uint8 Channel, uint8 ADC_Speed)

实现: C51_ADC.C

功能: 获取指定通道的 ADC 转换结果

输入参数: Channel 转换通道, 0~7

ADC_Speed 转换速度

#define ADC_SPEED_540CLOCK 0

#define ADC_SPEED_360CLOCK 1

#define ADC_SPEED_180CLOCK 2

#define ADC_SPEED_90CLOCK 3

返回: 无

五、注意事项

- >.由于程序定时器 0 实现使用了中断，所以程序不允许长期关闭全局中断
- >.定时处理函数请勿长时间占用，保证系统资源
- >.定时器使用请参考例程