目录

[2 准备 1](#_Toc499458288)

[2.1 加入需要的c文件 1](#_Toc499458289)

[2.2 添加头文件目录 2](#_Toc499458290)

[3 SYS模块 4](#_Toc499458291)

[3.1 延时time个毫秒 4](#_Toc499458292)

[3.2 初始化IO 4](#_Toc499458293)

[4 USART模块 4](#_Toc499458294)

[4.1 printf功能 4](#_Toc499458295)

[4.2 scanf()功能 5](#_Toc499458296)

[4.3 自动中断读写BUF（需要打开EA总开关） 5](#_Toc499458297)

[4.4 实例 5](#_Toc499458298)

[5 ADC模块 5](#_Toc499458299)

[5.1 初始化 5](#_Toc499458300)

[5.2 获取ADC转换值 6](#_Toc499458301)

[6 LCD1602模块 6](#_Toc499458302)

[6.1 硬件连接，IO分配设置位于LCD1602.h 6](#_Toc499458303)

[6.2 初始化 6](#_Toc499458304)

[6.3 打印信息至LCD 6](#_Toc499458305)

[7 PWM模块 6](#_Toc499458306)

[7.1 初始化 7](#_Toc499458307)

[7.2 开启通道 7](#_Toc499458308)

[7.3 设置占空比 7](#_Toc499458309)

[7.4 接口重映射 7](#_Toc499458310)

[8 LCD12864模块 7](#_Toc499458311)

[8.1 初始化 8](#_Toc499458312)

[8.2 打印信息到屏幕 8](#_Toc499458313)

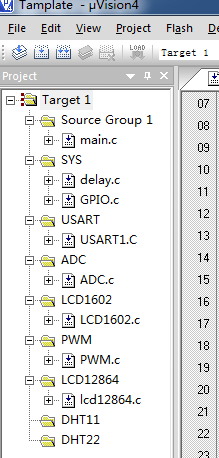
[8.3 使用底层写字符函数 8](#_Toc499458314)

# 准备

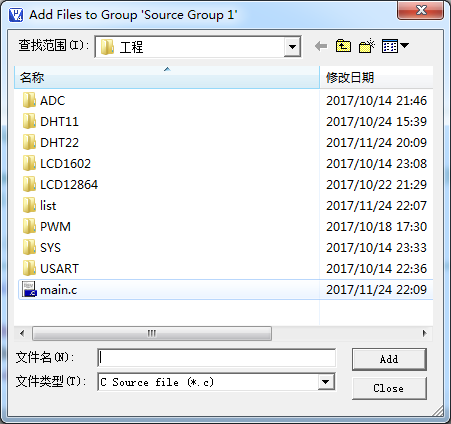
适用于stc15系列，使用时请将c文件加入KEIL，并引入相应头文件目录！

## 加入需要的c文件

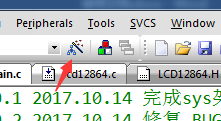
双击 Source Group 1

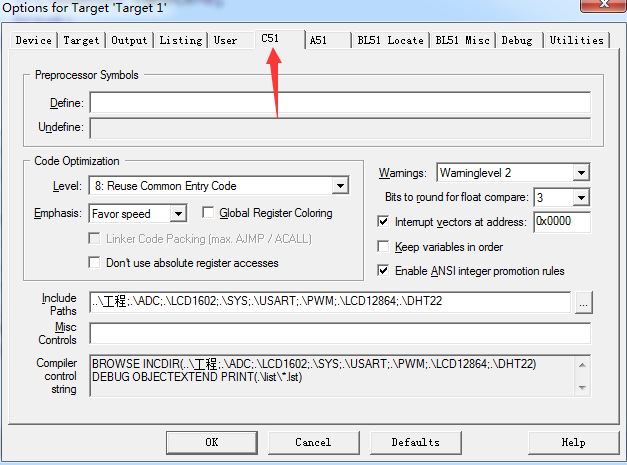


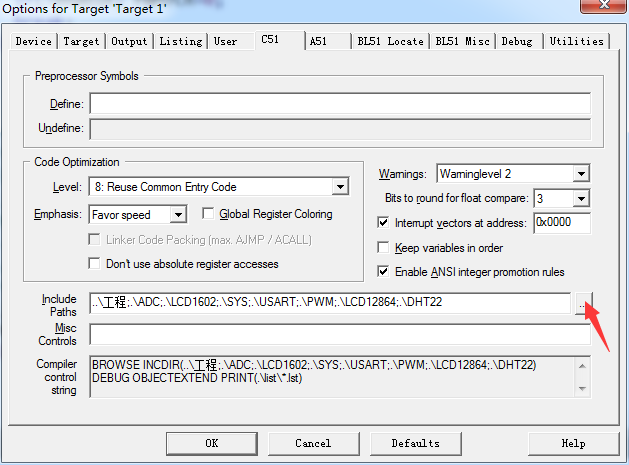
双击所需c文件

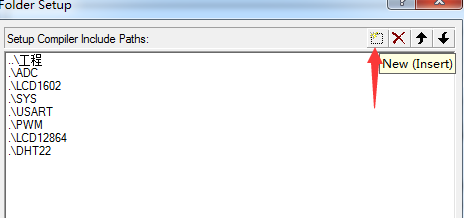


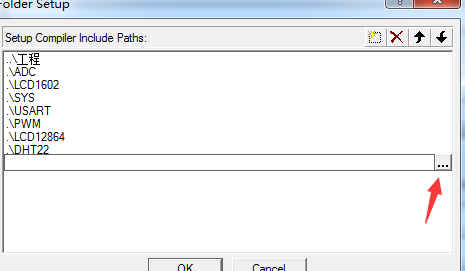
## 添加头文件目录

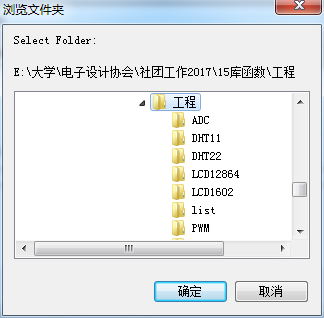












寻找刚才C文件所在目录，添加即可。

下再程序注意：KEIL生成的二进制文件位于list文件夹

首次打开工程因无list文件夹会出现一个警告，忽略即可。第二次即无。

# SYS模块

提供基础延时函数和GPIO初始化。

## 延时time个毫秒

delay\_ms(time);*//time为unsigned int型*

## 初始化IO

GPIO\_INIT();*//所有GPIO配置为准双向*

# USART模块

默认配置串口1,P30\_P31,使用定时器2 默认波特率115200,格式8,1

功能：

## printf功能

已配置好，可使用c语言printf()标准函数将信息打印到串口

例如：

printf("hello world**\n**");

## scanf()功能

使用c标准库函数来向单片机发送命令

注意：stc isp 内串口对此实验显示可能不正确，字符串以\n判定结束

scanf("%4s %d",name,&age);

## 自动中断读写BUF（需要打开EA总开关）

TX1\_Buffer //发送缓冲数组

RX1\_Buffer//接收缓冲数组

## 实例

void usart\_demo(){

int age=0;

unsigned char name[5]={"name"};

printf("name age**\n**");

scanf("%4s %d",name,&age);

printf("name:%sage:%d**\n**",name,age);

}

# ADC模块

内部ADC驱动

## 初始化

调用ADC\_config(ADC\_IO);//ADC基础配置，准备ADC\_IO口

参数ADC\_IO:

#define ADC\_P10 0x01 *//IO引脚 P1.0*

#define ADC\_P11 0x02 *//IO引脚 P1.1*

#define ADC\_P12 0x04 *//IO引脚 P1.2*

#define ADC\_P13 0x08 *//IO引脚 P1.3*

#define ADC\_P14 0x10 *//IO引脚 P1.4*

#define ADC\_P15 0x20 *//IO引脚 P1.5*

#define ADC\_P16 0x40 *//IO引脚 P1.6*

#define ADC\_P17 0x80 *//IO引脚 P1.7*

#define ADC\_P1\_All 0xFF *//IO所有引脚*

## 获取ADC转换值

结果为10位，int类型 1~1024

Get\_ADC10bitResult(u8 channel); *//channel = 0~7*

参数channel：0~7分别对应通道P1.0-P1.7

## 实例

unsigned int result=0;

ADC\_config(ADC\_P10);*//初始化P10为ADC通道*

result=Get\_ADC10bitResult(0);*//获取P10通道ADC值*

printf("ADC:%d**\n**",result);

# LCD1602模块

1602驱动

## 硬件连接，IO分配设置位于LCD1602.h

## 初始化

函数：LCD1602\_init();

LCD1602\_init();

## 打印信息至LCD

本模块提供print\_lcd()函数，使用方法与printf()完全相同,例如：

print\_lcd ("1+1=%d",(int)2);

# PWM模块

6路PWM控制器驱动

## 初始化

PWM\_INIT(DUTY);

初始化PWM周期，DUTY取值1-32767 最大周期即为32767/(主频/16)秒

## 开启通道

PWM\_CHN\_ENABLE(CHN)

初始化PWM通道

CHN取值1-6

对应IO:

通道1：P3.7

通道2：P2.1

通道3：P2.2

通道4：P2.3

通道5：P1.6

通道6：P1.7

## 设置占空比

PWM\_set\_ratio(duty,chn);

duty 取值1-PWM周期，duty应==占空比\* PWM周期

chn 1-6

## 接口重映射

PWM\_port\_remap(unsigned char chn, bit enable)

Chn 通道号

Enabe 1:使能0：否

# LCD12864模块

提供LCD12864核心驱动和显示函数

注意：中文个别字符有乱码BUG，系KEIL编译器导致，可尽量多用英文或安装补丁

## 初始化

调用init\_LCD12864()函数，IO分配设置位于LCD12864.h

## 打印信息到屏幕

使用高级函数print\_lcd12864(char\* fmt ,...)；

使用语法与printf（）相同

例如：print\_lcd12864(“1+1=%d\n”,(int)2);

## 使用底层写字符函数

也可使用底层写字符函数

void disp\_string\_LCD12864(unsigned char x,unsigned char y,unsigned char \*s);

功能：显示一个字符串s到液晶 x：第x行，y：第y列，从第X行第Y列开始写

## 实例

unsigned int adder=0;

init\_LCD12864();*//初始化LCD*

print\_lcd12864("1+1=%d**\n**",(int)2);*//显示数字2*

*//下面体现这钟的优势，写一个秒表，每秒加1*

**while**(1){

adder++;

print\_lcd12864("adder:%d**\n**",adder);

delay\_ms(1000);

}