

移植准备工作

软件准备：

- 1、Keil5(ARM)
- 2、涂鸦涂鸦 MCU SDK
- 3、STM32 Demo 程序

硬件准备：

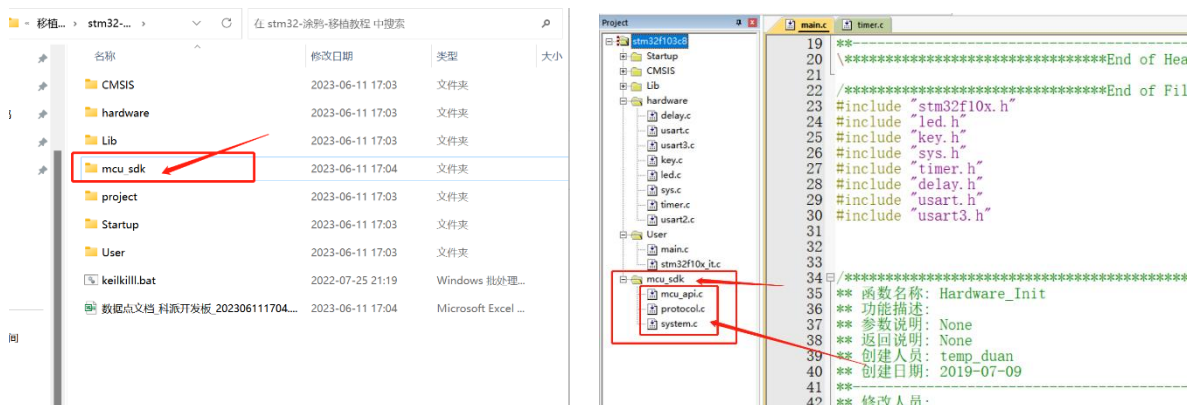
- 1、科派物联网开发板
- 2、数据线
- 3、ST-Link 下载器

涂鸦 MCU SDK 移植

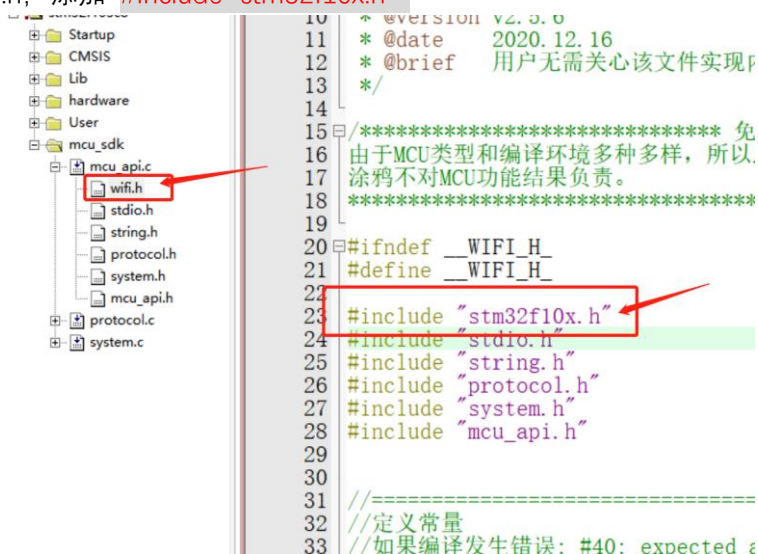
- 1、首先在涂鸦工作台下载所需要的 MCU SDK，如下图所示：



- 2、然后将里面所有的 .c 文件添加进工程



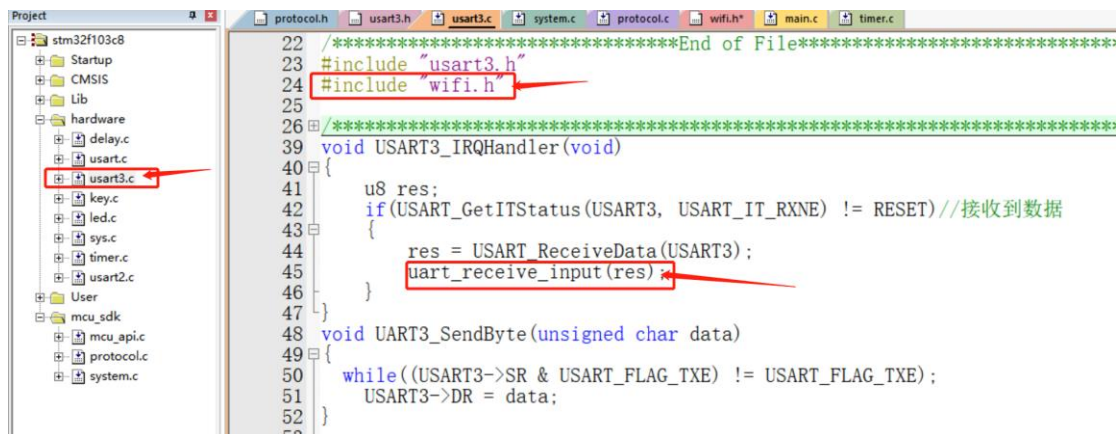
3、进入 wifi.h, 添加 `#include "stm32f10x.h"`



3、在 protocol.c 中, 找到 `uart_transmit_output` 函数 调用串口单字节发送函数。



4、usart3.c 中, 在串口接收中断中调用 `uart_receive_input(value)`, 参数 `value` 是中断接收到的数据。记得先打开串口中断, 程序运行过程中也尽量不要关闭所用到的串口中断。

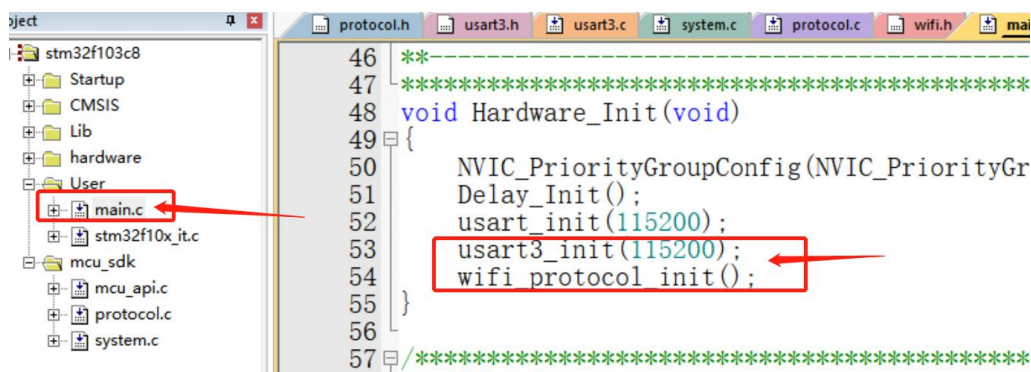


- 5、mcu_api.c 中，进入函数 `my_strcpy(char *dest, const char *src)`，将里面指针 p 的定义放到函数最前面。



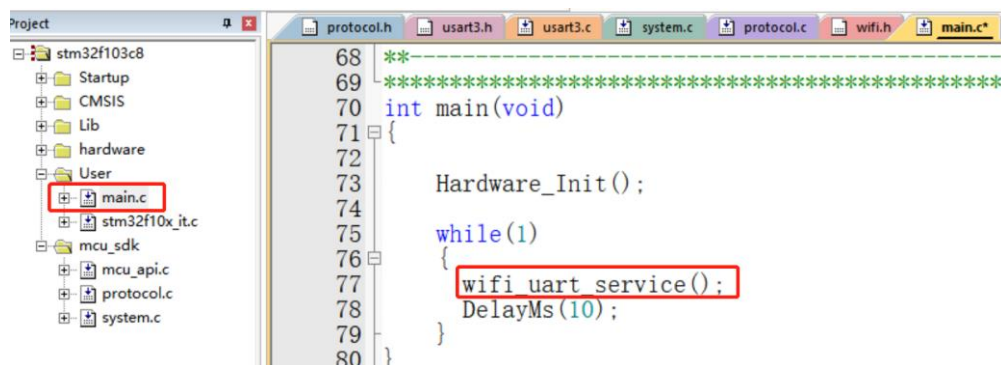
```
130  * @param[in] {src} 源地址
131  * @return 数据处理完后的源地址
132  */
133  char *my_strcpy(char *dest, const char *src)
134  {
135      char *p = NULL;
136      if((NULL == dest) || (NULL == src)) {
137          return NULL;
138      }
139      p = dest;
140      while(*src != '\0') {
141          *dest++ = *src++;
142      }
143      *dest = '\0';
144      return p;
145  }
146  }
147  /**
148  */
```

- 6、在主函数中初始化串口 3，波特率设置为 115200，添加 `wifi_protocol_init()` 完成 wifi 协议初始化。



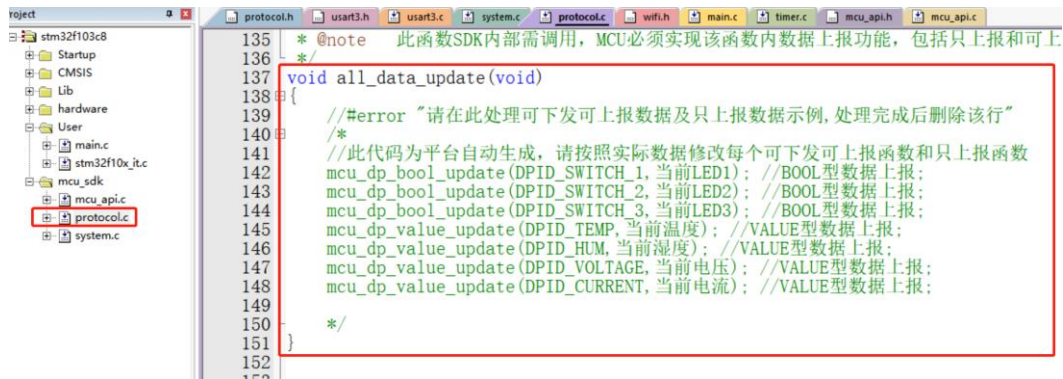
```
46  /**
47  ****
48  void Hardware_Init(void)
49  {
50      NVIC_PriorityGroupConfig(NVIC_PriorityGr
51      Delay_Init();
52      usart_init(115200);
53      usart3_init(115200);
54      wifi protocol init();
55  }
56  }
57  /**
58  ****
```

- 7、主函数 while 循环中调用函数：`wifi_uart_service()`



```
68  /**
69  ****
70  int main(void)
71  {
72      Hardware_Init();
73      while(1)
74      {
75          wifi_uart_service();
76          DelayMs(10);
77      }
78  }
79  }
80  }
```

8、根据自己的需求处理自动化生成数据上报函数 all_data_update()

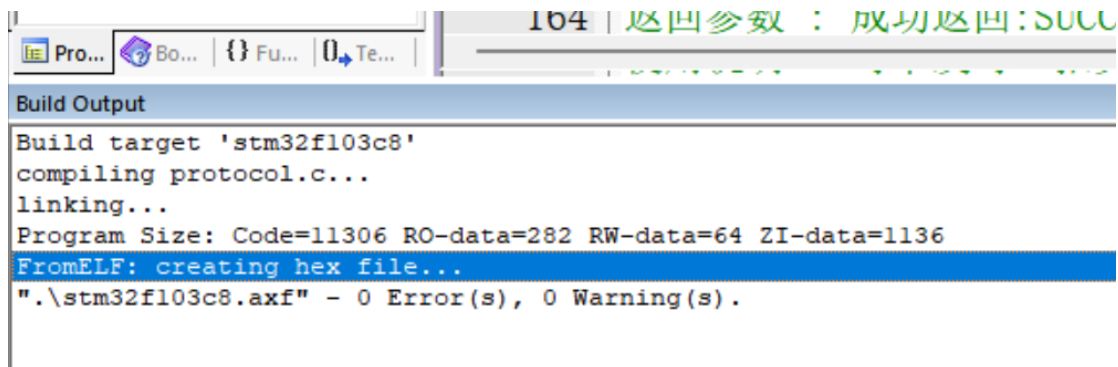


```

135 * @note 此函数SDK内部需调用，MCU必须实现该函数内数据上报功能，包括只上报和可上
136 */
137 void all_data_update(void)
138 {
139     // #error "请在此处理可下发可上报数据及只上报数据示例，处理完成后删除该行"
140     /*
141     // 此代码为平台自动生成，请按照实际数据修改每个可下发可上报函数和只上报函数
142     mcu_dp_bool_update(DPID_SWITCH_1, 当前LED1); // BOOL型数据上报;
143     mcu_dp_bool_update(DPID_SWITCH_2, 当前LED2); // BOOL型数据上报;
144     mcu_dp_bool_update(DPID_SWITCH_3, 当前LED3); // BOOL型数据上报;
145     mcu_dp_value_update(DPID_TEMP, 当前温度); // VALUE型数据上报;
146     mcu_dp_value_update(DPID_HUM, 当前湿度); // VALUE型数据上报;
147     mcu_dp_value_update(DPID_VOLTAGE, 当前电压); // VALUE型数据上报;
148     mcu_dp_value_update(DPID_CURRENT, 当前电流); // VALUE型数据上报;
149     */
150 }
151
152
153

```

9、其他报错直接注释掉引起报错的那行即可



```

164 返回参数：成功返回:SUCCESS
Build Output
Build target 'stm32f103c8'
compiling protocol.c...
linking...
Program Size: Code=11306 RO-data=282 RW-data=64 ZI-data=1136
FromELF: creating hex file...
".\stm32f103c8.axf" - 0 Error(s), 0 Warning(s).

```

10、按键配网

(1) 在 protocol.c 模块工作方式选择打开防误触模式



```

28 /***** 用户相关信息配置 *****/
29
30 /***** 1: 修改产品信息 *****/
31
32 #define PRODUCT_KEY "wthm8ygyenyr6o6m" // 开发平台创建产品后生成的16位字符产品唯一标识
33
34 #define MCU_VER "1.0.0" // 用户的软件版本, 用于MCU固件升级, MCU升级版本需修改
35
36 /* 模块工作方式选择, 只能三选一, 推荐使用防误触模式 */
37
38 // #define CONFIG_MODE CONFIG_MODE_DEFAULT // 默认工作模式
39 // #define CONFIG_MODE CONFIG_MODE_LOWPOWER // 安全模式 (低功耗配网方式)
40 #define CONFIG_MODE CONFIG_MODE_SPECIAL // 防误触模式 (特殊配网方式)
41
42 /* 设置低功耗配网方式和特殊配网方式的配网模式打开时间, 该宏处于注释状态将按三分钟处理, 可
43 // #define CONFIG_MODE_DELAY_TIME 10 // 配网模式打开时间 单位: 分钟
44
45 /* 选择smart模式和AP模式, 该宏都注释将保持smart模式和AP模式互相切换 */
46 // #define CONFIG_MODE_CHOOSE 0 // 模块同时支持AP连接配网和EZ配网无需用户切
47 // #define CONFIG_MODE_CHOOSE 1 // 仅只有AP配网模式
48
49

```


(2)通过一个按键实现配网操作

```

73  ~~~~~
74  int main(void)
75  {
76
77      Hardware_Init();
78
79      while(1)
80      {
81          wifi_uart_service();
82          if(KEY1 == RESET)
83          {
84              DelayMs(20);
85              if(KEY1 == RESET)
86              {
87                  mcu_reset_wifi();
88                  DelayMs(50);
89                  mcu_set_wifi_mode(SMART_CONFIG);
90                  printf("配网模式 SMART_CONFIG\r\n");
91              }
92              while(!KEY1);
93          }
94          DelayMs(10);
95      }
96  }
97
98
99

```

11、手机 APP 绑定设备

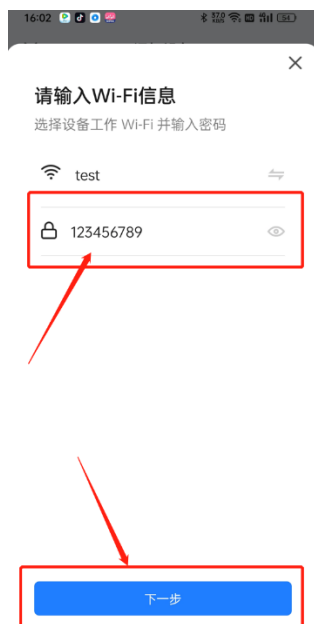
(1) 点击又上角加号添加设备



(2) 设备会发现新设备，点击添加



(3) 输入 WiFi 密码，点击下一步。(WiFi 的频段必须是 2.4GHz)



(4) 等待配网完成。

