请先阅读 STM32F103 机智云

开发手册 V1.0

移植部分再来看本教程

4.3.3 移植

1）添加文件

我们把之前下载的通用平台 SDK 下“Gizwits”文件夹复制到我们的工程下，如图 4.3.3.1 所示。

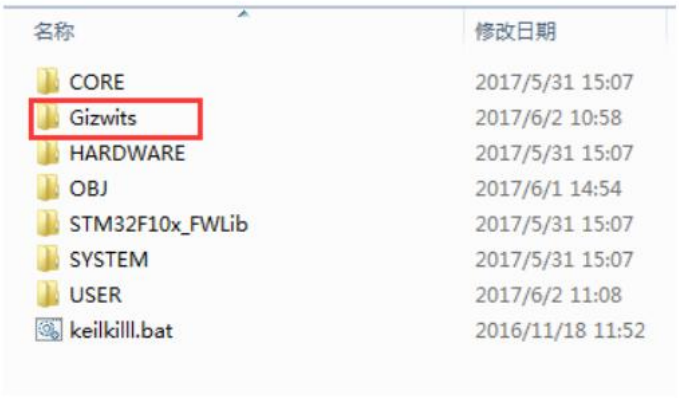
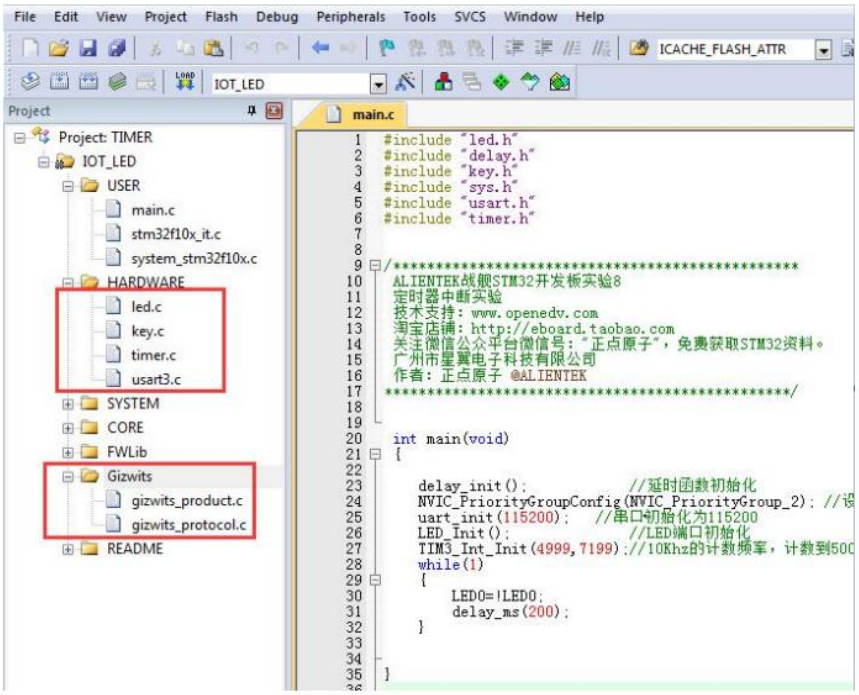


图 4.3.3.1 添加 Gizwits 文件

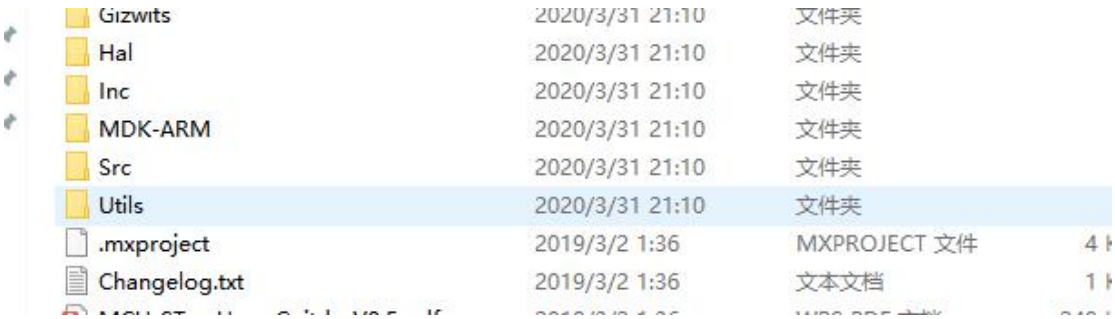
由于 WIFI 的通讯是插在 ATK MODULE 的接口上，而 ATK MODULE 接口是连接到板子的串口 3 上，在 HARDWARE 文件夹中我们创建 USART3 文件夹，并且在文件夹内创建 usart3.c 和 usart3.h 文件 如图 4.3.3.2 所示。

前面步骤完成后，需要将其添加到工程分组中，添加完成以后如图 4.3.3.4 所示。



然后就是区分内容了

1.



将 Utils 文件夹下内容添加至工程　库函数也要包含

2.gitwits_product.c

首先注释开头几行　　如果使用可写类型　include 相应.h 文件

添加 extern u8 wifi_sta;

extern dataPoint_t currentDataPoint;

```
12  *                  链接|增值|开放|中立|安全|自有|自由|生态
13  *                  www.gizwits.com
14  *
15  ****************************************************
16
17  #include <stdio.h>
18  #include <string.h>
19
20  #include "gizwits_product.h"
21  #include "common.h"
22  #include "usart3.h"
23  #include "led.h"
24
25  static uint32_t timerMsCount;
26  uint8_t aRxBuffer;
27  extern u8 wifi_sta;
28  /** User area the current device state structure*/
29  extern dataPoint_t currentDataPoint;
30  //extern keysTypedef_t keys;
31
32  //extern TIM_HandleTypeDef htim2;
33  //extern UART_HandleTypeDef huart1;
34  //extern UART_HandleTypeDef huart2;
35
36  /**@} */
```

可写内容的修改

```
#else
  moduleInfo_t *ptModuleInfo = (moduleInfo_t *)gizdata;
#endif

  if((NULL == info) || (NULL == gizdata))
  {
    return -1;
  }

  for(i=0; i<info->num; i++)
  {
    switch(info->event[i])
    {
      case EVENT_LEDonoff:
        currentDataPoint.valueLEDonoff = dataPointPtr->valueLEDonoff;
        GIZWITS_LOG("Evt: EVENT_LEDonoff %d \n", currentDataPoint.valueLEDonoff);
        if(0x01 == currentDataPoint.valueLEDonoff)
        {
         LED1=0; //user handle
        }
        else
        {
          //user handle
        LED1=1;
        }
        break;



      case WIFI_SOFTAP:
        break;
      case WIFI_AIRLINK:
        break;
      case WIFI_STATION:
        break;
      case WIFI_CON_ROUTER:

        break;
```

注释掉 userHandle()和 PUTCHAR_PROTOTYPE 和 uartInit（）、HAL_TIM_PeriodElapsedCallback、timerInit（）、HAL_UART_RxCpltCallback（）；

```
168 */
169  void userHandle(void)
170 {
171  /*
172
173      */
174
175 }
176
```

```
237   * @retval None
238   */
239  //void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
240  //{
241  //  if(htim==&htim2)
242  //  {
243  //      keyHandle((keysTypedef_t *)&keys);
244  //      gizTimerMs();
245  //  }
246  //}
247
```

```
224   */
225  //PUTCHAR_PROTOTYPE
226  //{
227  //  /* Place your implementation of fputc here */
228  //  /* e.g. write a character to the USART1 and Loop until the end of transmission
229  //  HAL_UART_Transmit(&huart1, (uint8_t *)&ch, 1, 0xFFFF);
230  //
231  //  return ch;
232  //}
233
234 /**
235   * @brief  Period elapsed callback in non blocking mode
236   * @param  htim : TIM handle
237   * @retval None
238   */
239  //void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
240  //{
241  //  if(htim==&htim2)
242  //  {
243  //      keyHandle((keysTypedef_t *)&keys);
244  //      gizTimerMs();
245  //  }
246  //}
247
248 /**
249  * @brief Timer TIM3 init function
250
251  * @param none
252  //* @return none
253 //*/
```

```
255  //{
256  //  HAL_TIM_Base_Start_IT(&htim2);
257  //}
258
259 /**
260   * @brief  This function handles USART IDLE interrupt.
261   */
262  //void HAL_UART_RxCpltCallback(UART_HandleTypeDef*UartHandle)
263  //{
264  //    if(UartHandle->Instance == USART2)
265  //    {
266  //        gizPutData((uint8_t *)&aRxBuffer, 1);
267
268  //        HAL_UART_Receive_IT(&huart2, (uint8_t *)&aRxBuffer, 1);//开启下一次接收中断
269  //    }
270  //}
271
272 /**
```

```
277 */
278 //void uartInit(void)
279 //{
280 //  HAL_UART_Receive_IT(&huart2, (uint8_t *)&aRxBuffer, 1);//开启下一次接收中断
281 //}
282
283 /**
```

修改 mcuRestart（）

```
211 */
212 void mcuRestart(void)
213 {
214     __set_FAULTMASK(1);
215     HAL_NVIC_SystemReset();
216 }
217
218 /**@} */
219
```

```
212 void mcuRestart(void)
213 {
214     __set_FAULTMASK(1);
215     NVIC_SystemReset();
216 }
217
```

修改 uartWrite（）函数

```
292 int32_t uartWrite(uint8_t *buf, uint32_t len)
293 {
294     uint8_t crc[1] = {0x55};
295     uint32_t i = 0;
296
297     if(NULL == buf)
298     {
299         return -1;
300     }
301
302     for(i=0; i<len; i++)
303     {
304 //      HAL_UART_Transmit_IT(&huart2, (uint8_t *)&buf[i], 1);
305 //      while (huart2.gState != HAL_UART_STATE_READY);//Loop until the end of transmission
306
307 //      if(i >=2 && buf[i] == 0xFF)
308 //      {
309 //          HAL_UART_Transmit_IT(&huart2, (uint8_t *)&crc, 1);
310 //          while (huart2.gState != HAL_UART_STATE_READY);//Loop until the end of transmission
311 //      }
312       USART_SendData(USART3,buf[i]);
313         while(USART_GetFlagStatus(USART3,USART_FLAG_TC)==RESET); //循环发送,直到发送完毕
314         if(i >=2 && buf[i] == 0xFF)
315         {
316       USART_SendData(USART3,0x55);
317           while(USART_GetFlagStatus(USART3,USART_FLAG_TC)==RESET); //循环发送,直到发送完毕
318         }
319
320     }
321
```

3.product.h

改开始几行即可

```
15  ************************************
16  #ifndef _GIZWITS_PRODUCT_H
17  #define _GIZWITS_PRODUCT_H
18
19  #ifdef __cplusplus
20  extern "C" {
21  #endif
22
23  #include <stdint.h>
24  #include <stm32f10x.h>
25  #include "gizwits_protocol.h"
26
27  /**
28  * MCU software version
```

4. gizwits_protocol.h

```
275
276  /** User Area Device State Structure */
277  typedef struct {
278    bool valueLEDonoff;
279  } dataPoint_t;
280
```

5.main 函数

定义 userHandle()　初始化各外设和协议

```
/* 用户区当前设备状态结构体*/
dataPoint_t currentDataPoint;

//协议初始化
void Gizwits_Init(void)
{

  TIM3_Int_Init(9,7199);//1MS系统定时
    usart3_init(9600);//WIFI初始化
  memset((uint8_t*)&currentDataPoint, 0, sizeof(dataPoint_t));//设备状态结构体初始化
  gizwitsInit();//缓冲区初始化
}

//数据采集
void userHandle(void)
{

    //判断当前LED1开关量
     if(LED1==0)
      currentDataPoint.valueLEDonoff = 1;
      else
      currentDataPoint.valueLEDonoff = 0;

}

//主函数
 int main(void)
 {
    int key;
   delay_init();         //延时函数初始化
   NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2); //设置NVIC中断分组2:2位抢占优先级
   uart_init(115200);       //串口初始化为115200
   LED_Init();          //LED端口初始化
   KEY_Init();              //按键初始化
     Gizwits_Init();          //协议初始化
   printf("--------机智云-协议移植LED测试实验----------\r\n");
   printf("KEY1:AirLink连接模式\t KEY_UP:复位\r\n\r\n");
     while(1)
   {
      userHandle();//用户采集

        gizwitsHandle((dataPoint_t *)&currentDataPoint);//协议处理
```

6.下载验证

同原文档

作者：yyc

特别鸣谢：爱吃栗子饼