串口收发不定长

七、串口通信优化版（不定长度）

（1）usart_RL.h

```
#define RXBUFFERSIZE  4096          //最大接收字节数
typedef struct UsartData
{
  uint8_t RxBuffer[RXBUFFERSIZE];  //接收数据
  uint8_t aRxBuffer;               //接收中断缓冲
  uint32_t Uart_Rx_Cn;             //接收缓冲计数

}UsartData;

void UART_SendMessage(char * cmd, UART_HandleTypeDef *huart);
void UART_ShowMessage(struct UsartData* Usart, UART_HandleTypeDef *huart);
void UART_MyRxCpltCallback_OnlyReceive_Message(struct UsartData* myUsartData);
void UART_MyRxCpltCallback_Receive_Send_Messasge(struct UsartData* myUsartData,
UART_HandleTypeDef *huart);
```

（2）usart_RL.c

```
/**
 * @brief 将 cmd 指令发送到指定的串口
 *
 * @param cmd 字符串："1234141"
 * @param huart 将接收到的数据发送此串口
 */
void UART_SendMessage(char * cmd, UART_HandleTypeDef *huart)
{
    HAL_UART_Transmit(huart, (uint8_t *)cmd,strlen(cmd),0xFFFF);
    while(HAL_UART_GetState(huart) == HAL_UART_STATE_BUSY_TX);  //检测UART发送结束
}



/**
 * @brief 将数据包发送到指定串口上
 * @param myUsartData 存储数据的额构体
 * @param huart 将接收到的数据发送此串口
 */
void UART_ShowMessage(struct UsartData* myUsartData, UART_HandleTypeDef *huart)
{
    HAL_UART_Transmit(huart, (uint8_t *)myUsartData->aRxBuffer, myUsartData->
```

```
>Uart_Rx_Cn,0xFFFF); //将收到的信息发送出去
    while(HAL_UART_GetState(huart) == HAL_UART_STATE_BUSY_TX);//检测UART发送结束
    myUsartData->Uart_Rx_Cn = 0;
    memset(myUsartData->RxBuffer, 0x00, sizeof(myUsartData->RxBuffer)); //清空数组
```

```
/**
 * @brief 只在中断中接收数据，不进行发送
 *        此函数在接收字符超过数据范围时不会进行任何提示，可以自行修改
 * @param myUsartData 接收数据的结构体
 */

void UART_MyRxCpltCallback_OnlyReceive_Message(struct _UsartData* myUsartData)
{
    myUsartData->RxBuffer[myUsartData->Uart_Rx_Cn++] = myUsartData->aRxBuffer;
    if(myUsartData->Uart_Rx_Cn >= RXBUFFERSIZE) // 假如超出数据范围
    {
        myUsartData->Uart_Rx_Cn = 0;
        // uint8_t errInfo[]="Data Overflow\r\n";
        // while(HAL_UART_Transmit_IT(huart,errInfo,18) == HAL_OK);
    }
    // while(HAL_UART_Receive_IT(huart, (uint8_t *)&(myUsartData->aRxBuffer), 1) ==
HAL_OK);
}

/**
 * @brief 在中断中接收数据，并发送到指定的串口中
 * @param myUsartData 接收数据的结构体
 * @param huart 将接收到的数据发送此串口
 */
void UART_MyRxCpltCallback_Receive_Send_Messasge(struct _UsartData* myUsartData,
UART_HandleTypeDef *huart)
{
    // UART_MyRxCpltCallback_ReceiveMessage(&myUsart1);
    myUsartData->RxBuffer[myUsartData->Uart_Rx_Cn++] = myUsartData->aRxBuffer;
    if(myUsartData->aRxBuffer == 0x0A)
    {
        while(HAL_UART_Transmit_IT(huart, myUsartData->RxBuffer, myUsartData->
Uart_Rx_Cn)==HAL_OK);
        myUsartData->Uart_Rx_Cn = 0;
    }
    if(myUsartData->Uart_Rx_Cn >= RXBUFFERSIZE) // 假如超出数据范围
    {
```

```
            myUsartData->Uart_Rx_Cn = 0;
            uint8_t errInfo[]="too many data!!!\r\n";
            while(HAL_UART_Transmit_IT(huart,errInfo,18) == HAL_OK);
        }
        while(HAL_UART_Receive_IT(huart, (uint8_t *)&(myUsartData->aRxBuffer), 1) ==
HAL_OK);

usart_RL.h](_resources/0b45e98b6bfd4212ac1f0340140c55c0.h)



usart_RL.c](_resources/c4f1b403ee754773ab11ae90b39089d4.c)



        // 注意在中断中重启启动 HAL_UART_Receive_IT，开启中断
```

```
//main.c中
#include "usart_RL.h"

（3）创建全局变量，计数位0，开启串口中断
// 创建结构体“全局”变量
UsartData myUsart1, myUsart3;



// 主函数代码
myUsart1.Uart_Rx_Cn = 0;
myUsart3.Uart_Rx_Cn = 0;


//开启中断
HAL_UART_Receive_IT(&huart3, (uint8_t *)&(myUsart3.aRxBuffer), 1);
HAL_UART_Receive_IT(&huart1, (uint8_t *)&(myUsart1.aRxBuffer), 1);

（4）中断函数调用：
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
  /* Prevent unused argument(s) compilation warning */
  /* UNUSED(huart); */
  /* NOTE : This function should not be modified, when the callback is needed,
            the HAL_UART_RxCpltCallback can be implemented in the user file
   */
    if(huart->Instance == USART3)
    {
```

```
        // 需要手动开启中断
        UART_MyRxCpltCallback_Receive_Send_Messasge(&myUsart3, &huart1);
        HAL_UART_Receive_IT(&huart3, (uint8_t *)&(myUsart3.aRxBuffer), 1);
    }
    else if(huart->Instance == USART1)
    {
        // 需要手动开启中断
        UART_MyRxCpltCallback_Receive_Send_Messasge(&myUsart1, &huart3);
        // UART_MyRxCpltCallback_OnlyReceive_Message(&myUsart1);
        HAL_UART_Receive_IT(&huart1, (uint8_t *)&(myUsart1.aRxBuffer), 1);
    }
}
```

usart_RL.h

usart_RL.c

注意：仅适用于英文字符，不支持中文