ESP8266WIFI模块配置手册-V1.0

1. 运行环境

- stm32f1xx系列 @72Mhz
- AT固件版本 ESP8266_AT_Bin_V1.6.2
- 硬件模块 ESP-WROOM-02D

(若要在别的主频下或者stm32f4xx系列中使用，请根据时钟速度，自行更改TIM4计时器中的参数达到时钟匹配以及底层配置头文件和外设配置)

2. 文件结构说明（code目录下）

- esp8266.c : ESP8266模块功能状态机实现
- esp8266.h
- espusart.c： 配置与ESP8266模块通信的串口
- espusart.h
- key.c： 调试使用的按键(中断)
- key.h
- sys.c： delay.c依赖文件
- sys.h
- delay.c： 提供延时函数
- delay.h

3. 文件使用说明

3.1 串口配置（espusart.c & espusart.h）

3.1.1 配置ESP通信串口

1.串口初始化

根据自己硬件与ESP8266连接的串口对`espusart.c`的`EspUsartInit（）`函数中串口号进行相应的更改

```
void EspUsartInit(u32 baud)
{
  ......
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1|RCC_APB2Periph_GPIOA, ENABLE);
//此处需要修改
    //USART1_TX   GPIOA.9
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_9; //此处需要修改
    ......
    GPIO_Init(GPIOA, &GPIO_InitStructure);//此处需要修改

    //USART1_RX   GPIOA.10初始化
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10;//此处需要修改
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
    GPIO_Init(GPIOA, &GPIO_InitStructure);//此处需要修改

    //Usart1 NVIC 配置
    NVIC_InitStructure.NVIC_IRQChannel = USART1_IRQn;//此处需要修改
```

```
    ......

    //USART 初始化设置
    ......
    USART_Init(USART1, &USART_InitStructure); //此处需要修改
    USART_ITConfig(USART1, USART_IT_RXNE, ENABLE);//此处需要修改
    USART_Cmd(USART1, ENABLE);                    //此处需要修改
    USART_ClearITPendingBit(USART1, USART_IT_RXNE);//此处需要修改
    TIM4_Init(1000-1, 720-1);
    #ifdef DEBUG
        DebugUsartInit(115200);
    #endif
}
```

2.串口中断函数

根据自己硬件与ESP8266连接的串口对`espusart.c`的`USART1_IRQHandler()`函数中串口号进行相应的更改

```
//ESP串口中断服务程序
void USART1_IRQHandler(void)     //此处需要修改串口号
{
    u8 Res;
    if(USART_GetITStatus(USART1, USART_IT_RXNE) != RESET)  //此处需要修改串口号
    {
        USART_ClearITPendingBit(USART1, USART_IT_RXNE);  //此处需要修改串口号
        Res =USART_ReceiveData(USART1);   //此处需要修改串口号
        ......
            case NORMAL:
            ......
                if(USART_GetFlagStatus(USART1, USART_FLAG_ORE) == SET){//此处需要
修改串口号
                    USART_ClearFlag(USART1, USART_FLAG_ORE);//此处需要修改串口号
                    USART_ReceiveData(USART1);//此处需要修改串口号
                }
                break;
            ......
        }
    }
}
```

3. 串口支持printf

根据自己硬件与ESP8266连接的串口对`espusart.c`的`EspPrintf()`函数中串口号进行相应的更改

```
void EspPrintf(char* fmt, ...){
    ......
    for(j = 0; j < i; j++){
        while(USART_GetFlagStatus(USART1, USART_FLAG_TC) == RESET);//此处需要修改串
口号
        USART_SendData(USART1, usart1TBuffer[j]);//此处需要修改串口号
    }
}
```

4. 串口特殊说明

在检测ESP模块连接状态过程中，串口数据接收使用到了TIM4计时器定时10ms做接收超时判断，如若使用不同主频的单片机，需要自行计算计数值并修改`EspUsartInit()`中定时器初始化调用函数`TIM4_Init(1000-1, 720-1)`的参数。

3.1.2 配置调试串口

1. 关闭串口调试

将`espusart.h`中的`#degine DEBUG`注释掉

```
/***************espusart.h******************/
//#define DEBUG
```

2. 启用串口调试

根据自己硬件调试串口对`espusart.c`的`DebugUsartInit()`函数中串口号进行相应的更改

```
void DebugUsartInit(u32 baud)
{
     ......
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOB, ENABLE);//此处需要修改
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART3, ENABLE);//此处需要修改

  //USART3_TX   GPIOB.10
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10;//此处需要修改
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
    GPIO_Init(GPIOB, &GPIO_InitStructure);//此处需要修改

  //USART3_RX     GPIOB.11
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_11;//此处需要修改
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
    GPIO_Init(GPIOB, &GPIO_InitStructure);//此处需要修改

  //USART 初始化设置
    ......
    USART_Init(USART3, &USART_InitStructure); //此处需要修改


  //Usart3 NVIC 中断初始化
    NVIC_InitStructure.NVIC_IRQChannel = USART3_IRQn;//此处需要修改
    ......
    USART_ITConfig(USART3, USART_IT_RXNE, ENABLE);//此处需要修改
    USART_Cmd(USART3, ENABLE);                    //此处需要修改

}
```

根据自己硬件与ESP8266连接的串口对`espusart.c`的`USART3_IRQHandler（）`函数中串口号进行相应的更改

```
//调试串口中断服务程序
void USART3_IRQHandler(void)   //此处需要修改
{
    u8 Res;
    if(USART_GetITStatus(USART3, USART_IT_RXNE) != RESET)  //此处需要修改
    {
        Res =USART_ReceiveData(USART3); //此处需要修改
        USART_SendData(USART1,Res);
    }

}
```

3.2 按键控制配置（key.c & key.h）

3.2.1 使用实体按键配网

根据按键硬件连接的GPIO口对`key.c`的`KeyExtiInit()`中GPIO口进行相应的修改

```
void KeyExtiInit(void)
{
    ......

    //打开GPIOE时钟和AFIO时钟
    RCC_APB2PeriphClockCmd( RCC_APB2Periph_GPIOA | RCC_APB2Periph_AFIO, ENABLE
);//此处需要修改
    ......
    NVIC_InitStructure.NVIC_IRQChannel = EXTI9_5_IRQn;//此处需要修改
    ......
    // PE4外部中断配置
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_6;//此处需要修改
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
    GPIO_Init( GPIOA, &GPIO_InitStructure );//此处需要修改

    GPIO_EXTILineConfig( GPIO_PortSourceGPIOA, GPIO_PinSource6 );//此处需要修改
    EXTI_InitStructure.EXTI_Line = EXTI_Line6;//此处需要修改
    EXTI_InitStructure.EXTI_Mode = EXTI_Mode_Interrupt;
    EXTI_InitStructure.EXTI_Trigger = EXTI_Trigger_Falling;
    EXTI_InitStructure.EXTI_LineCmd = ENABLE;
    EXTI_Init( &EXTI_InitStructure );

}
```

根据按键硬件连接的GPIO口对`key.c`的`EXTI9_5_IRQHandler()`中中断线进行相应的修改

```
void EXTI9_5_IRQHandler( void )//此处需要修改
{
    if( EXTI_GetITStatus( EXTI_Line6 ) != RESET )//此处需要修改
    {
        ......
        EXTI_ClearITPendingBit( EXTI_Line6 );//此处需要修改
    }
}
```

3.2.2 其他方式触发配网

将以下代码放到触发配网的函数中

```
switch(espState){       //切换WIFI模块工作模式
            case NORMAL:
                ChangeState(START_CFG);          //进入配网模式
                break;
            default:
                break;
        }
```

将以下代码放到强制退出配网的函数中

```
switch(espState){       //切换WIFI模块工作模式
            case WAIT_CFG:
            case CHECK_CONT:
            case CHECK_TCP:
                ChangeState(FORCE_STOP);         //手动退出配网
                break;
            default:
                break;
        }
```

3.3 用户数据处理（esp8266.c & esp8266.h & espusart.c)

3.3.1 串口接收配置

修改 `espusart.c` 文件 `USART1_IRQHandler()` 函数 `case NORMAL` 中的代码段即可

```
case NORMAL:
                #ifdef DEBUG
                    USART_SendData(USART3,Res);
                #endif
                usart1Buffer[usart1Rx] = Res;
                usart1Rx++;
                usart1Rx &= 0xFF;
                if(usart1Buffer[usart1Rx-1] == 0x5A){
                    usart1S = usart1Rx - 1;
                }
                if((usart1Buffer[usart1S] == 0x5A) && (usart1Buffer[usart1Rx -
1] == 0xA5)){
                    usart1Len = usart1Rx-1 -usart1S;
                    usart1Sta = 1;
                }
                if(USART_GetFlagStatus(USART1, USART_FLAG_ORE) == SET){
                    USART_ClearFlag(USART1, USART_FLAG_ORE);
                    USART_ReceiveData(USART1);
                }
                break;
```

3.3.2 数据解析配置

将自己的数据解析代码放到`esp8266.c`文件的`ProcessData()`函数中即可，其中两个函数为测试函数，实际使用中请将`SendHeart()`函数中内容重写，将`EspTest()`函数注释掉

```
/*************************************************
*函数名：    ProcessData
*函数功能：   传输数据处理
*输入参数：   无
*返回值：    无
*************************************************/
void ProcessData(void){
    SendHeart();    //发送心跳包测试
    EspTest();      //注册样机测试
}
```

4. 加入模块

将修改后的`code`文件夹文件添加到工程中

- 主文件中添加`#include"esp8266.h"`
- 主程序初始化中加入`EspInit()`函数
- 主循环中添加`EspTask()`函数
- 修改`esp8266.c`中的变量`espIP`、`espSSID`、`espPWD`变量或保持不变

```
char* espIP = "192.168.1.1";    //esp配置模式热点IP
char* espSSID = "ESP8266";      //esp配置模式热点SSID
char* espPWD = "12345678";      //esp配置模式热点PWD
char* espPORT = "1234";         //esp配置模式热点PORT
```

在主程序任务刷新定时器（0.1ms）中添加变量`heartCounter`自增（心跳包测试函数依赖）。