淘宝店铺

优秀不够，你是否无可替代

知识从未如此性感。 烂程序员关心的是代码,好程序员关心的是数据结构和它们之间的关系 --QQ群: 607064330 --本人QQ:946029359 --淘宝 https://shop411638453.taobao.com/

随笔 - 693, 文章 - 0, 评论 - 311, 阅读 - 173万

导航

博客园
首页
新随笔
联系
订阅 XML
管理

公告

渡我不渡她 -
Not available
00:00 / 00:00

1 渡我不渡她
2 小镇姑娘
3 PDD洪荒之力

加入QQ群

昵称： 杨奉武
园龄： 5年8个月
粉丝： 607
关注： 1

搜索

找找看
谷歌搜索

我的标签

8266(88)
MQTT(50)
GPRS(33)
SDK(29)
Air202(28)
云服务器(21)
ESP8266(21)
Lua(18)
小程序(17)
STM32(16)
更多

随笔分类

Android(22)
Android 开发(8)
C# 开发(4)
CH395Q学习开发(5)
ESP32学习开发(8)
ESP8266 AT指令开发(基于STC89C52单片机)(3)
ESP8266 AT指令开发(基于STM32)(1)
ESP8266 AT指令开发基础入门篇备份(12)
ESP8266 LUA脚本语言开发(13)

5-网络芯片CH395Q学习开发-模块使用Socket0作为TCP客户端和电脑上位机TCP服务器局域网通信

网络芯片CH395Q学习开发

开发板链接:开发板链接

模组原理图:模组原理图

资料源码下载链接:https://github.com/yangfengwu45/CH395Q.g

- 学习Android
  教程中搭配的Android，C#等教程如上，各个教程正在整理。

- 1-硬件测试使用说明
- 2-学习资料说明,测试通信,获取硬件版本,程序移植说明
- 3-芯片初始化,网线连接检测实验
- 4-关于中断检测和DHCP实验
- 5-模块使用Socket0作为TCP客户端和电脑上位机TCP服务器局域网通信
-
-
-
-
-
-

最新评论

1. Re:C#委托+回调详解
好文，撒也不说了，直接收藏！
--杨咩咩plus
2. Re:2-STM32 替换说明-CKS32, HK32, MM32, APM32, CH32, GD32, BLM32, AT32(推荐), N32, HC华大系列
有用，谢谢！
--你跟游戏过吧

阅读排行榜

说明

这节演示一下模组作为TCP客户端和电脑上位机TCP服务器局域网通信

提醒:无论是SPI,USART,并口,程序操作步骤都是一样的!

只是不同的接口发指令发给模块,然后用不同的接收接收数据而已.

测试本节代码

1.用户可以使用杜邦线根据自己的情况设置和连接引脚

4. 用ESP8266+android,制作自己的WIFI小车(ESP8266篇)(62517)
5. 有人WIFI模块使用详解(38092)
6. (一)基于阿里云的MQTT远程控制(Android 连接MQTT服务器,ESP8266连接MQTT服务器实现远程通信控制----简单的连接通信)(35372)
7. 关于TCP和MQTT之间的转换(32223)
8. android 之TCP客户端编程(31277)
9. android客服端+eps8266+单片机+路由器之远程控制系统(31126)
10. C#中public与private与static(30925)

推荐排行榜

1. C#委托+回调详解(9)
2. 用ESP8266+android,制作自己的WIFI小车(ESP8266篇)(8)
3. 用ESP8266+android,制作自己的WIFI小车(Android 软件)(6)
4. ESP8266使用详解(AT,LUA,SDK)(6)
5. 关于TCP和MQTT之间的转换(5)

ch395cmd.h | CH395INC.H | CH395SPI.C | usart.c | delay.c | timer.c | main.c | delay.h | CH395SPI.H

```
#ifndef __CH395SPI_H_
#define __CH395SPI_H_

#include "CH395INC.H"

/*****************************配置GPIO (根据自己的修改)*********************************
//时钟
#define CH395_CONFIG_SPI_CLK()    ( RCC_APB1PeriphClockCmd( RCC_APB1Periph_SPI2,ENABLE) )
#define CH395_CONFIG_GPIO_CLK()   ( RCC_APB2PeriphClockCmd( RCC_APB2Periph_GPIOA | RCC_APB2Peri
//设置使用的SPI
#define USE_SPI_SPI2
//SPI_CS  --  连接模块SCS引脚
#define CH395_CS_PORT    GPIOB
#define CH395_CS_PIN     GPIO_Pin_12
//SPI_CLK --  连接模块SCK引脚
#define CH395_CLK_PORT   GPIOB
#define CH395_CLK_PIN    GPIO_Pin_13
//SPI_MISO --  连接模块SDO引脚
#define CH395_MISO_PORT  GPIOB
#define CH395_MISO_PIN   GPIO_Pin_14
//SPI_MOSI --  连接模块SDI引脚
#define CH395_MOSI_PORT  GPIOB
#define CH395_MOSI_PIN   GPIO_Pin_15
//RST --  连接模块RST引脚
#define CH395_RST_PORT   GPIOA
#define CH395_RST_PIN    GPIO_Pin_8
//TX --  连接模块TX引脚
#define CH395_TX_PORT  GPIOA
#define CH395_TX_PIN   GPIO_Pin_3
//INT  -- 连接模块INT引脚 (检测到该引脚低电平信号之后再获取数据)
#define CH395_INT_PORT  GPIOA
#define CH395_INT_PIN   GPIO_Pin_0
/*************************************************************************/
```

2,注意!

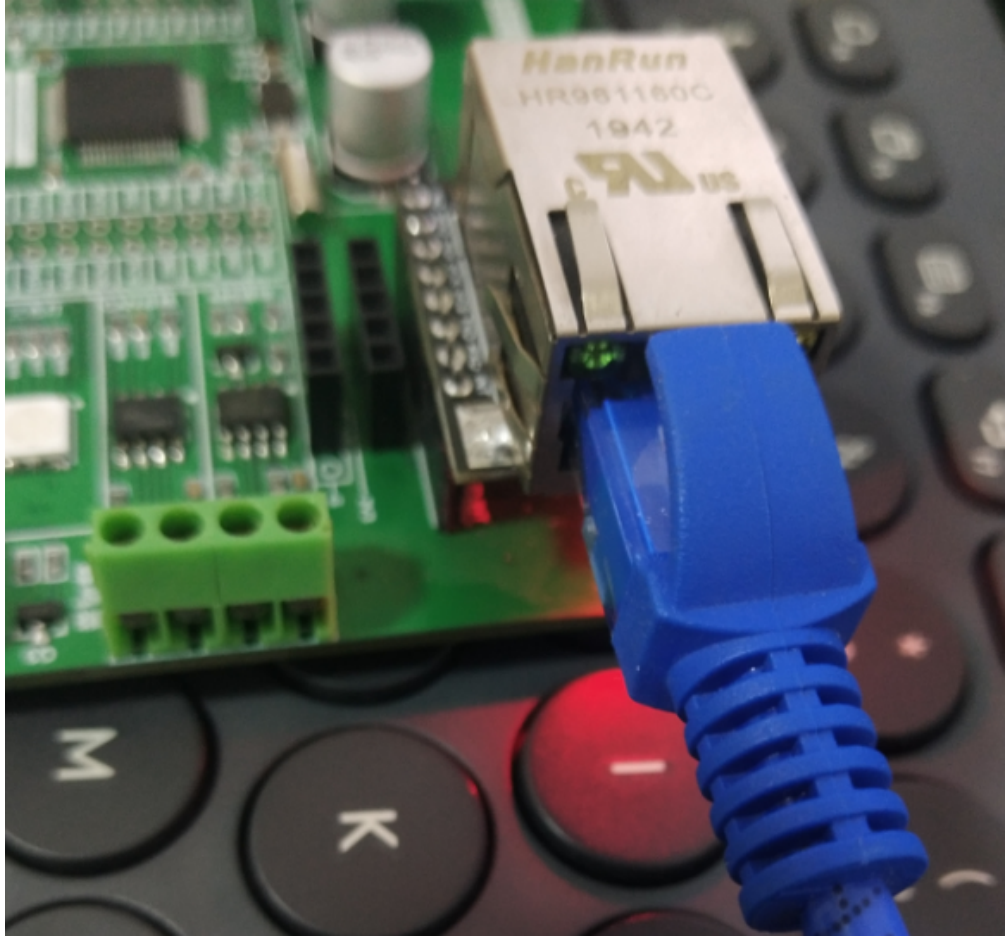
要想模块使用SPI通信,模块的TX引脚需要在模块重启之前设置为低电平.

上面的引脚分配把模块的TX引脚接到了单片机的PA3上,也就是串口2的RX上,如果用户使用了串口2,请注意!

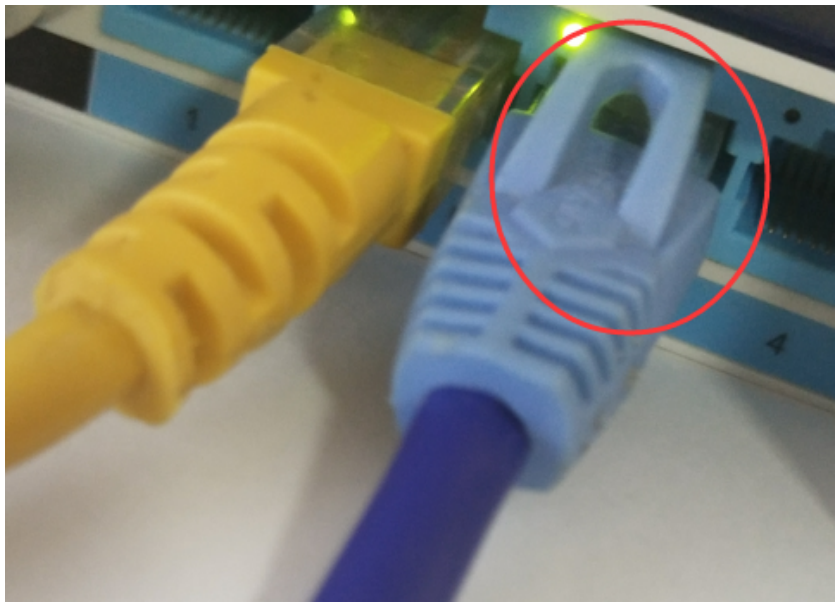
CH395 与单片机之间支持三种通讯接口：8 位并行接口、SPI 同步串行接口、异步串口。在芯片上电复位时，CH395 将采样 SEL 和 TXD 引脚的状态，根据这 2 个引脚状态的组合选择通讯接口，参考下表（表中 X 代表不关心此位，0 代表低电平，1 代表高电平或者悬空）。

SEL 引脚	TXD 引脚	选择通讯接口
1	1	异步串口
1	0	SPI 接口
0	1	8 位并口
0	0	错误接口

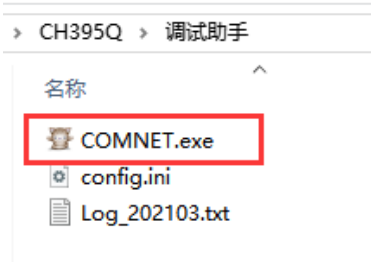
3.把模块用网线和路由器或者交换机(和上位机在同一个局域网下)



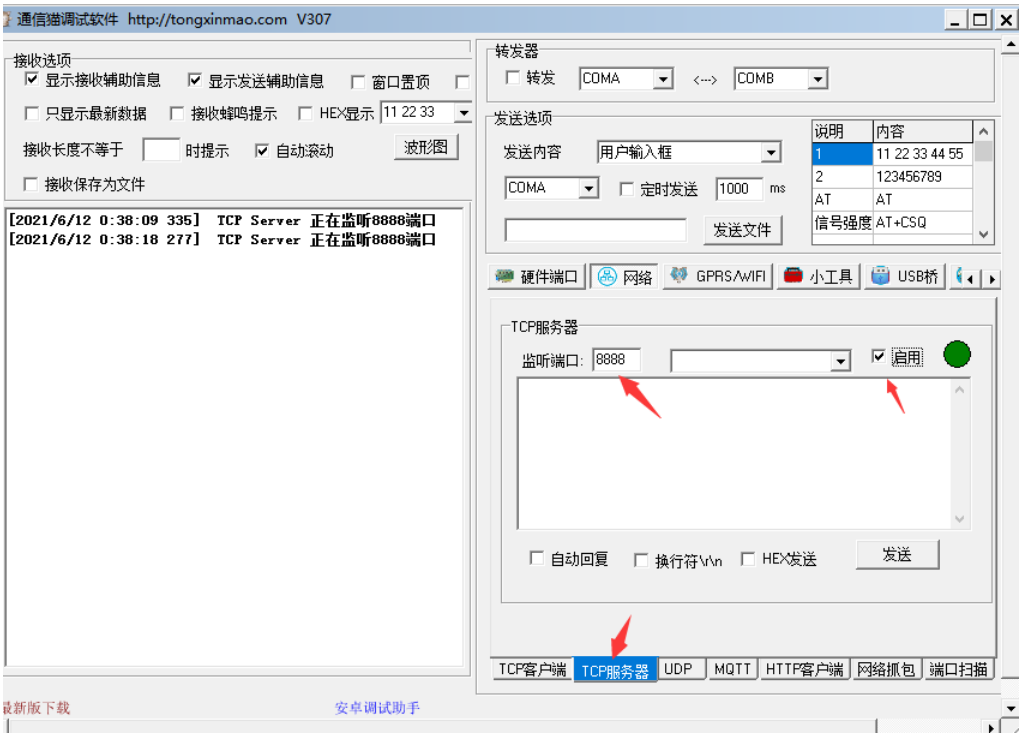
注意,连接路由器或者交换机的时候是连接其LAN口.



4,在电脑上运行网络调试助手,开启TCP服务器



我设置监听的端口为8888

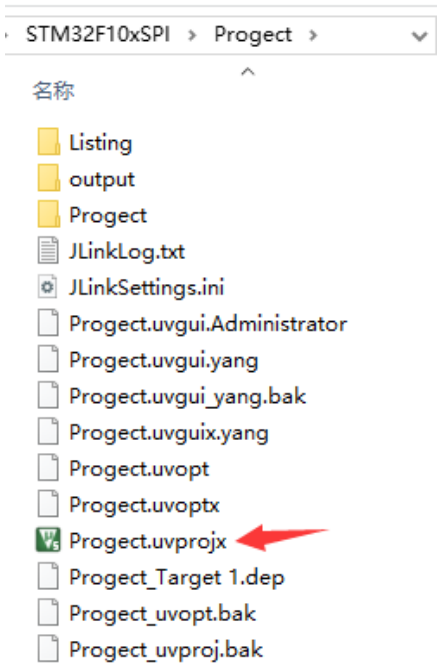


5.查看自己电脑的IP地址

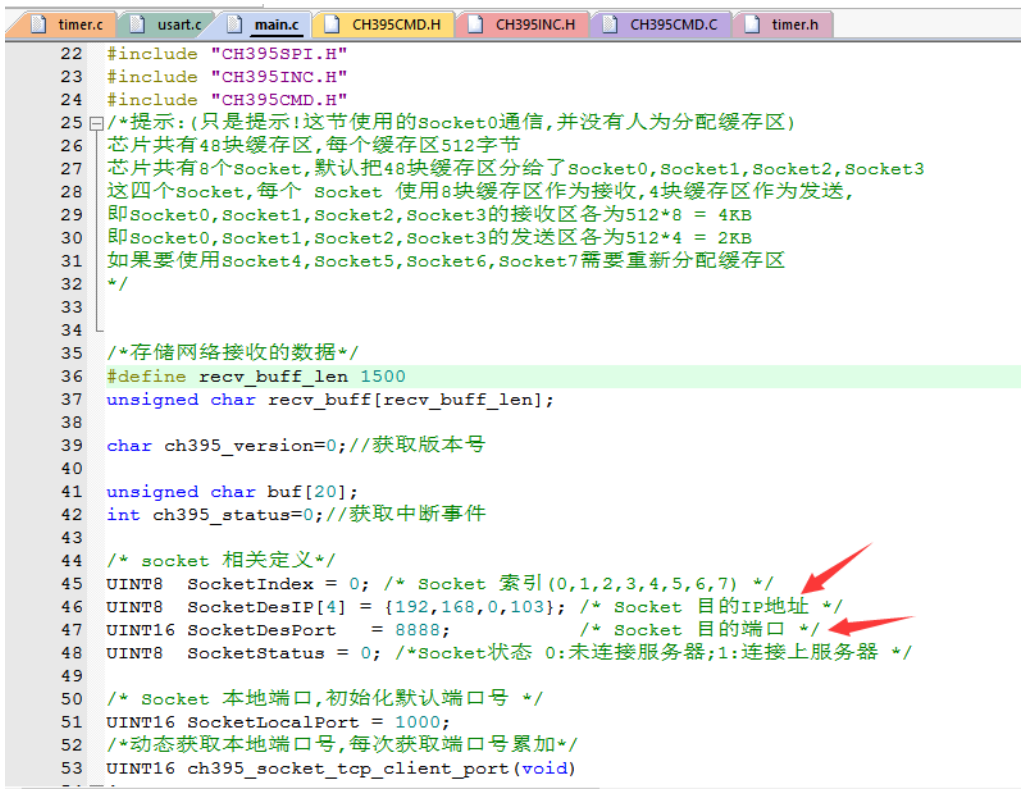
我的为 192.168.0.103

所以我的TCP服务器地址为 192.168.0.103,端口号为8888

6,打开这节程序

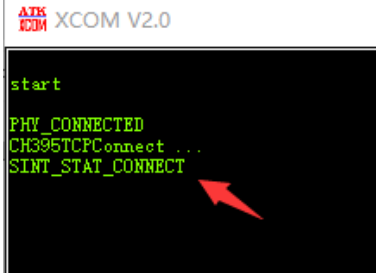


7,根据自己的修改服务器IP地址和端口号

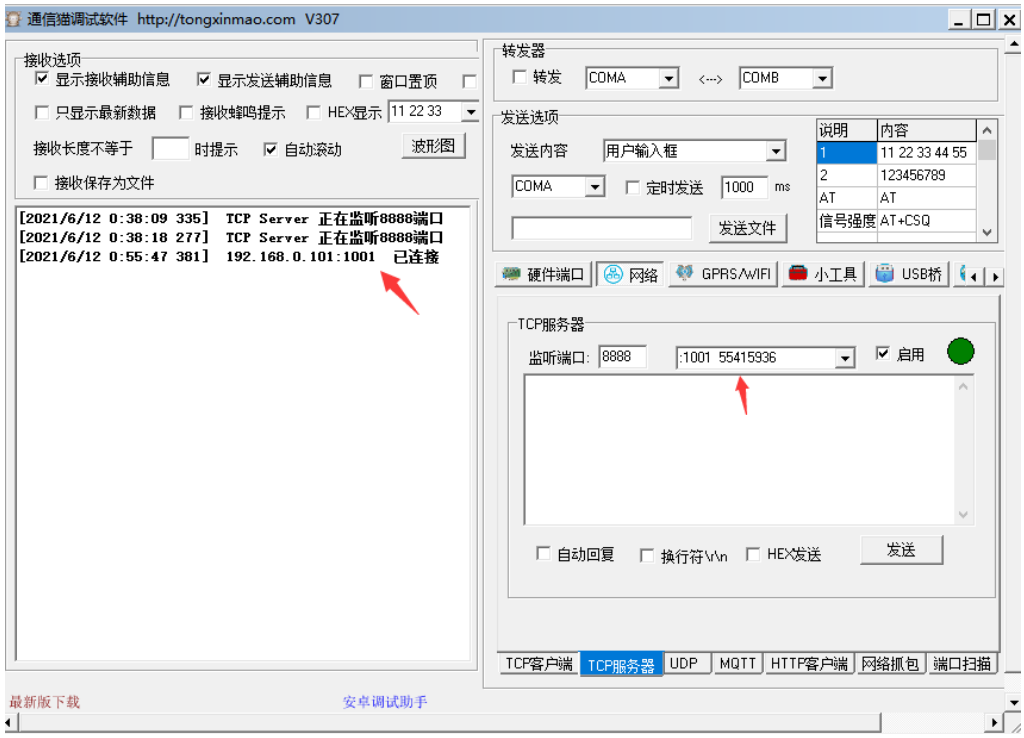


8.下载到单片机,单片机串口1作为日志打印口

连接上服务器会显示

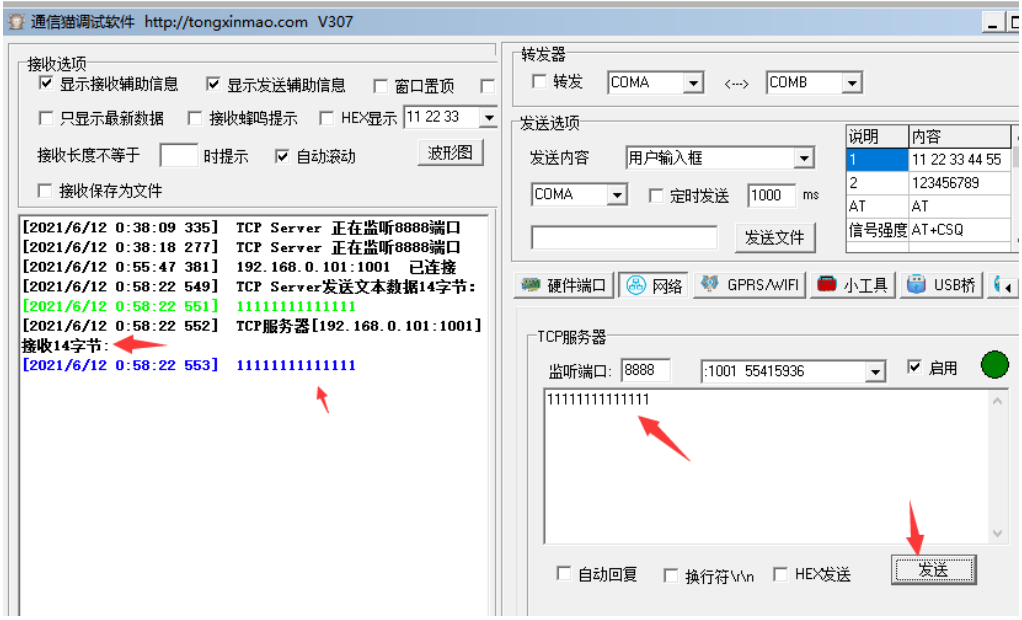


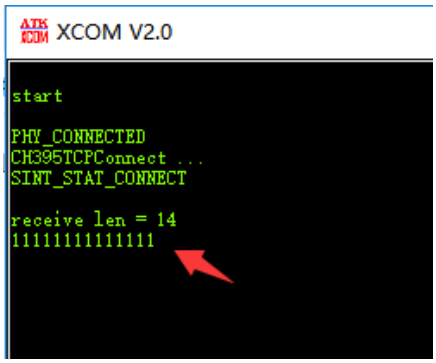
调试助手会显示连接



9.服务器给客户端发送消息

单片机程序里面写的是把接收的服务器返回给服务器,并使用串口打印接收的消息


通信猫调试软件 http://tongxinmao.com V307
接收选项
显示接收辅助信息
显示发送辅助信息
窗口置顶
只显示最新数据
接收蜂鸣提示
HEX显示
11 22 33
接收长度不等于
时提示
自动滚动
波形图
接收保存为文件
[2021/6/12 0:38:09 335] TCP Server 正在监听8888端口
[2021/6/12 0:38:18 277] TCP Server 正在监听8888端口
[2021/6/12 0:55:47 381] 192.168.0.101:1001 已连接
[2021/6/12 0:58:22 549] TCP Server发送文本数据14字节:
[2021/6/12 0:58:22 551] 11111111111111
[2021/6/12 0:58:22 552] TCP服务器[192.168.0.101:1001]
接收14字节:
[2021/6/12 0:58:22 553] 11111111111111
转发器
转发
COMA
<-->
COMB
发送选项
发送内容
用户输入框
COMA
定时发送
1000
ms
发送文件
说明
内容
1
11 22 33 44 55
2
123456789
AT
AT
信号强度
AT+CSQ
硬件端口
网络
GPRS/WIFI
小工具
USB桥
TCP服务器
监听端口:
8888
:1001 55415936
启用
11111111111111
自动回复
换行符\r\n
HEX发送
发送


XCOM V2.0
start
PHY_CONNECTED
CH395TCPConnect ...
SINT_STAT_CONNECT
receive len = 14
11111111111111

程序说明

1.初始化

timer.c | usart.c | main.c | CH395CMD.H | CH395INC.H | CH3

```c
int main(void)
{
  NVIC_Configuration();
  uart_init(115200);   //串口初始化为115200
  delay_init();
  timer2_config();

  //初始化CH395使用的GPIO
  CH395_PORT_INIT();
  //复位 CH395
  CH395_RST();

  IWDG_Init(IWDG_Prescaler_256,156*10);

  /*获取芯片版本*/
  while((ch395_version = CH395CMDGetVer()) < 0x40)
  {
    printf("CH395CMDGetVer ERR\r\n");
    delay_ms(100);
  }

  /*测试命令，按位取反返回说明测试通过*/
  while(CH395CMDCheckExist(0x55) != 0xaa)
  {
    printf("\r\nCH395CMDCheck ERR\r\n");
    delay_ms(100);
  }

  /*初始化模块:成功返回 0 */
  while(CH395CMDInitCH395() != 0)
  {
    printf("\r\nCH395CMDInitCH395 ERR\r\n");
    delay_ms(100);
  }

  printf("\r\nstart\r\n");
  while(1)
  {
    IWDG_Feed();//喂狗
```

2.因为是局域网,连接了路由器,所以需要启用DHCP

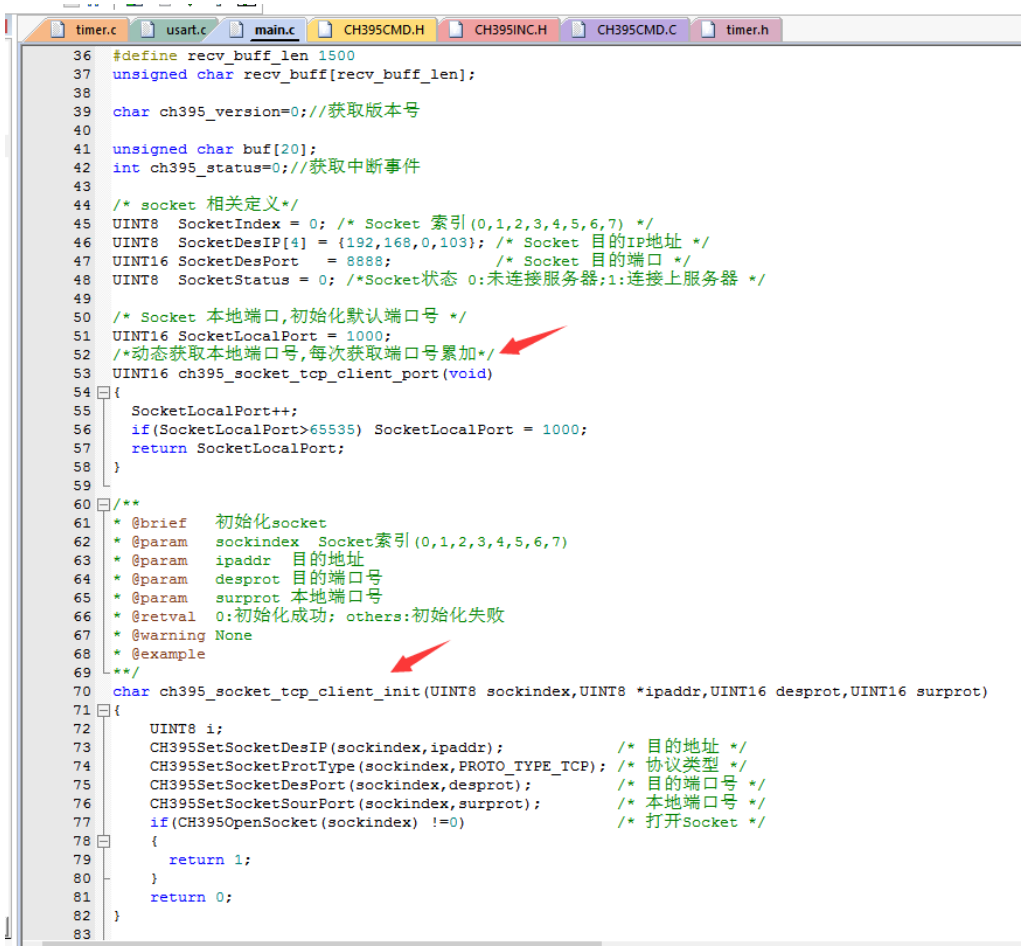
timer.c | usart.c | main.c | CH395CMD.H | CH395INC.H | CH395CMD.C | timer.

```
207
208     //INT引脚产生低电平中断以后进去判断
209     if(Query395Interrupt())
210     {
211       /*获取中断事件*/
212       if(ch395_version>=0x44)
213       {
214         ch395_status = CH395CMDGetGlobIntStatus_ALL();
215       }
216       else
217       {
218         ch395_status = CH395CMDGetGlobIntStatus();
219       }
220
221       /* 处理PHY改变中断*/
222       if(ch395_status & GINT_STAT_PHY_CHANGE)
223       {
224         if(CH395CMDGetPHYStatus() == PHY_DISCONN)//网线断开
225         {
226           printf("\r\nPHY_DISCONN\r\n");
227         }
228         else//网线连接
229         {
230           printf("\r\nPHY_CONNECTED\r\n");
231           CH395DHCPEnable(1);//启动DHCP
232         }
233       }
234
235       /* 处理DHCP/PPPOE中断 */
236       if(ch395_status & GINT_STAT_DHCP)
237       {
238         if(CH395GetDHCPStatus() == 0)//DHCP OK
239         {
240         }
241       }
242
243       /* 处理不可达中断，读取不可达信息 */
244       if(ch395_status & GINT_STAT_UNREACH){
245         CH395CMDGetUnreachIPPT(buf);
246       }
247
248       /* 处理IP冲突中断，建议重新修改CH395的 IP，并初始化CH395*/
249       if(ch395_status & GINT_STAT_IP_CONFLI){
250
251       }
252       /* 处理 SOCK0 中断 */
253       if(ch395_status & GINT_STAT_SOCK0){
254         ch395_socket_tcp_client_interrupt(SocketIndex);
```

3.每隔8S判断,如果Socket没有连接,则初始化Socket和控制Socket连接服务器

timer.c | usart.c | main.c | CH395CMD.H | CH395INC.H | CH395CMD.C | timer.h

```
189   {
190     IWDG_Feed();//喂狗
191
192     /*每隔8S初始化Socket并执行连接函数*/
193     /*芯片内部连接超时时间为5S,必须大于此时间*/
194     if(Timer2Cnt>8000)
195     {
196       Timer2Cnt = 0;
197       if(!SocketStatus)
198       {
199         if(ch395_socket_tcp_client_init(SocketIndex,SocketDesIP,SocketDesPort,ch395_socket_tcp_client_port()) == 0)
200         {
201           printf("CH395TCPConnect ... \r\n");
202           CH395TCPConnect(SocketIndex);//连接服务器
203         }
204       }
205     }
```



4.在中断检测事件里面处理Socket相关事件

timer.c | usart.c | main.c | CH395CMD.H | CH395INC.H | CH395CMD.C | timer.h

```
234
235         /* 处理DHCP/PPPOE中断 */
236         if(ch395_status & GINT_STAT_DHCP)
237         {
238           if(CH395GetDHCPStatus() == 0)//DHCP OK
239           {
240           }
241         }
242
243         /* 处理不可达中断，读取不可达信息 */
244         if(ch395_status & GINT_STAT_UNREACH){
245           CH395CMDGetUnreachIPPT(buf);
246         }
247
248         /* 处理IP冲突中断，建议重新修改CH395的 IP，并初始化CH395*/
249         if(ch395_status & GINT_STAT_IP_CONFLI){
250
251         }
252         /* 处理 SOCK0 中断 */
253         if(ch395_status & GINT_STAT_SOCK0){
254           ch395_socket_tcp_client_interrupt(SocketIndex);
255         }
256         /* 处理 SOCK1 中断 */
257         if(ch395_status & GINT_STAT_SOCK1){
258
259         }
260         /* 处理 SOCK2 中断 */
261         if(ch395_status & GINT_STAT_SOCK2){
262
263         }
264         /* 处理 SOCK3 中断 */
265         if(ch395_status & GINT_STAT_SOCK3){
266
267         }
```

timer.c | usart.c | main.c | CH395CMD.H | CH395INC.H | CH395CMD.C | timer.h

```
83
84
85  /**
86   * @brief   socket处理函数(把此函数放到全局socket中断里面)
87   * @param   sockindex  Socket索引(0,1,2,3,4,5,6,7)
88   * @param   None
89   * @param   None
90   * @param   None
91   * @retval  None
92   * @warning None
93   * @example
94   **/
95  void ch395_socket_tcp_client_interrupt(UINT8 sockindex)
96  {
97    UINT8  sock_int_socket;
98    UINT16 len;
99
100   /* 获取socket 的中断状态 */
101   sock_int_socket = CH395GetSocketInt(sockindex);
102
103   /* 发送缓冲区空闲，可以继续写入要发送的数据 */
104   if(sock_int_socket & SINT_STAT_SENBUF_FREE)
105   {
106   }
107
108   /* 发送完成中断 */
109   if(sock_int_socket & SINT_STAT_SEND_OK)
110   {
111   }
112
113   /* 接收数据中断 */
114   if(sock_int_socket & SINT_STAT_RECV)
115   {
116     len = CH395GetRecvLength(sockindex);/* 获取当前缓冲区内数据长度 */
117     printf("\r\nreceive len = %d\r\n",len);
118     if(len == 0)return;
119     if(len > recv_buff_len)len = recv_buff_len;
120     CH395GetRecvData(sockindex,len,recv_buff);/* 读取数据 */
121
122     /*把接收的数据发送给服务器*/
123     CH395SendData(sockindex,recv_buff,len);
124
125     /*使用串口打印接收的数据*/
```

```
124
125     /*使用串口打印接收的数据*/
126     PutData(&rb_t_usart1_send,recv_buff,len);
127     USART_ITConfig(USART1, USART_IT_TXE, ENABLE);
128   }
129
130   /* 连接中断，仅在TCP模式下有效*/
131   if(sock_int_socket & SINT_STAT_CONNECT)
132   {
133     SocketStatus = 1;//设置连接状态为连接
134     printf("SINT_STAT_CONNECT\n");
135   }
136
137   /* 断开中断，仅在TCP模式下有效 */
138   if(sock_int_socket & SINT_STAT_DISCONNECT)
139   {
140     printf("SINT_STAT_DISCONNECT \n");
141     SocketStatus = 0;//设置连接状态为未连接
142   }
143
144   /* 超时中断，仅在TCP模式下有效 ,TCP CLIENT无法顺利连接服务器端会进入此中断*/
145   if(sock_int_socket & SINT_STAT_TIM_OUT)
146   {/*此时可以把Socket源端口号进行自加处理，以新的端口去连接服务器*/
147     printf("SINT_STAT_TIM_OUT\n");
148     SocketStatus = 0;//设置连接状态为未连接
149   }
150 }
151
```

5.提示

在执行完CH395TCPConnect(SocketIndex);//连接服务器

以后,如果连接成功,会进入Socket连接成功回调,然后就是正常通信

```
196       Timer2Cnt = 0;
197       if(!SocketStatus)
198       {
199         if(ch395_socket_tcp_client_init(SocketIndex,Sock
200         {
201           printf("CH395TCPConnect ... \r\n");
202           CH395TCPConnect(SocketIndex);//连接服务器
203         }
204       }
205     }
206
```

timer.c | usart.c | main.c | CH395CMD.H | CH395INC.H | CH395CMD.C | timer.h

```
122     /*把接收的数据发送给服务器*/
123     CH395SendData(sockindex,recv_buff,len);
124
125     /*使用串口打印接收的数据*/
126     PutData(&rb_t_usart1_send,recv_buff,len);
127     USART_ITConfig(USART1, USART_IT_TXE, ENABLE);
128   }
129
130   /* 连接中断，仅在TCP模式下有效*/
131   if(sock_int_socket & SINT_STAT_CONNECT)
132   {
133     SocketStatus = 1;//设置连接状态为连接
134     printf("SINT_STAT_CONNECT\n");
135   }
136
137   /* 断开中断，仅在TCP模式下有效 */
138   if(sock_int_socket & SINT_STAT_DISCONNECT)
139   {
140     printf("SINT_STAT_DISCONNECT \n");
141     SocketStatus = 0;//设置连接状态为未连接
142   }
143
144   /* 超时中断，仅在TCP模式下有效 ,TCP CLIENT无法顺利连接服务器端会进入此中断*/
145   if(sock_int_socket & SINT_STAT_TIM_OUT)
146   {/*此时可以把Socket源端口号进行自加处理，以新的端口去连接服务器*/
147     printf("SINT_STAT_TIM_OUT\n");
148     SocketStatus = 0;//设置连接状态为未连接
149   }
150 }
151
```

在执行完CH395TCPConnect(SocketIndex);//连接服务器

以后,如果连接超时(默认5S)会进入Socket连接超时回调

然后程序到了8S之后便会再次尝试连接.

timer.c | usart.c | main.c | CH395CMD.H | CH395INC.H | CH395CMD.C | timer.h

```
128   }
129
130   /* 连接中断，仅在TCP模式下有效*/
131   if(sock_int_socket & SINT_STAT_CONNECT)
132   {
133     SocketStatus = 1;//设置连接状态为连接
134     printf("SINT_STAT_CONNECT\n");
135   }
136
137   /* 断开中断，仅在TCP模式下有效 */
138   if(sock_int_socket & SINT_STAT_DISCONNECT)
139   {
140     printf("SINT_STAT_DISCONNECT \n");
141     SocketStatus = 0;//设置连接状态为未连接
142   }
143
144   /* 超时中断，仅在TCP模式下有效 ,TCP CLIENT无法顺利连接服务器端会进入此中断*/
145   if(sock_int_socket & SINT_STAT_TIM_OUT)
146   {/*此时可以把Socket源端口号进行自加处理，以新的端口去连接服务器*/
147     printf("SINT_STAT_TIM_OUT\n");
148     SocketStatus = 0;//设置连接状态为未连接
149   }
150 }
151
```

6.如果在其它位置发送数据,推荐的方式

```
/*Socket连接*/
if(SocketStatus)
{
  CH395SendData(SocketIndex,"111111",6);//发送数据
}
```

分类: CH395Q学习开发

好文要顶 关注我 收藏该文

杨奉武
关注 - 1
粉丝 - 607

0 0

« 上一篇： 4-网络芯片CH395Q学习开发-关于中断检测和DHCP实验

posted on 2021-06-12 01:16 杨奉武 阅读(0) 评论(0) 编辑 收藏 举报

刷新评论 刷新页面 返回顶部

发表评论

编辑 预览

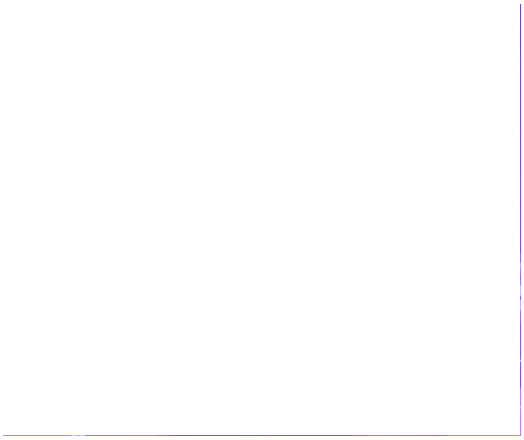
支持 Markdown

自动补全

提交评论 退出

[Ctrl+Enter快捷键提交]

【推荐】百度智能云618年中大促，限时抢购，新老用户同享超值折扣
【推荐】大型组态、工控、仿真、CAD\GIS 50万行VC++源码免费下载!
【推荐】阿里云爆品销量榜单出炉，精选爆款产品低至0.55折
【推荐】限时秒杀！国云大数据魔镜，企业级云分析平台
【推荐】华为应用软件专题日 | 生态市场企业特惠GO



园子动态：
· 致园友们的一封检讨书：都是我们的错
· 数据库实例 CPU 100% 引发全站故障
· 发起一个开源项目：博客引擎 fluss

最新新闻：
· 2021 苹果设计奖出炉！《原神》《英雄联盟》获奖，还有 10 个年度 App
· K 歌、主播、B 站 UP 主…… 这一届「银发族」如何开启第二人生
· 黑客可向邮件服务器发送数据以干扰HTTPS连接
· AI产品正逐渐走向成熟，行业竞争已经出现两极分化
· 解读滴滴招股书：提供“移动”价值的全球共享经济企业潜力几何？
» 更多新闻...

Powered by:
博客园
Copyright © 2021 杨奉武
Powered by .NET 5.0 on Kubernetes

单片机,物联网,上位机,…

扫一扫二维码，加入群聊。