

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 - 701, 文章 - 0, 评论 - 311, 阅读 - 173万

导航

博客园

首页

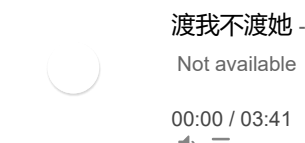
新随笔

联系

订阅 

管理

公告



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

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(13)
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)

13-网络芯片CH395Q学习开发-模块使用Socket0作为MAC

RAW

```
<p><iframe name="ifd" src="https://mnifdv.cn/resource/cnblogs/LearnCH395Q"
frameborder="0" scrolling="auto" width="100%" height="1500"></iframe></p>
```

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.c>

- [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](#)
- [6-模块使用Socket0-3作为4路TCP客户端和电脑上位机TCP服务器局域网通信](#)
- [7-模块使用Socket0-5作为6路TCP客户端和电脑上位机TCP服务器局域网通信\(Socket缓存区配置\)](#)
- [8-模块使用Socket0作为TCP服务器和电脑上位机TCP客户端局域网通信\(单连接和多连接\)](#)
- [9-模块使用Socket0作为UDP和电脑上位机UDP局域网通信](#)
- [10-模块使用Socket0作为UDP广播通信](#)
- [11-模块使用Socket0作为UDP组播\(多播\)通信MAC地址过滤](#)

ESP8266 LUA开发基础入门篇
备份(22)
ESP8266 SDK开发(32)
ESP8266 SDK开发基础入门篇
备份(30)
GPRS Air202 LUA开发(11)
HC32F460(华大) +
BC260Y(NB-IOT) 物联网开发
(5)
NB-IOT Air302 AT指令和LUA
脚本语言开发(25)
PLC(三菱PLC)基础入门篇(2)
STM32+Air724UG(4G模组)
物联网开发(43)
STM32+BC26/260Y物联网开
发(37)
STM32+ESP8266(ZLESP8266/
物联网开发(1)
STM32+ESP8266+AIR202/30:
远程升级方案(16)
STM32+ESP8266+AIR202/30:
终端管理方案(6)
STM32+ESP8266+Air302物
联网开发(58)
STM32+W5500+AIR202/302
基本控制方案(25)
STM32+W5500+AIR202/302
远程升级方案(6)
UCOSii操作系统(1)
W5500 学习开发(8)
编程语言C#(11)
编程语言Lua脚本语言基础入
门篇(6)
编程语言Python(1)
单片机(LPC1778)LPC1778(2)
单片机(MSP430)开发基础入门
篇(4)
单片机(STC89C51)单片机开发
板学习入门篇(3)
单片机(STM32)基础入门篇(3)
单片机(STM32)综合应用系列
(16)
电路模块使用说明(10)
感想(6)
软件安装使用: MQTT(8)
软件安装使用: OpenResty(6)
数据处理思想和程序架构(24)
数据库学习开发(12)
更多

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

1. ESP8266使用详解(AT,LUA,
SDK)(172088)
2. 1-安装MQTT服务器(Windo
ws),并连接测试(96516)
3. ESP8266刷AT固件与node
mcu固件(63771)

- 信,MAC地址过滤
- 12-模块使用Socket0作为IP RAW模式和调试助手
测试通信
- 13-模块使用Socket0作为MAC RAW
-
-
-

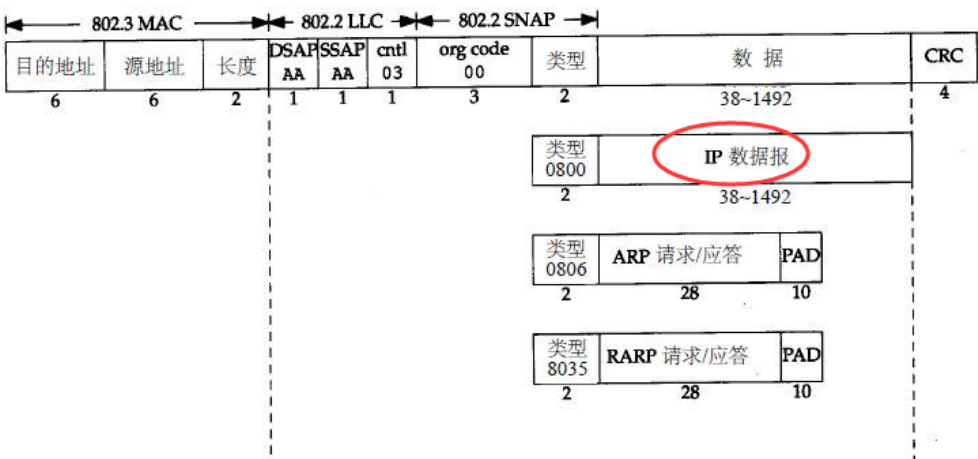
什么是MAC RAW

MAC RAW也叫以太网数据原始数据,是IP数据下一层的数据.

目的 MAC	源 MAC	类型	数据	CRC32
6 Byte	6 Byte	2 Byte	46-1500 Byte	4 Byte

咱们操作最顶层的TCP或UDP接口,然后数据会使用IP层函数封装

最后IP层的再放到MAC层封装



4. 用ESP8266+android,制作自己的WIFI小车(ESP8266篇)(62599)
5. 有人WIFI模块使用详解(38101)
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)(35393)
7. 关于TCP和MQTT之间的转换(32237)
8. android 之TCP客户端编程(31292)
9. android服务端+eps8266+单片机+路由器之远程控制系统(31138)
10. C#中public与private与static(30952)

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)

说明

这节演示一下模块使用Socket0作为MAC RAW模式

这节只是开启MAC RAW模式,然后把接收的网络数据用串口打印.

注意:实现MAC RAW是需要设置MAC地址的,但是模块内部自带全球唯一MAC,所以不需要设置.

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(STM32F103xxxx)

1.用户可以使用杜邦线根据自己的情况设置和连接引脚

```
ch395cmd.h  CH395INC.H  CH395SPI.C  usart.c  delay.c  timer.c  main.c  delay.h  CH395SPLH
2  #ifndef __CH395SPI_H_
3  #define __CH395SPI_H_
4
5  #include "CH395INC.H"
6
7  /*****配置GPIO (根据自己的修改)*****/
8  //时钟
9  #define CH395_CONFIG_SPI_CLK() ( RCC_APB1PeriphClockCmd( RCC_APB1Periph_SPI2,ENABLE) )
10 #define CH395_CONFIG_GPIO_CLK() ( RCC_APB2PeriphClockCmd( RCC_APB2Periph_GPIOA | RCC_APB2Peri
11 //设置使用的SPI
12 #define USE_SPI SPI2
13 //SPI_CS -- 连接模块SCS引脚
14 #define CH395_CS_PORT GPIOB
15 #define CH395_CS_PIN GPIO_Pin_12
16 //SPI_CLK -- 连接模块SCK引脚
17 #define CH395_CLK_PORT GPIOB
18 #define CH395_CLK_PIN GPIO_Pin_13
19 //SPI_MISO -- 连接模块SDO引脚
20 #define CH395_MISO_PORT GPIOB
21 #define CH395_MISO_PIN GPIO_Pin_14
22 //SPI_MOSI -- 连接模块SDI引脚
23 #define CH395_MOSI_PORT GPIOB
24 #define CH395_MOSI_PIN GPIO_Pin_15
25 //RST -- 连接模块RST引脚
26 #define CH395_RST_PORT GPIOA
27 #define CH395_RST_PIN GPIO_Pin_8
28 //TX -- 连接模块Tx引脚
29 #define CH395_TX_PORT GPIOA
30 #define CH395_TX_PIN GPIO_Pin_3
31 //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)
32 #define CH395_INT_PORT GPIOA
33 #define CH395_INT_PIN GPIO_Pin_0
34 /*****/
```

2,注意!

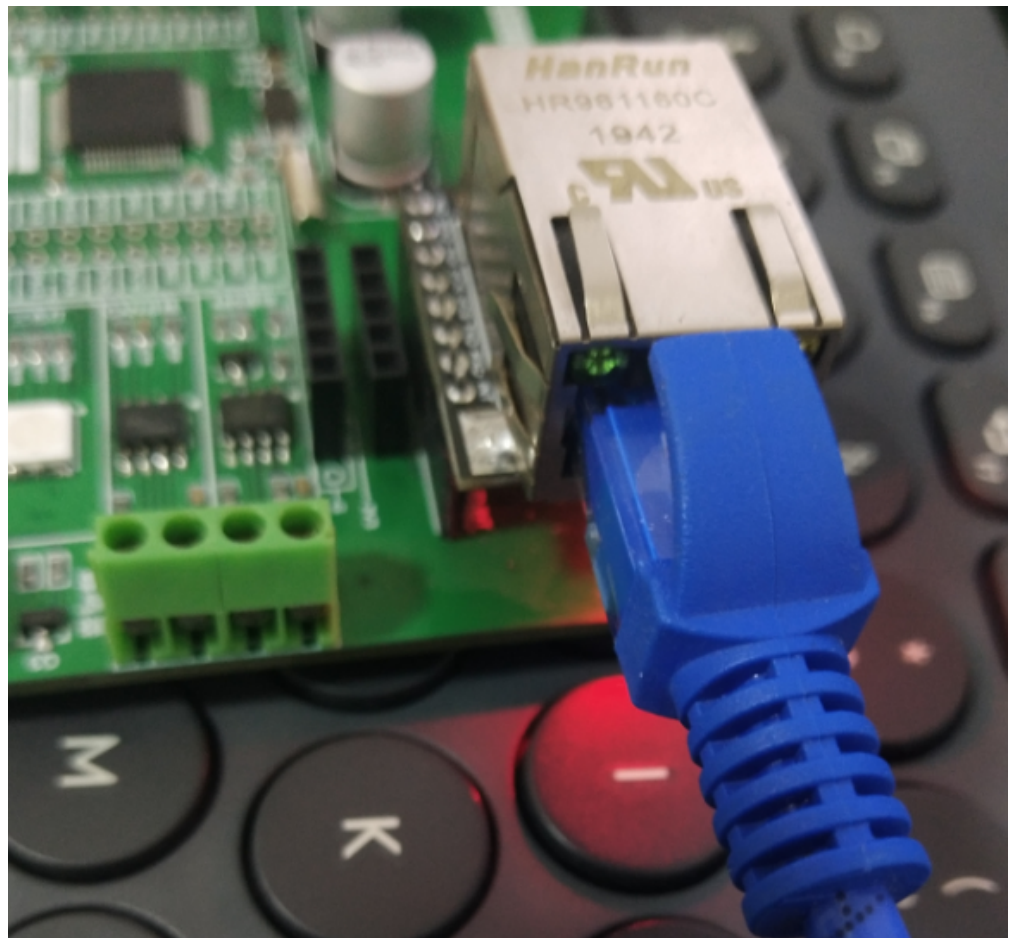
要想模块使用SPI通信,模块的TX引脚需要在模块重启之前设置为低电平.

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.



WAN端口：连接网线

LAN端口：连接电脑（任选一个端口就行）

7. 下载程序到单片机

正常情况不时的打印一些数据,这些数据具体是啥我也没去研究
会打印路由器的MAC,网关地址,网卡名称,



程序说明

1.初始化MAC RAW

```
timer.c  usart.c  main.c  CH395CMD.H  CH395INC.H  CH395CMD.C  timer.h
141 IWDG_Init(IWDG_Prescaler_256,156*10);
142
143 /*获取芯片版本*/
144 while((ch395_version = CH395CMDGetVer()) < 0x40)
145 {
146     printf("CH395CMDGetVer ERR\r\n");
147     delay_ms(100);
148 }
149
150 /*测试命令, 按位取反返回说明测试通过*/
151 while(CH395CMDCheckExist(0x55) != 0xaa)
152 {
153     printf("\r\nCH395CMDCheck ERR\r\n");
154     delay_ms(100);
155 }
156
157 /*初始化模块:成功返回 0 */
158 while(CH395CMDInitCH395() != 0)
159 {
160     printf("\r\nCH395CMDInitCH395 ERR\r\n");
161     delay_ms(100);
162 }
163
164 CH395CMDGetMACAddr(buf);
165 printf("\r\nCH395MAC %02x:%02x:%02x:%02x:%02x:%02x\r\n",buf[0],buf[1],buf[2],buf[3],buf[4],buf[5]);
166
167 /*初始化MAC RAW*/
168 ch395_socket_tcp_client_init(SocketIndex);
169
170 printf("\r\nstart\r\n");
171 while(1)
172 {
173     IWDG_Feed(); //喂狗
174 }
```

```
timer.c  usart.c  main.c  CH395CMD.H  CH395INC.H  CH395CMD.C  timer.h
39 char ch395_version=0; //获取版本号
40
41 unsigned char buf[20];
42 int ch395_status=0; //获取中断事件
43
44 /* socket 相关定义 */
45 UINT8 SocketIndex = 0; /* Socket 索引, 仅Socket0支持 */
46
47 /**
48  * @brief 初始化socket
49  * @param sockindex Socket索引(0,1,2,3,4,5,6,7)
50  * @param ipaddr 目的地址
51  * @param desprot 目的端口号
52  * @param surprot 本地端口号
53  * @retval 0:初始化成功; others:初始化失败
54  * @warning None
55  * @example
56  */
57 char ch395_socket_tcp_client_init(UINT8 sockindex)
58 {
59     CH395SetSocketProtType(sockindex,PROTO_TYPE_MAC_RAW); /* 协议类型 */
60     if(CH395OpenSocket(sockindex) !=0) /* 打开Socket */
61     {
62         return 1;
63     }
64     return 0;
65 }
66
```

2.因为连接路由器,所以需要DHCP

```
timer.c  usart.c  main.c  CH395CMD.H  CH395INC.H  CH395CMD.C  timer.h
189
190 //INT引脚产生低电平中断以后进去判断
191 if(Query395Interrupt())
192 {
193     /*获取中断事件*/
194     if(ch395_version>=0x44)
195     {
196         ch395_status = CH395CMDGetGlobIntStatus_ALL();
197     }
198     else
199     {
200         ch395_status = CH395CMDGetGlobIntStatus();
201     }
202
203     /* 处理PHY改变中断*/
204     if(ch395_status & GINT_STAT_PHY_CHANGE)
205     {
206         if(CH395CMDGetPHYStatus() == PHY_DISCONN)//网线断开
207         {
208             printf("\r\nPHY_DISCONN\r\n");
209         }
210         else//网线连接
211         {
212             printf("\r\nPHY_CONNECTED\r\n");
213             CH395DHCPEnable(1);//启动DHCP
214         }
215     }
216
217     /* 处理DHCP/PPPOE中断 */
218     if(ch395_status & GINT_STAT_DHCP)
219     {
220         if(CH395GetDHCPStatus() == 0)//DHCP OK
221         {
222             CH395GetIPInf(buf);//获取IP地址,网关和子网掩码
223             printf("IP:%d.%d.%d.%d\r\n",buf[0],buf[1],buf[2],buf[3]);
224             printf("GWIP:%d.%d.%d.%d\r\n",buf[4],buf[5],buf[6],buf[7]);
225             printf("Mask:%d.%d.%d.%d\r\n",buf[8],buf[9],buf[10],buf[11]);
226             printf("DNS1:%d.%d.%d.%d\r\n",buf[12],buf[13],buf[14],buf[15]);
227             printf("DNS2:%d.%d.%d.%d\r\n",buf[16],buf[17],buf[18],buf[19]);
228         }
229     }
```

3.在中断检测事件里面处理Socket相关事件(本例中使用的Socket 0)

```
timer.c  usart.c  main.c  CH395CMD.H  CH395INC.H  CH395CMD.C  timer.h
225     printf("Mask:%d.%d.%d.%d\r\n",buf[8],buf[9],buf[10],buf[11]);
226     printf("DNS1:%d.%d.%d.%d\r\n",buf[12],buf[13],buf[14],buf[15]);
227     printf("DNS2:%d.%d.%d.%d\r\n",buf[16],buf[17],buf[18],buf[19]);
228 }
229
230
231 /* 处理不可达中断,读取不可达信息 */
232 if(ch395_status & GINT_STAT_UNREACH){
233     CH395CMDGetUnreachIPPT(buf);
234 }
235
236 /* 处理IP冲突中断,建议重新修改CH395的 IP,并初始化CH395*/
237 if(ch395_status & GINT_STAT_IP_CONFLI){
238
239 }
240 /* 处理 sock0 中断 */
241 if(ch395_status & GINT_STAT_SOCKET0){
242     ch395_socket_tcp_client_interrupt(SocketIndex);
243 }
244 /* 处理 sock1 中断 */
245 if(ch395_status & GINT_STAT_SOCKET1){
246
247 }
```

```

timer.c  usart.c  main.c  CH395CMD.H  CH395INC.H  CH395CMD.C  timer.h
70  * @param sockindex Socket索引 (0,1,2,3,4,5,6,7)
71  * @param None
72  * @param None
73  * @param None
74  * @retval None
75  * @warning None
76  * @example
77  **/
78 void ch395_socket_tcp_client_interrupt(UINT8 sockindex)
79 {
80     UINT8 sock_int_socket;
81     UINT16 len;
82
83     /* 获取socket 的中断状态 */
84     sock_int_socket = CH395GetSocketInt(sockindex);
85
86     /* 发送缓冲区空闲，可以继续写入要发送的数据 */
87     if(sock_int_socket & SINT_STAT_SENBUF_FREE)
88     {
89     }
90
91     /* 发送完成中断 */
92     if(sock_int_socket & SINT_STAT_SEND_OK)
93     {
94     }
95
96     /* 接收数据中断 */
97     if(sock_int_socket & SINT_STAT_RECV)
98     {
99         len = CH395GetRecvLength(sockindex);/* 获取当前缓冲区内数据长度 */
100        if(len == 0) return;
101        if(len > recv_buff_len) len = recv_buff_len;
102        CH395GetRecvData(sockindex, len, recv_buff);/* 读取数据 */
103
104        /*使用串口打印接收的数据*/
105        PutData(&rb_t_usart1_send, recv_buff, len);
106        USART_ITConfig(USART1, USART_IT_TXE, ENABLE);
107    }
108
109    /* 连接中断，仅在TCP模式下有效*/
110    if(sock_int_socket & SINT_STAT_CONNECT)
111    {
112        printf("SINT_STAT_CONNECT\n");
113    }
114
115    /* 断开中断，仅在TCP模式下有效 */
116    if(sock_int_socket & SINT_STAT_DISCONNECT)
117    {
118        printf("SINT_STAT_DISCONNECT \n");
119    }
120
121    /* 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*/
122    if(sock_int_socket & SINT_STAT_TIM_OUT)
123    {
124        /*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*/
125        printf("SINT_STAT_TIM_OUT\n");
126    }
}

```

注意事项

在 MACRAW 模式下，CH395 会透明传输以太网和单片机之间的数据，不会对数据进行 TCP/IP 封装，CH395 接收数据时会以以太网冗余校验 CRC32 进行校验，如果校验错误，数据包不会转发给单片机。CH395 发送数据时会在数据包尾部加入以太网冗余校验 CRC32。单片机每次向 CH395 写入的数据长度不得大于 1514，CH395 会将单片机每次写入的数据封装成一帧数据进行发送。当 CH395 从以太网收到数据后会通知单片机，此时单片机应立即将所有数据从 CH395 内部接收缓冲区读走。

仅 Socket0 可以设置此模式，且其他 Socket 将不可用。

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: 12-网络芯片CH395Q学习开发-模块使用Socket0作为IP RAW模式和调试助手测试通信

posted on 2021-06-14 13:11 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

B



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：

- 李书福呼吁从效率优先向公平优先转变 要给员工职业尊严
 - 剑桥在Nature子刊发表最新研究：石墨烯可将硬盘容量提高十倍
 - 埃里克森心脏一度停止跳动：15分钟急救刷屏 “救命神器”火了
 - 王传福：劝雷军别造车是误读 正和小米洽谈造车合作
 - 体验评测米家新风空调：不仅全屋互联 它还会“呼吸”
- » 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，入群聊。