

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 - 700, 文章 - 0, 评论 - 311, 阅读 - 173万

### 导航

博客园

首页

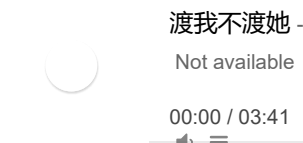
新随笔

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(12)  
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)

## 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

<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(96512)  
3. ESP8266刷AT固件与node  
mcu固件(63766)

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

- 
- 
- 
- 

## 什么是IP RAW

好多人常说:TCP/IP通信.我问下什么是TCP/IP?

咱们一般都是使用TCP或UDP的API函数做网络通信.这一层在网络通信中是最顶层.

这一层的下一层就是IP层,咱操作TCP或UDP的API函数的时候其实就是按照下面的格式进行打包.

IP层的数据是这样子的(IP RAW)

目的 MAC	源 MAC	类型	IP 首部	IPRAW 数据	CRC32
6 Byte	6 Byte	2 Byte	20 Byte	最大 1480 Bytes	4 Byte

假设使用TCP通信,然后客户端的IP为:192.168.0.102 端口号为1000 MAC地址为:84:C2:E4:EC:AC:43

假设服务器的IP为:192.168.0.103 端口号为6000 MAC地址为:F4:B5:20:09:8A:F9

- 4. 用ESP8266+android,制作自己的WIFI小车(ESP8266篇)(62597)
- 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)(35390)
- 7. 关于TCP和MQTT之间的转换(32237)
- 8. android 之TCP客户端编程(31291)
- 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(30951)

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)

那么咱们调用TCP发送数据API,实际打包的数据为: (IP层的数据)

目的MAC	源MAC	类型	整个IP首部
	(192.168.0.102)	(192.168.0.103)	
F4 B5 20 09 8A F9	84 C2 E4 EC AC 43	08 00	45 00 00 1A 00 04 00 00 80 06 B8 BC C0 A8 00 66 C0 A8 00 67
(1000) (6000) IPRAW数据部分 (后面还有各种标识然后还有真实数据)			
03E8	1770	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	

咱们在使用最上层的TCP或者UDP的API函数的时候,这些数据已经被解析到了最上层.

所以不需要关心IP层的数据.

当然再下一层就是物理层,其实就是用信号线做数据传输.网络的信号线和RS422类似,其中两根用差分信号做信号接收,另外两根用差分信号做信号发送.

说明

这节演示一下模块使用Socket0作为IP RAW模式和调试助手测试通信

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.

安装软件

一般的调试助手只有TCP和UDP最上层功能,如果要测试IP层通信,咱需要下载安装个软件

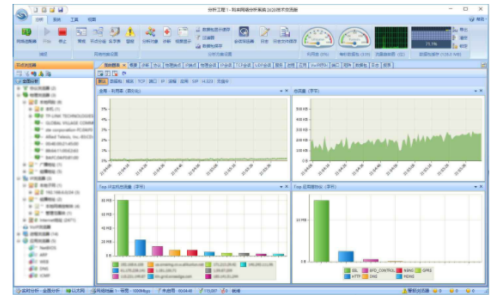
科来网络分析系统

## 科来网络分析系统 2020 (技术交流版)

科来于2001年首次发布CSNAS, 历经18年时间, 用户遍及全球, 是超百万人在工作、学习中使用的“常青藤”产品。CSNAS还曾被美国PC Magazine评选为《全球最佳科技产品》奖。

科来网络分析系统是网络故障分析、数字安全取证、协议分析学习等使用场景的“利器”。它无需复杂的部署工作, 当您有网络流量分析的需求时, 可直接安装在您的随行电脑中使用, 无论是固定节点使用, 还是临时需求, 都可以灵活、高效的帮助用户解决网络性能与安全方面的实际问题。

免费下载



## 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  CH395SPI.H
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!

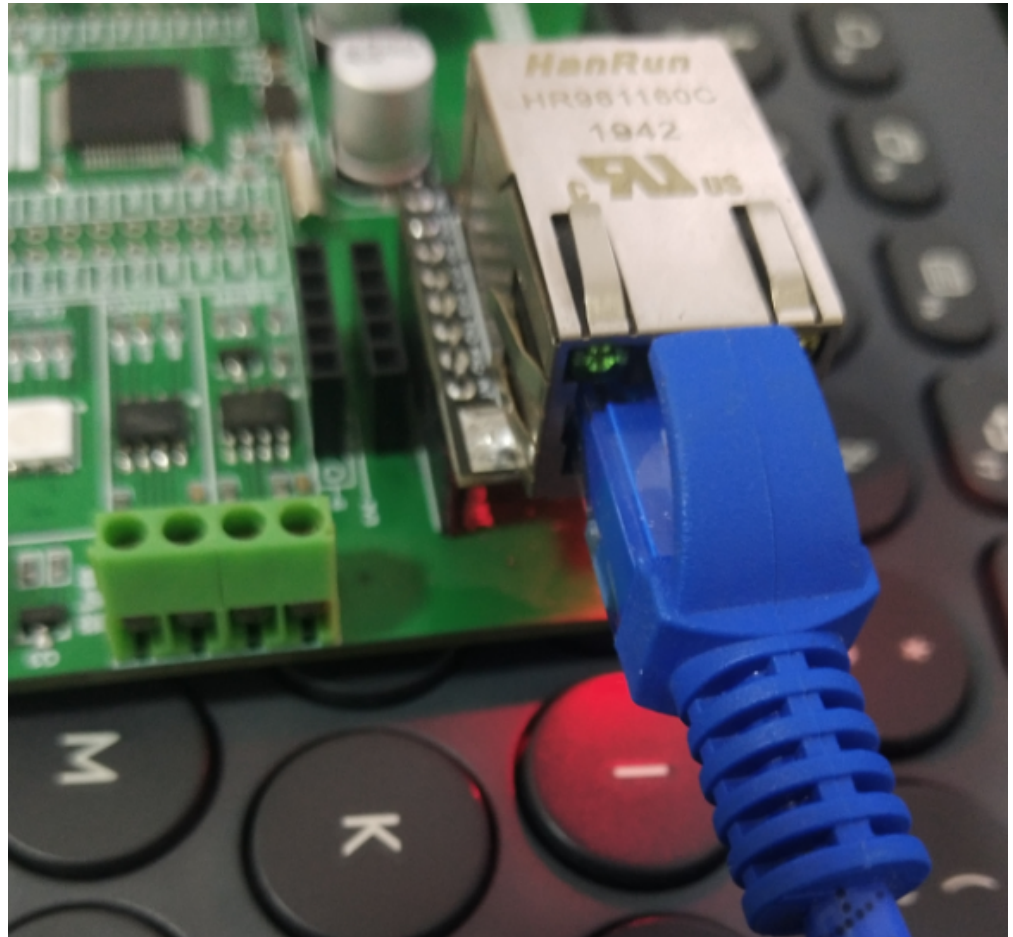
要想模块使用SPI通信,模块的TX引脚需要在模块重启之前设置为低电平.

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）。

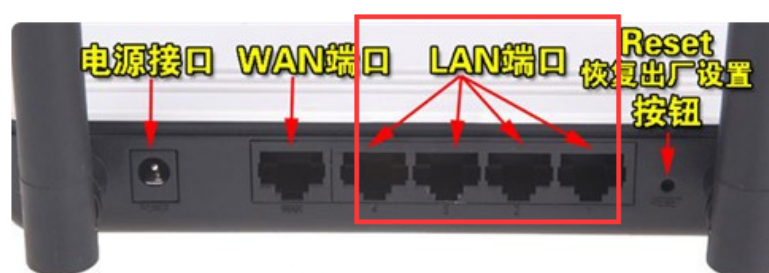
SEL 引脚	TXD 引脚	选择通讯接口
1	1	异步串口
1	0	SPI 接口
0	1	8 位并口
0	0	错误接口

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）**

#### 4.查看自己电脑的IP地址

我的为 192.168.0.103



## 5.修改为自己电脑的IP地址

```
timer.c  usart.c  main.c  CH395CMD.H  CH395INC.H  CH395CMD.C  timer.h
30  即Socket0,Socket1,Socket2,Socket3的发送区各为512*4 = 2KB
31  如果要使用Socket4,Socket5,Socket6,Socket7需要重新分配缓存区
32  */
33
34
35  /*存储网络接收的数据*/
36  #define recv_buff_len 1500
37  unsigned char recv_buff[recv_buff_len];
38
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(0,1,2,3,4,5,6,7) */
46  UINT8 SocketDesIP[4] = {192,168,0,103}; /* Socket 目的IP地址 */
47  const UINT8 IPRawProto = 0x06; /* IP包协议类型 */
48
49  /**
50  * @brief 初始化socket
51  * @param sockindex Socket索引(0,1,2,3,4,5,6,7)
52  * @param ipaddr 目的地址
53  * @param port 目的端口号
54  */
```

## 6.关于IP包协议类型(我写的是0x06,IP数据就是TCP数据)

最上层打包的时候如果是TCP那么到了IP层,协议类型就是0x06, 如果是UDP就是0x11 (17)

0 HOPOPT IPv6 逐跳选项

1 ICMP Internet 控制消息

2 IGMP Internet 组管理

3 GGP 网关对网关

4 IP IP 中的 IP ( 封装 )

5 ST 流

6 TCP 传输控制

7 CBT CBT

8 EGP 外部网关协议

9 IGP 任何专用内部网关  
( Cisco 将其用于 IGRP )

10 BBN-RCC-MON BBN RCC 监视

11 NVP-II 网络语音协议

12 PUP PUP

13 ARGUS ARGUS

14 EMCON EMCON

15 XNET 跨网调试器

16 CHAOS Chaos

17 UDP 用户数据报

18 MUX 多路复用

19 DCN-MEAS DCN 测量子系统

20 HMP 主机监视

21 PRM 数据包无线测量

22 XNS-IDP XEROX NS IDP

23 TRUNK-1 第 1 主干

24 TRUNK-2 第 2 主干

25 LEAF-1 第 1 叶

26 LEAF-2 第 2 叶

27 RDP 可靠数据协议

28 IRTP Internet 可靠事务

29 ISO-TP4 ISO 传输协议第 4 类

30 NETBLT 批量数据传输协议

31 MFE-NSP MFE 网络服务协议

32 MERIT-INP MERIT 节点间协议

33 SEP 顺序交换协议

34 3PC 第三方连接协议

35 IDPR 域间策略路由协议

36 XTP XTP



37 DDP 数据报传送协议

38 IDPR-CMTP IDPR 控制消息传输协议

39 TP++ TP++ 传输协议

40 IL IL 传输协议

41 IPv6 Ipv6

42 SDRP 源要求路由协议

43 IPv6-Route IPv6 的路由标头

44 IPv6-Frag IPv6 的片断标头

45 IDRP 域间路由协议

46 RSVP 保留协议

47 GRE 通用路由封装

48 MHRP 移动主机路由协议

49 BNA BNA

50 ESP IPv6 的封装安全负载

51 AH IPv6 的身份验证标头

52 I-NLSP 集成网络层安全性 TUBA

53 SWIPE 采用加密的 IP

54 NARP NBMA 地址解析协议

55 MOBILE IP 移动性

56 TLSP 传输层安全协议

使用 Kryptonet 密钥管理

57 SKIP SKIP

58 IPv6-ICMP 用于 IPv6 的 ICMP

59 IPv6-NoNxt 用于 IPv6 的无下一个标头

60 IPv6-Opts IPv6 的目标选项

61 任意主机内部协议

62 CFTP CFTP

63 任意本地网络

64 SAT-EXPAK SATNET 与后台 EXPAK

65 KRYPTOLAN Kryptolan

66 RVD MIT 远程虚拟磁盘协议

67 IPPC Internet Pluribus 数据包核心

68 任意分布式文件系统

69 SAT-MON SATNET 监视

70 VISA VISA 协议

71 IPCV Internet 数据包核心工具

72 CPNX 计算机协议网络管理

73 CPHB 计算机协议检测信号

74 WSN 王安电脑网络

75 PVP 数据包视频协议

76 BR-SAT-MON 后台 SATNET 监视

77 SUN-ND SUN ND PROTOCOL-Temporary

78 WB-MON WIDEBAND 监视

79 WB-EXPAK WIDEBAND EXPAK

80 ISO-IP ISO Internet 协议

81 VMTP VMTP

82 SECURE-VMTP SECURE-VMTP

83 VINES VINES

84 TTP TTP

85 NSFNET-IGP NSFNET-IGP

86 DGP 异类网关协议

87 TCF TCF

88 EIGRP EIGRP

89 OSPFIGP OSPFIGP

90 Sprite-RPC Sprite RPC 协议

91 LARP 轨迹地址解析协议

92 MTP 多播传输协议

93 AX.25 AX.25 帧

94 IPIP IP 中的 IP 封装协议

95 MICP 移动互联控制协议

96 SCC-SP 信号通讯安全协议

97 ETHERIP IP 中的以太网封装

98 ENCAP 封装标头

99 任意专用加密方案

100 GMTP GMTP

101 IFMP Ipsilon 流量管理协议

102 PNNI IP 上的 PNNI

103 PIM 独立于协议的多播

104 ARIS ARIS

105 SCPS SCPS

106 QNX QNX

107 A/N 活动网络

108 IPComp IP 负载压缩协议

109 SNP Sitara 网络协议

110 Compaq-Peer Compaq 对等协议

111 IPX-in-IP IP 中的 IPX

112 VRRP 虚拟路由器冗余协议

113 PGM PGM 可靠传输协议

114 任意 0 跳协议

115 L2TP 第二层隧道协议

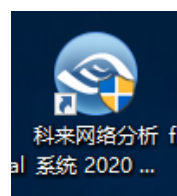
- 116 DDX D-II 数据交换 (DDX)
- 117 IATP 交互式代理传输协议
- 118 STP 计划传输协议
- 119 SRP SpectraLink 无线协议
- 120 UTI UTI
- 121 SMP 简单邮件协议
- 122 SM SM
- 123 PTP 性能透明协议
- 124 ISIS over IPv4
- 125 FIRE
- 126 CRTP Combat 无线传输协议
- 127 CRUDP Combat 无线用户数据报
- 128 SSCOPMCE
- 129 IPLT
- 130 SPS 安全数据包防护
- 131 PIPE IP 中的专用 IP 封装
- 132 SCTP 流控制传输协议
- 133 FC 光纤通道
- 134-254 未分配
- 255 保留

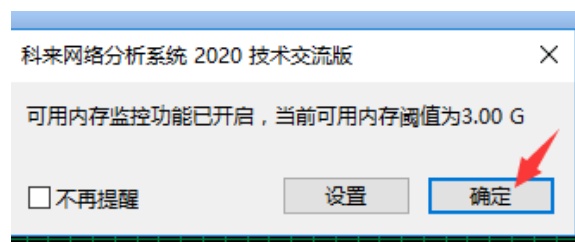
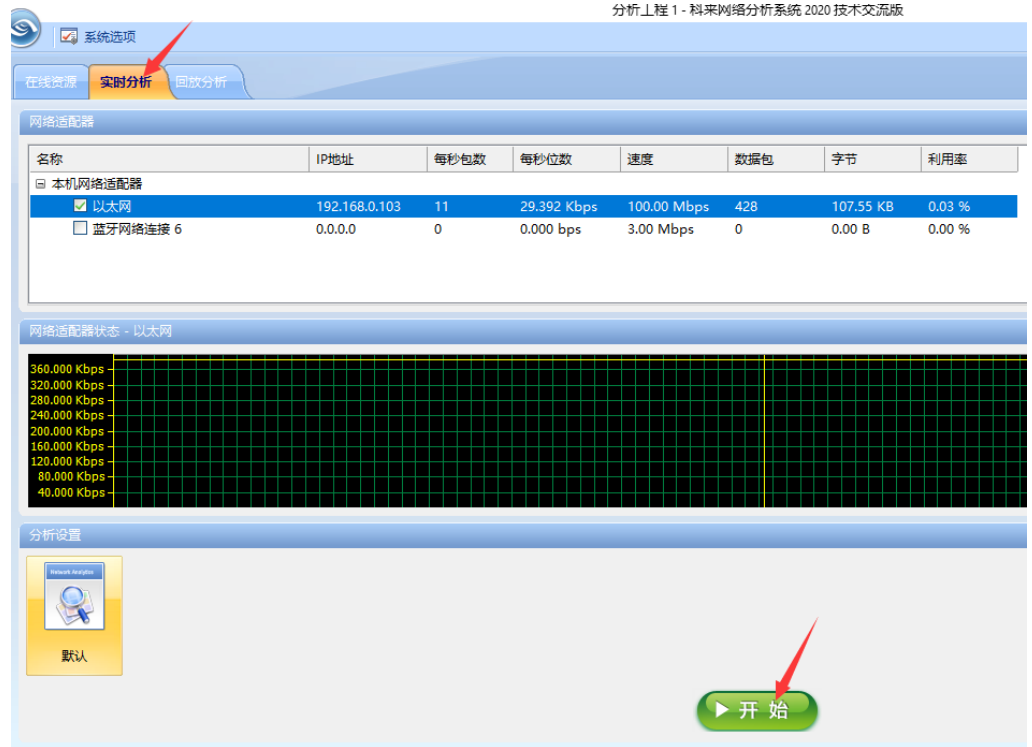
## 7.下载程序到单片机

正常情况会打印模块的MAC地址, IP地址等信息

```
ATK XCOM V2.0
CH395MAC 84:c2:e4:ec:ac:43
start
PHY_CONNECTED
IP:192.168.0.102
GWIP:192.168.0.1
Mask:255.255.255.0
DNS1:192.168.1.1
DNS2:192.168.0.1
```

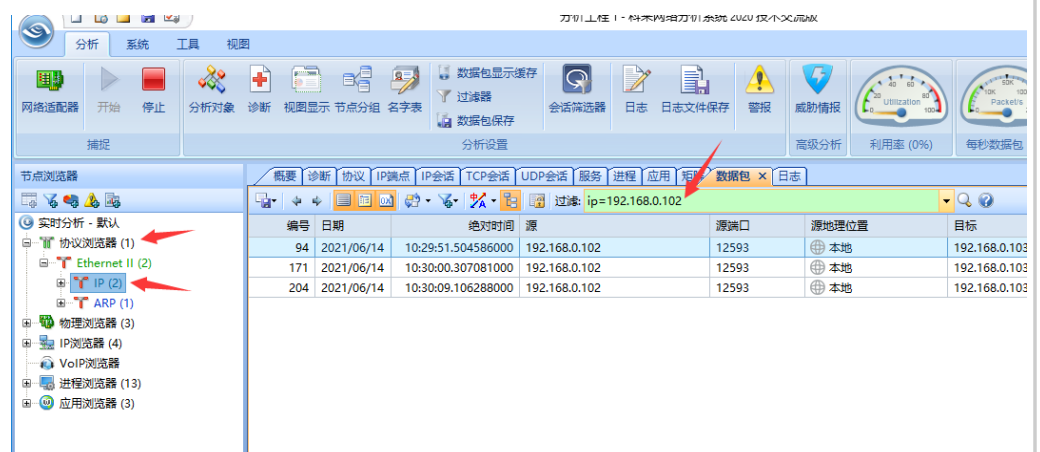
## 8.打开软件





## 9.按照下图操作

注:过滤填写的地址是咱网络模块的地址,根据自己的网络模块地址填写(输入法在英文状态下才可以输入)



## 10. 点击某条信息就可以查看

注:单片机每隔一段时间发送一条IP 数据给电脑.

The image shows a Wireshark packet capture. The top pane displays a list of 16 packets, all with source IP 192.168.0.102 and destination IP 192.168.0.103. Packet 94 is selected. The bottom pane shows the details of packet 94, which is an Ethernet II frame containing an Internet Protocol (IP) packet. The IP packet has a version of 4, a header length of 20, and a total length of 60. The destination address is F4:B5:20:09:8A:F9 and the source address is 84:C2:E4:EC:AC:43. The protocol is 8000. The packet bytes pane shows the raw data, with a red box highlighting the destination MAC address (F4:B5:20:09:8A:F9) and the source MAC address (84:C2:E4:EC:AC:43).

## 11. 提醒

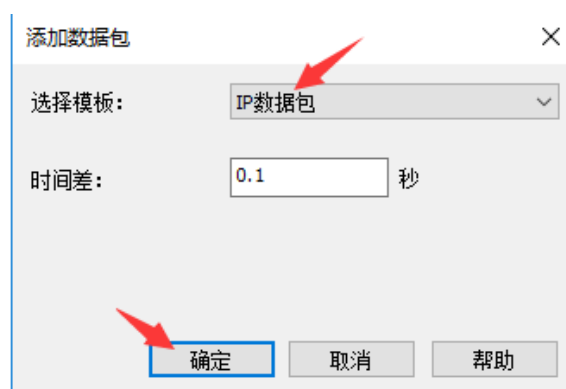
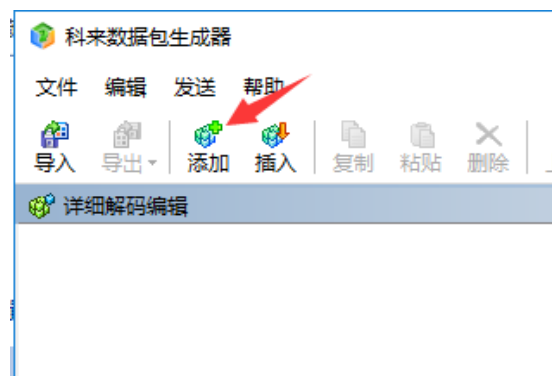
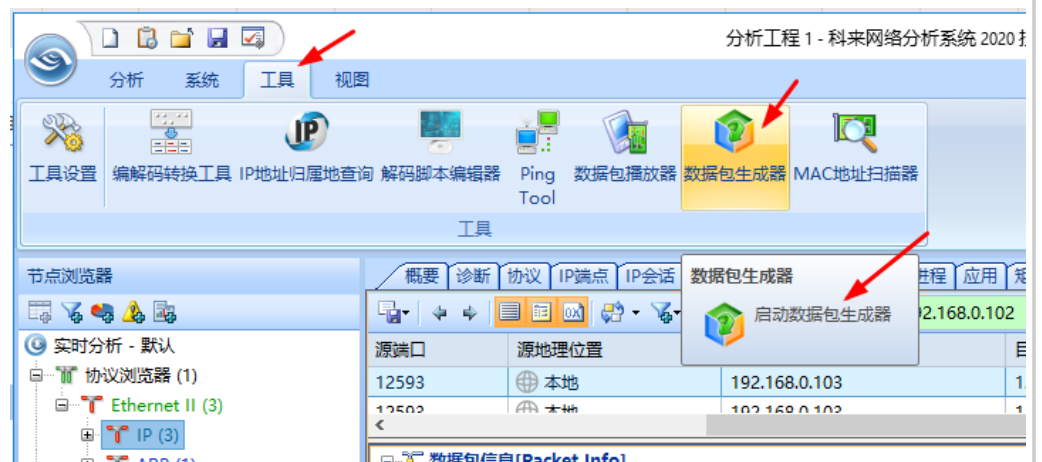
我使用单片机发送的并不是完整的IP 数据,以下红色标识的都是正确的数据

绿线的不是哈,并不是按照IP协议来的.

TCP传输控制协议那里我就是写了个 "111111" 然后发送了出来.....

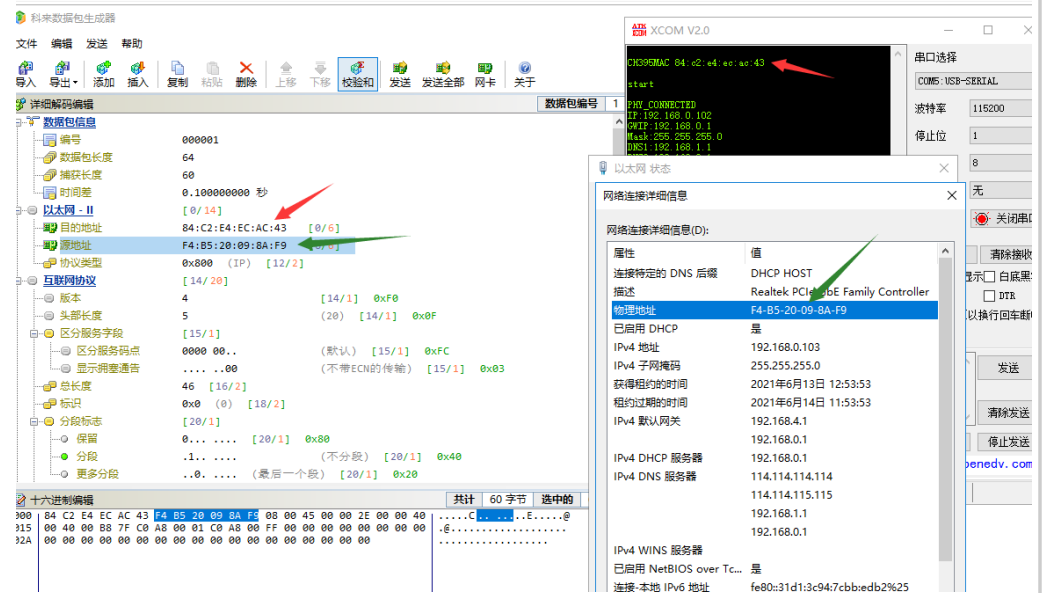
The image shows a Wireshark packet capture. The top pane displays a list of 16 packets, all with source IP 192.168.0.102 and destination IP 192.168.0.103. Packet 94 is selected. The bottom pane shows the details of packet 94, which is an Ethernet II frame containing an Internet Protocol (IP) packet. The IP packet has a version of 4, a header length of 20, and a total length of 60. The destination address is F4:B5:20:09:8A:F9 and the source address is 84:C2:E4:EC:AC:43. The protocol is 8000. The packet bytes pane shows the raw data, with a red box highlighting the destination MAC address (F4:B5:20:09:8A:F9) and the source MAC address (84:C2:E4:EC:AC:43). The packet details pane shows the Ethernet II frame, the Internet Protocol (IP) packet, and the Transmission Control Protocol (TCP) packet. The TCP packet has a source port of 12593 and a destination port of 12593. The sequence number is 8252948. The packet bytes pane shows the raw data, with a red box highlighting the destination MAC address (F4:B5:20:09:8A:F9) and the source MAC address (84:C2:E4:EC:AC:43). A green arrow points to the TCP packet details.

## 12.使用调试软件发送IP数据给模块

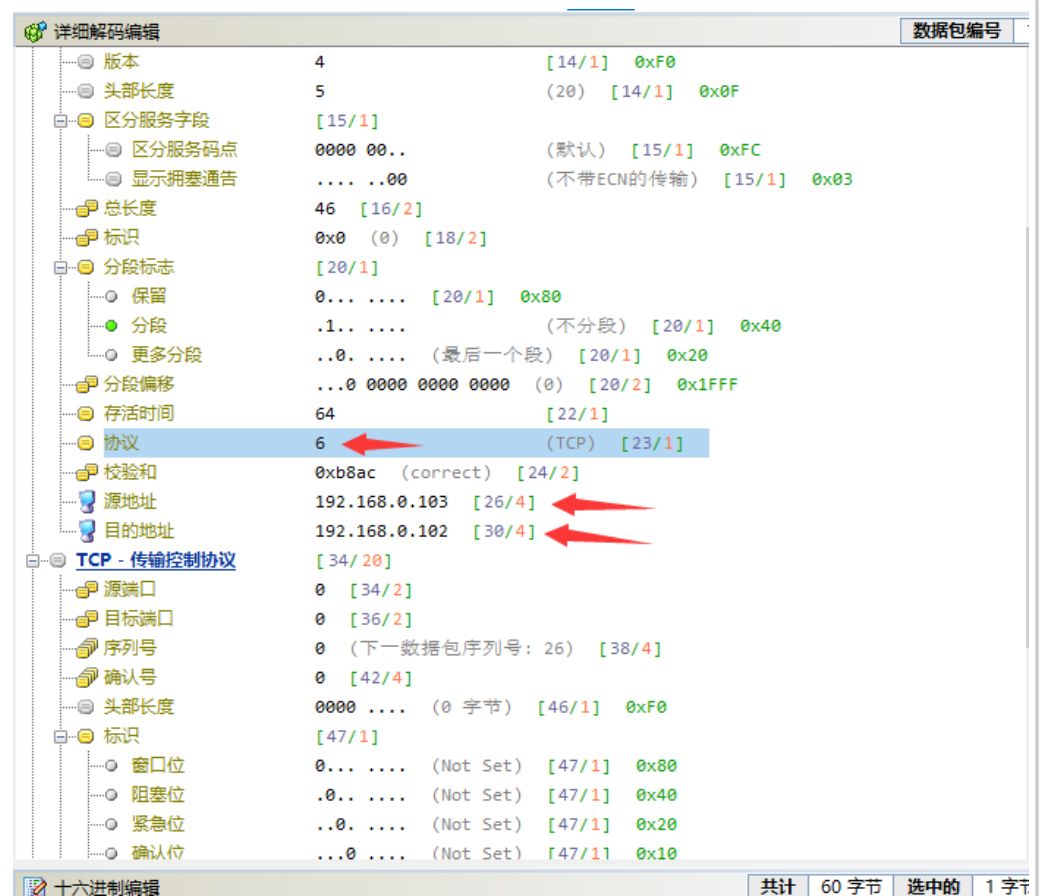


## 13.目的地址填写网络模块的MAC,原地址填写本电脑MAC

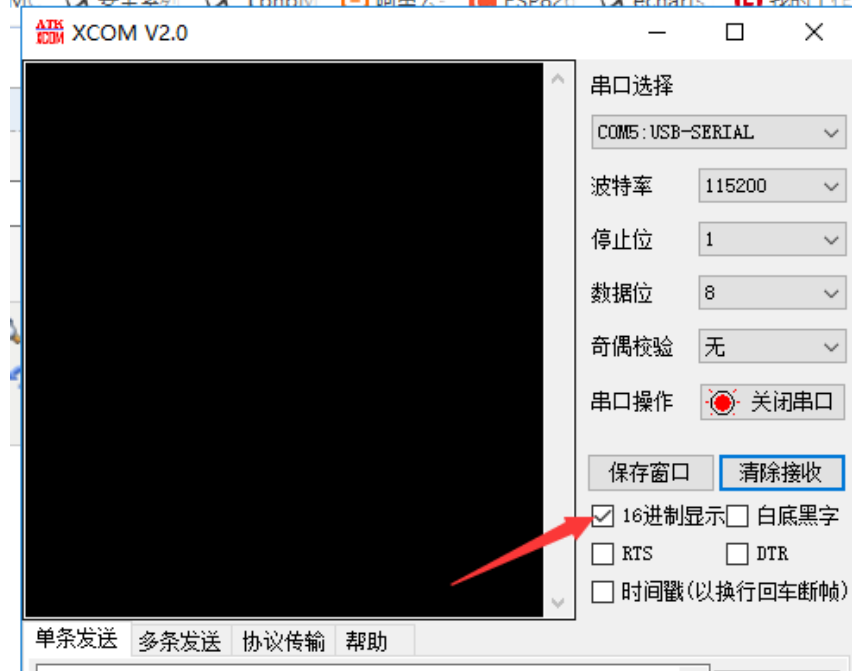




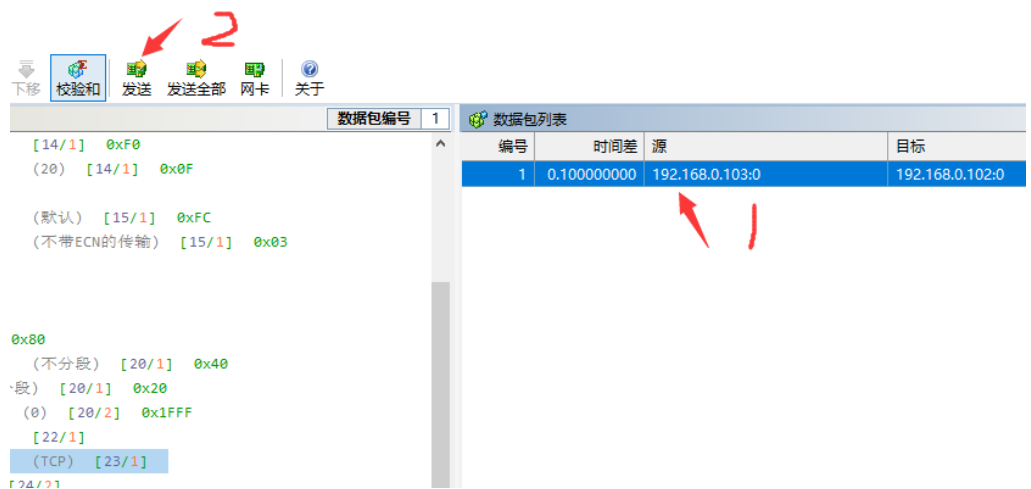
## 14.协议填写6(TCP),原地址填写本机IP,目的地址填写网络模块IP



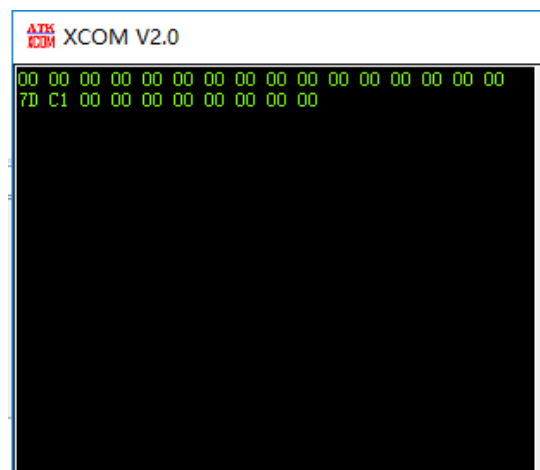
## 15.把网络调试助手调整为16进制显示



16,点击发送



17.如果有多个网卡,需要选择网卡,最后点击开始



18,当然这也不是完整的IP数据(TCP数据)

有经验的用户可以按照标准的数据设置完以后再发送.

详细解码编辑			
[-] 互联网协议	[ 14 / 20 ]		
[-] 版本	4	[ 14 / 1 ]	0xF0
[-] 头部长度	5	(20) [ 14 / 1 ]	0x0F
[-] 区分服务字段	[ 15 / 1 ]		
[-] 区分服务码点	0000 00..	(默认) [ 15 / 1 ]	0xFC
[-] 显示拥塞通告	.... ..00	(不带ECN的传输) [ 15 / 1 ]	0x03
[-] 总长度	46 [ 16 / 2 ]		
[-] 标识	0x0 (0) [ 18 / 2 ]		
[-] 分段标志	[ 20 / 1 ]		
[-] 保留	0... .... [ 20 / 1 ]	0x80	
[-] 分段	.1.. ....	(不分段) [ 20 / 1 ]	0x40
[-] 更多分段	..0. ....	(最后一个段) [ 20 / 1 ]	0x20
[-] 分段偏移	...0 0000 0000 0000 (0) [ 20 / 2 ]	0x1FFF	
[-] 存活时间	64 [ 22 / 1 ]		
[-] 协议	6 (TCP) [ 23 / 1 ]		
[-] 校验和	0xb8ac (correct) [ 24 / 2 ]		
[-] 源地址	192.168.0.103 [ 26 / 4 ]		
[-] 目的地址	192.168.0.102 [ 30 / 4 ]		
[-] TCP - 传输控制协议	[ 34 / 20 ]		
[-] 源端口	0 [ 34 / 2 ]		
[-] 目标端口	0 [ 36 / 2 ]		
[-] 序列号	0 (下一数据包序列号: 26) [ 38 / 4 ]		
[-] 确认号	0 [ 42 / 4 ]		
[-] 头部长度	0000 .... (0 字节) [ 46 / 1 ]	0xF0	
[-] 标识	[ 47 / 1 ]		
[-] 窗口位	0... .... (Not Set) [ 47 / 1 ]	0x80	
[-] 阻塞位	.0.. .... (Not Set) [ 47 / 1 ]	0x40	
[-] 紧急位	..0. .... (Not Set) [ 47 / 1 ]	0x20	
[-] 确认位	...0 .... (Not Set) [ 47 / 1 ]	0x10	
[-] 急迫位	.... 0... (Not Set) [ 47 / 1 ]	0x08	
[-] 重置位	.... .0.. (Not Set) [ 47 / 1 ]	0x04	
[-] 同步位	.... ..0. (Not Set) [ 47 / 1 ]	0x02	
[-] 终止位	.... ...0 (Not Set) [ 47 / 1 ]	0x01	
[-] 窗口	0 [ 48 / 2 ]		
[-] 校验和	0... .. (正确) [ 50 / 2 ]		

## 程序说明

### 1.初始化IP RAW

```
timer.c  usart.c  main.c  CH395CMD.H  CH395INC.H  CH395CMD.C  timer.h
147 IWDG_Init(IWDG_Prescaler_256,156*10);
148
149 /*获取芯片版本*/
150 while((ch395_version = CH395CMDGetVer()) < 0x40)
151 {
152     printf("CH395CMDGetVer ERR\r\n");
153     delay_ms(100);
154 }
155
156 /*测试命令, 按位取反返回说明测试通过*/
157 while(CH395CMDCheckExist(0x55) != 0xaa)
158 {
159     printf("\r\nCH395CMDCheck ERR\r\n");
160     delay_ms(100);
161 }
162
163 /*初始化模块:成功返回 0 */
164 while(CH395CMDInitCH395() != 0)
165 {
166     printf("\r\nCH395CMDInitCH395 ERR\r\n");
167     delay_ms(100);
168 }
169
170 CH395CMDGetMACAddr(buf);
171 printf("\r\nCH395MAC %02x:%02x:%02x:%02x:%02x:%02x\r\n",buf[0],buf[1],buf[2],buf[3],buf[4],buf[5]);
172
173 /*初始化IP RAW*/
174 ch395_socket_tcp_client_init(SocketIndex,SocketDesIP);
175
176 printf("\r\nstart\r\n");
177 while(1)
178 {
179     IWDG_Feed();//喂狗
180
181     /*每隔一段时间发送个数据*/
182     //填写的数据为IPRAW数据,应该根据协议包含端口号等等,但是我只是随意写了111111
183     if(Timer2Cnt>8000)
184     {
185         Timer2Cnt = 0;
186         CH395SendData(SocketIndex,"111111",6);
187     }
188 }
```

```
timer.c  usart.c  main.c  CH395CMD.H  CH395INC.H  CH395CMD.C  timer.h
36 #define recv_buff_len 1500
37 unsigned char recv_buff[recv_buff_len];
38
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(0,1,2,3,4,5,6,7) */
46 UINT8 SocketDesIP[4] = {192,168,0,103}; /* Socket 目的IP地址 */
47 const UINT8 IPRawProto = 0x06; /* IP包协议类型 */
48
49 /**
50  * @brief 初始化socket
51  * @param sockindex Socket索引(0,1,2,3,4,5,6,7)
52  * @param ipaddr 目的地址
53  * @param desprot 目的端口号
54  * @param surprot 本地端口号
55  * @retval 0:初始化成功; others:初始化失败
56  * @warning None
57  * @example
58  */
59 char ch395_socket_tcp_client_init(UINT8 sockindex,UINT8 *ipaddr)
60 {
61     CH395SetSocketDesIP(sockindex,ipaddr); /* 目的地址 */
62     CH395SetSocketProtType(sockindex,PROTO_TYPE_IP_RAW); /* 协议类型 */
63     CH395SetSocketIPRAWProto(sockindex,IPRawProto); /* 设置协议字段 */
64     if(CH395OpenSocket(sockindex) !=0) /* 打开Socket */
65     {
66         return 1;
67     }
68     return 0;
69 }
70
```

2.因为是局域网通信,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.每隔一段时间发送个数据出去

```
timer.c  usart.c  main.c  CH395CMD.H  CH395INC.H  CH395CMD.C  timer.h
168 }
169
170 CH395CMDGetMACAddr(buf);
171 printf("\r\nCH395MAC %02x:%02x:%02x:%02x:%02x:%02x\r\n",buf[0],buf[1],buf[2],buf[3],buf[4],buf[5]);
172
173 /*初始化IP RAW*/
174 ch395_socket_tcp_client_init(SocketIndex,SocketDesIP);
175
176 printf("\r\nstart\r\n");
177 while(1)
178 {
179     IWDG_Feed();//喂狗
180
181     /*每隔一段时间发送个数据*/
182     //填写的数据为IPRAW数据,应该根据协议包含端口号等等,但是我只是随意写了111111
183     if(Timer2Cnt>8000)
184     {
185         Timer2Cnt = 0;
186         CH395SendData(SocketIndex,"111111",6);
187     }
188
189     //INT引脚产生低电平中断以后进去判断
190     if(Query395Interrupt())
```

### 4.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_SOCK0){
242     ch395_socket_tcp_client_interrupt(SocketIndex);
243 }
244 /* 处理 SOCK1 中断 */
245 if(ch395_status & GINT_STAT_SOCK1){
246
247 }

```

```

timer.c  usart.c  main.c  CH395CMD.H  CH395INC.H  CH395CMD.C  timer.h
72 /**
73  * @brief    socket处理函数(把此函数放到全局socket中断里面)
74  * @param    sockindex    Socket索引(0,1,2,3,4,5,6,7)
75  * @param    None
76  * @param    None
77  * @param    None
78  * @retval    None
79  * @warning   None
80  * @example
81  */
82 void ch395_socket_tcp_client_interrupt(UINT8 sockindex)
83 {
84     UINT8  sock_int_socket;
85     UINT16 len;
86
87     /* 获取socket 的中断状态 */
88     sock_int_socket = CH395GetSocketInt(sockindex);
89
90     /* 发送缓冲区空闲, 可以继续写入要发送的数据 */
91     if(sock_int_socket & SINT_STAT_SENBUF_FREE)
92     {
93     }
94
95     /* 发送完成中断 */
96     if(sock_int_socket & SINT_STAT_SEND_OK)
97     {
98     }
99
100    /* 接收数据中断 */
101    if(sock_int_socket & SINT_STAT_RECV)
102    {
103        len = CH395GetRecvLength(sockindex);/* 获取当前缓冲区内数据长度 */
104        if(len == 0)return;
105        if(len > recv_buff_len)len = recv_buff_len;
106        CH395GetRecvData(sockindex,len,recv_buff);/* 读取数据 */
107
108        /*把接收的数据发送给服务器*/
109        CH395SendData(sockindex,recv_buff,len);
110
111        /*使用串口打印接收的数据*/
112        PutData(&rb_t_usart1_send,recv_buff,len);
113        USART_ITConfig(USART1, USART_IT_TXE, ENABLE);

```

```

114 }
115
116 /* 连接中断, 仅在TCP模式下有效*/
117 if(sock_int_socket & SINT_STAT_CONNECT)
118 {
119     printf("SINT_STAT_CONNECT\n");
120 }
121
122 /* 断开中断, 仅在TCP模式下有效 */
123 if(sock_int_socket & SINT_STAT_DISCONNECT)
124 {
125     printf("SINT_STAT_DISCONNECT \n");
126 }
127
128 /* 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*/
129 if(sock_int_socket & SINT_STAT_TIM_OUT)
130 { /*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*/
131     printf("SINT_STAT_TIM_OUT\n");
132 }
133 }

```

## 注意事项

### 关于协议字段设置的注意事项

CH395 处理 IPRAW 的优先级高于 UDP 和 TCP, 如果 IP 协议字段设置为 17 (UDP) 或者 6 (TCP), 则可能存在和其他 socket 冲突的可能性, 在使用时应当注意避免, 下面列举两种情况进行说明:

① Socket0 设置为 IPRAW 模式, IP 协议字段为 17, Socket1 为 UDP 模式。在 UDP 模式下, IP 包的协议字段也是 17, 这样就会导致 Socket1 通讯的数据会被 Socket0 拦截, 无法接收到数据。

② Socket0 设置为 IPRAW 模式, IP 协议字段为 6, Socket1 为 TCP 模式。在 TCP 模式下, IP 包的协议字段也是 6, 这样就会导致 Socket1 通讯的数据会被 Socket0 拦截, 无法接收到数据。

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: [11-网络芯片CH395Q学习开发-模块使用Socket0作为UDP组播\(多播\)通信,MAC地址过滤](#)

posted on 2021-06-14 11:11 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

[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：

- 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”
  - 传日本最早将在6月启动对苹果和谷歌反垄断调查
- » 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。