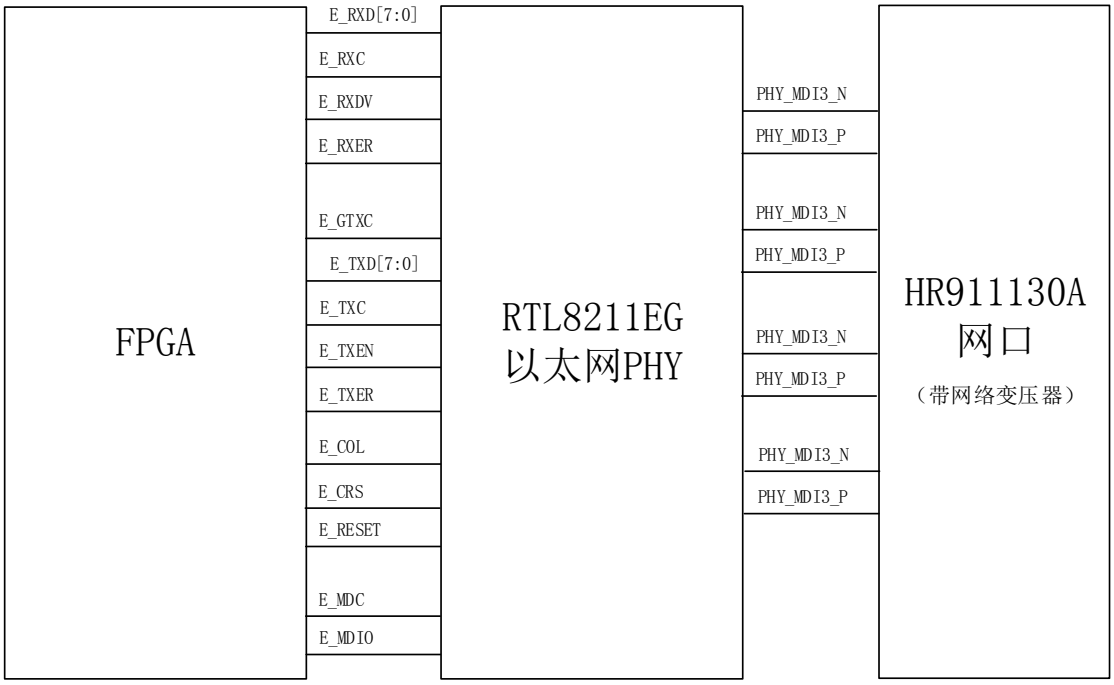
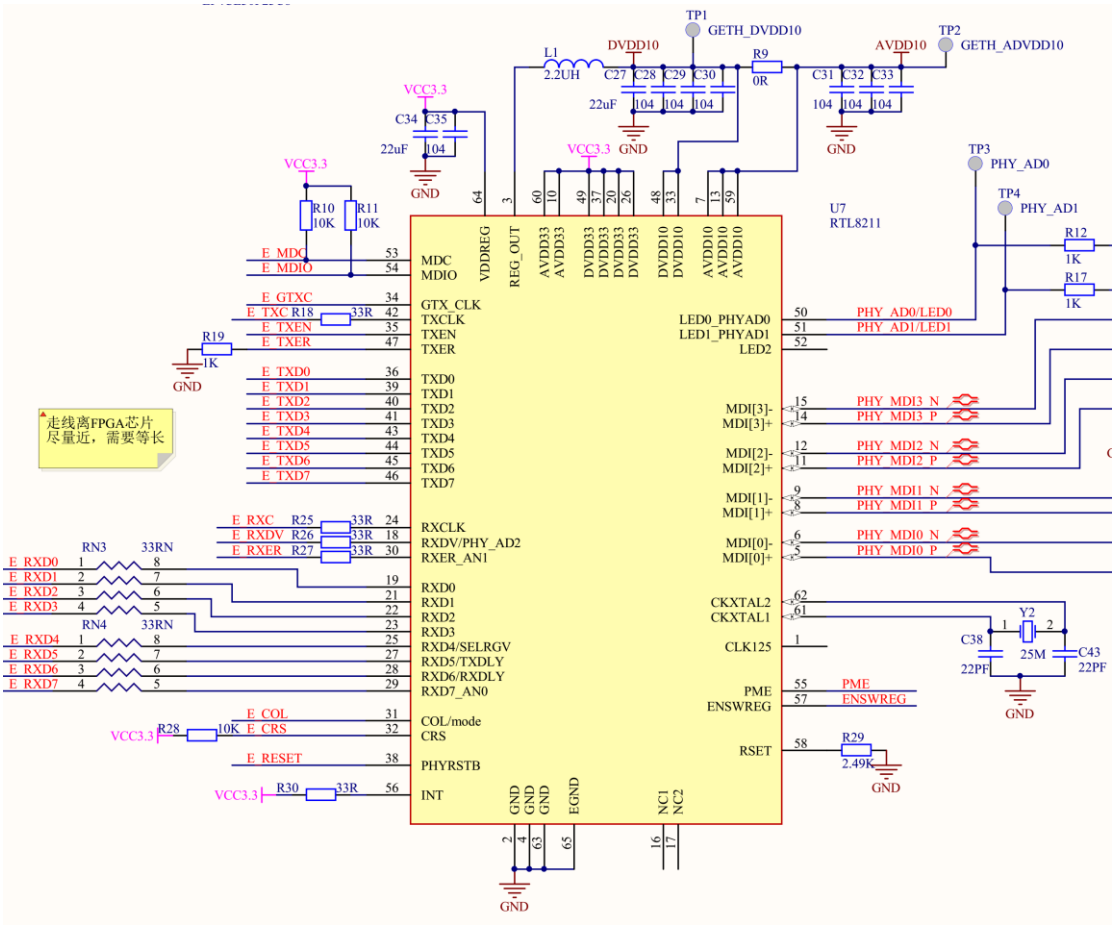
ACM8211 千兆网模块使用说明书

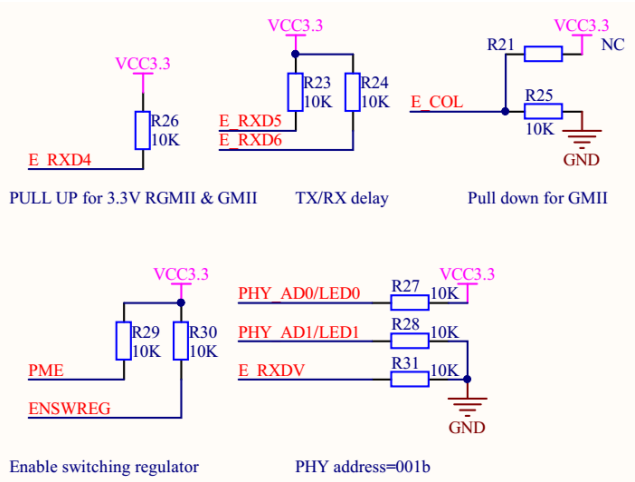
以太网接口作为一种互联型接口，当下应用非常的广泛。从家用宽带，到服务器数据交换，再到工业控制，各个地方都能看到以太网的身影。FPGA 系统使用以太网，则主要用于高速远距离的数据传输，如 LED 大屏显示、监控系统等。以太网数据链路，也由早期的电话线传输，到后来的专用双绞线，发展到光纤，高频无线电，以及现在比较新颖的 LIFI。

ACM8211 模块给 FPGA 开发板提供千兆以太网接口，该接口由千兆以太网 PHY 和网络变压器接口组成。当需要发送以太网数据时，FPGA 把数据发送给 PHY 芯片，PHY 芯片将数据编码后，通过网络变压器将数据加载到网线上。数据经由网络传递到接收方。远端发送过来的数据，经由网线传递给网络变压器，网络变压器的输出连接到 PHY 芯片上，PHY 芯片对信号进行解码后，得到实际的数据，然后将数据传递给 FPGA 芯片。FPGA 实现千兆以太网数据传输的功能框图如下所示：





RTL8211 是一款支持 GMII、RGMII、MII 接口的以太网物理层收发器，能够工作在 100M Base 或 1000M Base 模式。接口可设置为 GMII、RGMII、MII 接口。并提供了若干引脚用于配置工作模式。



ACM8211 模块默认使用 GMII 接口，实现该配置的方法是将对应的配置引脚 E_COL（对，就是这个引脚，芯片在上电时会检测该引脚的状态，并根据该引脚的状态确定工作模式，正常工作时又是正常的 COL 功能）通过 10K 的电阻接到 GND。

针对工程师用户，如果确实希望在使用 RGMII 接口，可以将 R21 焊接为 10K 电阻，将

R25 断开。

以太网物理层芯片都有一个器件地址，该地址可通过外部引脚设置，上图中，R20、R21、R24 通过连接到 VCC 或者 GND，决定了芯片的器件地址为 001b。当然，这些状态也仅在芯片上电时刻被读取，当芯片正常工作后，这些引脚就又恢复了普通功能。

AC6102 开发板千兆以太网 UDP 传输实验

ACM8211 模块设计实现了一路 GMII 接口的千兆以太网电路，通过该以太网电路，用户可以将 FPGA 采集或运算得到的数据传递给其他设备如 PC 或服务器，或者接收其他设备传输过来的数据并进行处理。

接触过以太网的用户，应该最常听说的是 TCP/IP 协议，确实，在 PC 端或者嵌入式系统中，TCP/IP 协议应用非常广泛，因此，当大家看到 FPGA 上带有以太网接口时，可能第一个想到的也是实现 TC/IP 协议。这里，首先可以很肯定的告诉大家，使用 FPGA 实现 TCP/IP 协议是完全没有问题的，但是，实现的方式却不是大家最期望的直接使用 Verilog 编写协议层代码来实现。FPGA 发展到现在，三十多年了，却鲜见有成功商用的 RTL 级的 TCP/IP 的设计，而大部分使用 Verilog 或者 VHDL 实现的以太网传输，都是基于非常简单的 UDP 协议的。当然，探索或者实现其中部分功能的人还是有的，只是，很难做到像 PC 那样灵活应用。

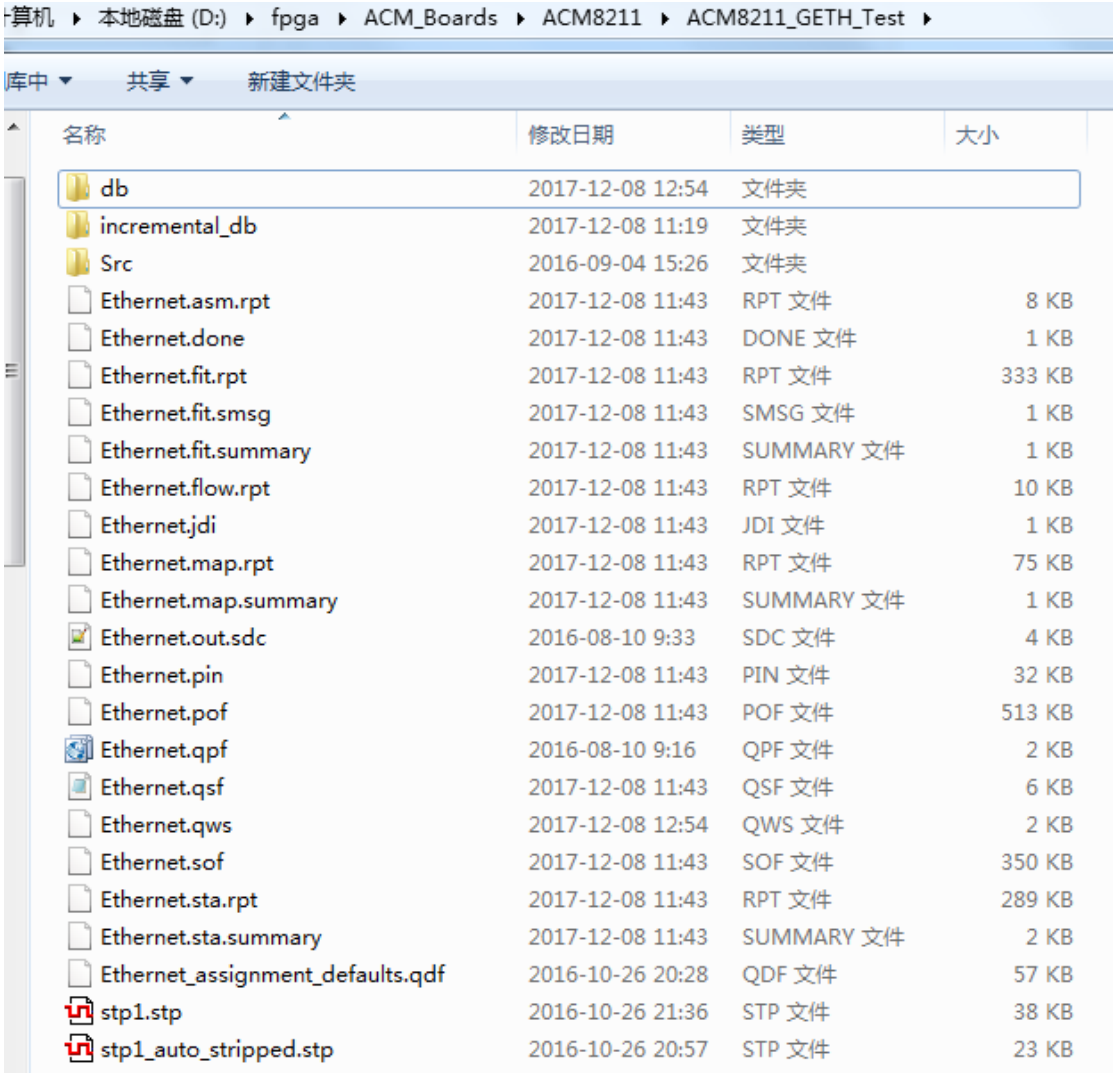
个人理解，TCP/IP 协议设计之初就是根据软件灵活性设计的，因此在很多设计考虑上，并不适合使用硬线逻辑实现。TCP/IP 协议非常的复杂，如果使用硬件逻辑实现，工程量必然十分浩大，而且功能和性能都无法得到保证。

那么怎样在 FPGA 上实现 TCP/IP 协议呢？答案就是 SOPC 技术。即使用嵌入式软核技术，

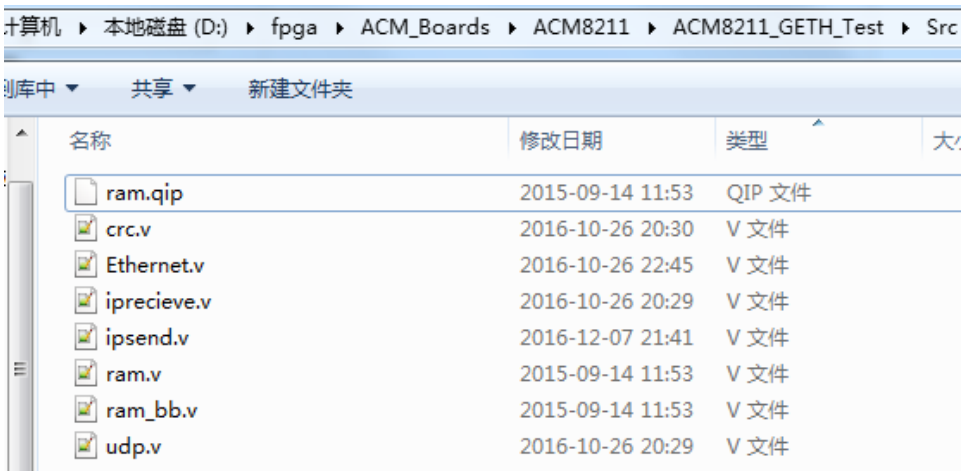
在 FPGA 上搭建软核 CPU 系统，再通过 CPU 来运行软件 TCP/IP 协议，从而实现相应功能。但是这种实现方式对于很多用户来说，前期系统创建的过程比较繁琐，因此很多朋友都难以上手，因此这种方式使用的也并不是很广泛。

上面说过，在 FPGA 上，可以使用 Verilog 实现 UDP 协议来进行数据的传输。UDP 协议是一种不可靠传输，发送方只负责将数据发送出去，而不管接收方是否正确的接收。非常类似于 UART 串口传输。但是，在很多场合，是可以接受这种潜在的不可靠性的，例如视频实时传输显示。在这类系统中，由于数据并不需要进行运算并得到非常精确的结果用于其他功能，而仅仅是显示在屏幕上，因此可以接受一定程度的丢包或者误码。此类应用在 LED 大屏显示系统中应用非常广泛。本节就介绍提供给大家的一个基于 UDP 传输例程的使用方法。

1、本例提供的 UDP 传输例程工程压缩包名为 ACM8211_GETH_Test.rar，该文件可在我们提供的配套资料中找到。该例程是基于小梅哥 AC620 FPGA 开发板开发的，其他板卡需要修改引脚信息才能工作。
2、解压 ACM8211_GETH_Test.rar 到不含中文或者空格的目录中，如 D:\fpga。解压后工程目录下内容如下所示：



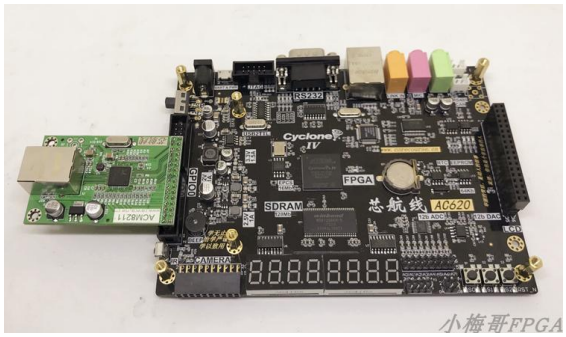
其中 Src 文件夹下存放的为程序源码，如下图所示：



这里对其中几个重要文件（夹）简单说明下功能：

文件或文件夹名	文件或文件夹功能描述
Ethernet.qpf	工程文件，使用 Quartus II 13.0 打开。
Ethernet.qsf	工程配置文件，记录了工程相关的各个设置，包括引脚分配。
Ethernet.sof	工程编译后得到的配置文件，用以下载到 FPGA 中运行
stp1.stp	signaltap ii 设置文件，测试时双击运行此文件以查看 MII 接口上发送和接受的数据内容，通过抓取的波形内容分析收发是否正确。
Ethernet.v	测试工程顶层，例化了以太网 UDP 协议收发代码和测试代码，组成完整的测试工程。
udp.v	UDP 协议顶层，例化了 UDP 协议接受和发送代码，以及 CRC32 校验电路，以实现对发送的数据追加 32 位校验码的功能。
ipsend.v	UDP 协议发送模块，实现将发送缓存（RAM）中的数据组建成 UDP 协议包并发送出去的功能
iprecieve.v	UDP 协议接收模块，实现将以太网口接收到的数据解包、得到最后的用户数据部分并写入到接收缓存（RAM）中。
crc.v	CRC32 校验逻辑，实现对数据的 CRC32 校验功能
ram.qip、ram.v	发送和接受缓存，本例中，发送和接受使用同一个缓存，即网口接受到的数据会被直接写入到发送缓存中，所以，每当网口接受到数据后，就会更新发送缓存中的内容，因此网口发送的内容也会变成新接受到的那一帧内容。

3、将 ACM8211 模块连接到芯航线 FPGA 开发板或者友晶 DE 系列开发板上。注意，由于模块为 32 针接口，而支持的开发板的通用接口为 40 针，因此存在对齐的问题，连接时候，模块的 1 脚（方形焊盘）和开发板上 40 针扩展接口的 1 脚对齐即可。下图分别为模块与 DE2-115、AC620 开发板连接的示意图，其他开发板请类比连接。



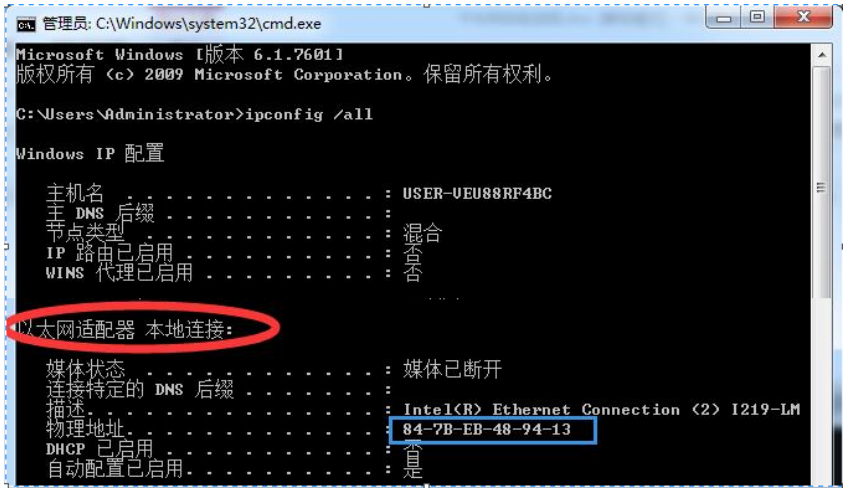
4、确认自己 PC 的网卡是否是千兆网卡，用户可以点击本地连接查看。如果不是千兆网卡，请更换带千兆网卡的电脑或加装千兆网卡。再用五类+或者六类网线连接开发板的网口和 PC 的网口。给开发板供电，打开电源开关，插上 USB Blaster 下载器。

5、双击 Ethernet.qpf 以打开工程（强烈建议使用工程创建时候对应的版本即 Quartus II 13.0，使用其他版本打开或编译遇到问题，请邮件告知我们，以获得解决方案邮箱：xiaomeige_fpga@foxmail.com）

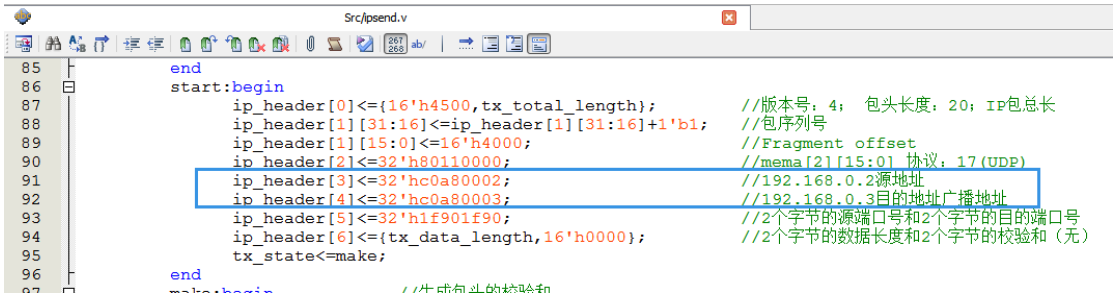
6、修改 UDP 发送模块(ipsend.v)中的目标 mac address 为你使用的网卡的 mac address，修改后重新编译一边。

```
//此段为调试主机（例如用户的PC）的MAC地址，在调试时候，需要根据自己的电脑MAC地址值修改本端内容
 mac_addr[0]<=8'h84;                    //目的MAC地址 84-7B-EB-48-94-13
 mac_addr[1]<=8'h7b;
 mac_addr[2]<=8'heb;
 mac_addr[3]<=8'h48;
 mac_addr[4]<=8'h94;
 mac_addr[5]<=8'h13;
```

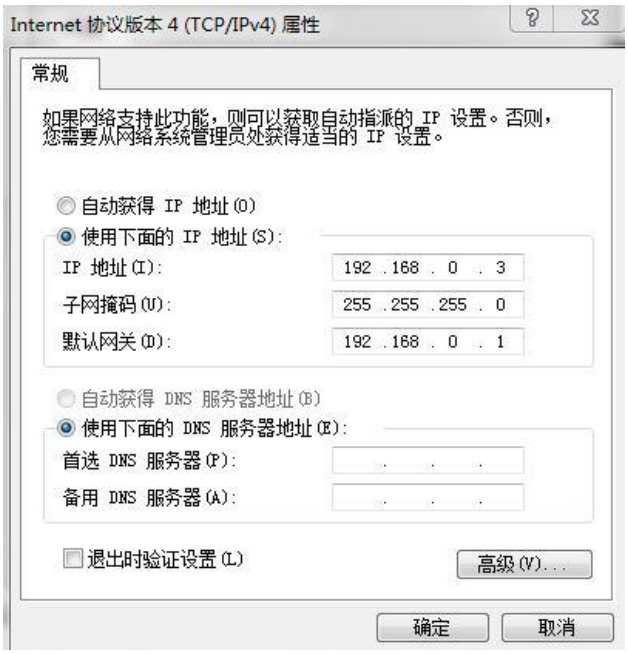
如果不知道自己 PC 网卡的 mac address，就在 DOS 命令窗口，用 ipconfig -all 命令看一下。



7、修改 PC 的 IP 地址为 192.168.0.3。PC 的 IP Address 需要和发送模块(ipsend.v)中设置一致，不然 PC 端会接收不到开发板发送的 UDP 数据包。当然，用户也可以修改代码中的目标 IP 地址，如下所示：



代码中本机（FPGA）和目标机（PC）IP 地址



修改 PC 的 IP 地址

8、在 DOS 命令窗口绑定开发板的 IP 地址和 MAC 地址，（由于本测试工程不支持 ARP 协议，因此只能通过这种 IP 和 MAC 绑定的方式来强制将开发板的 IP 地址和 MAC 地址关联在一起，这样，当 PC 发送给 192.168.0.2 的数据包的时候，目标 MAC 地址自动为开发板的 MAC 地址。）

运行命令：ARP -s 192.168.0.2 00-0a-35-01-fe-c0
绑定后我们可以用 arp -a 命令来查看 PC 上绑定的结果。

```
C:\Users\Administrator>arp -s 192.168.0.2 00-0a-35-01-fe-c0

C:\Users\Administrator>arp -a

接口: 192.168.0.3 --- 0xb
  Internet 地址         物理地址              类型
  192.168.0.2           00-0a-35-01-fe-c0     静态
  192.168.0.255         ff-ff-ff-ff-ff-ff     静态
  224.0.0.2             01-00-5e-00-00-02     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  224.0.0.251           01-00-5e-00-00-fb     静态
  224.0.0.252           01-00-5e-00-00-fc     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态
  255.255.255.255       ff-ff-ff-ff-ff-ff     静态
```

如果运行 ARP 出现加载失败，换另一种方法绑定

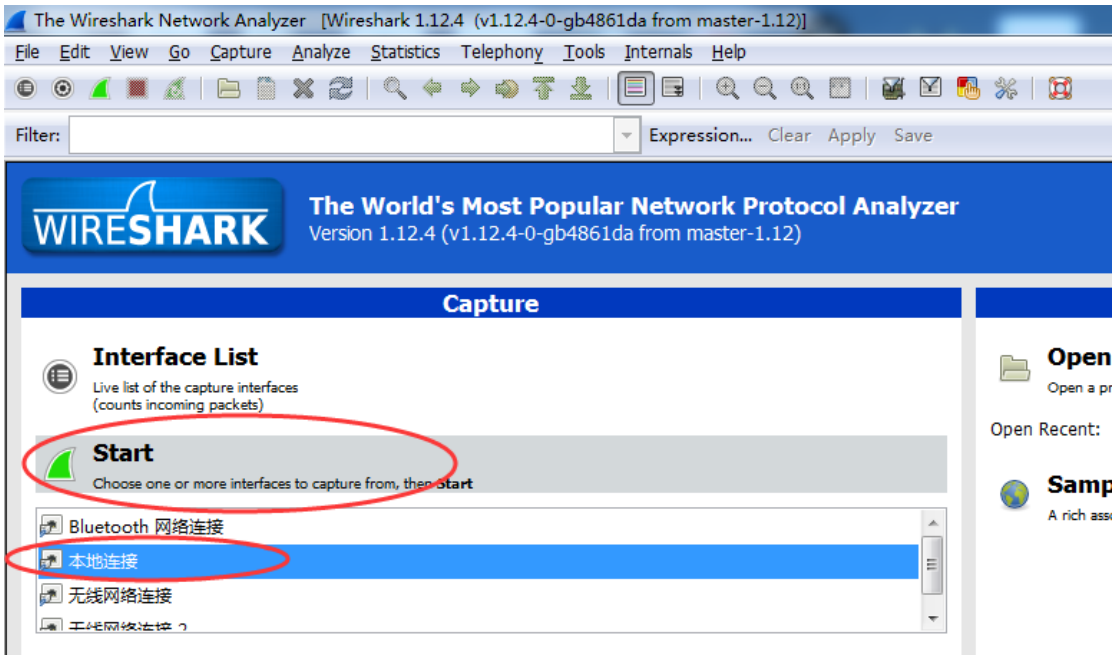
1) 使用 netsh i i show in 命令查看本地连接的 idx 编号，如“11”
2) 使用 netsh -c “i i” add neighbors 11(idx 编号) “192.168.0.2” “00-0a-35-01-fe-c0”
3) 使用 arp -a 命令来查看 PC 上帮定的结果

```
C:\Users\Administrator>arp -s 192.168.0.2 00-0a-35-01-fe-c0
ARP 项添加失败: 拒绝访问。


C:\Users\Administrator>netsh i i show in

Idx     Met         MTU          状态                名称
---  ----------  ----------  ------------  ---------------------------
  1          50  4294967295  connected     Loopback Pseudo-Interface 1
 11           5        1500  disconnected  本地连接


C:\Users\Administrator>netsh -c "i i" add neighbors 11 "192.168.0.2" "00-0a-35-0
1-fe-c0"
```

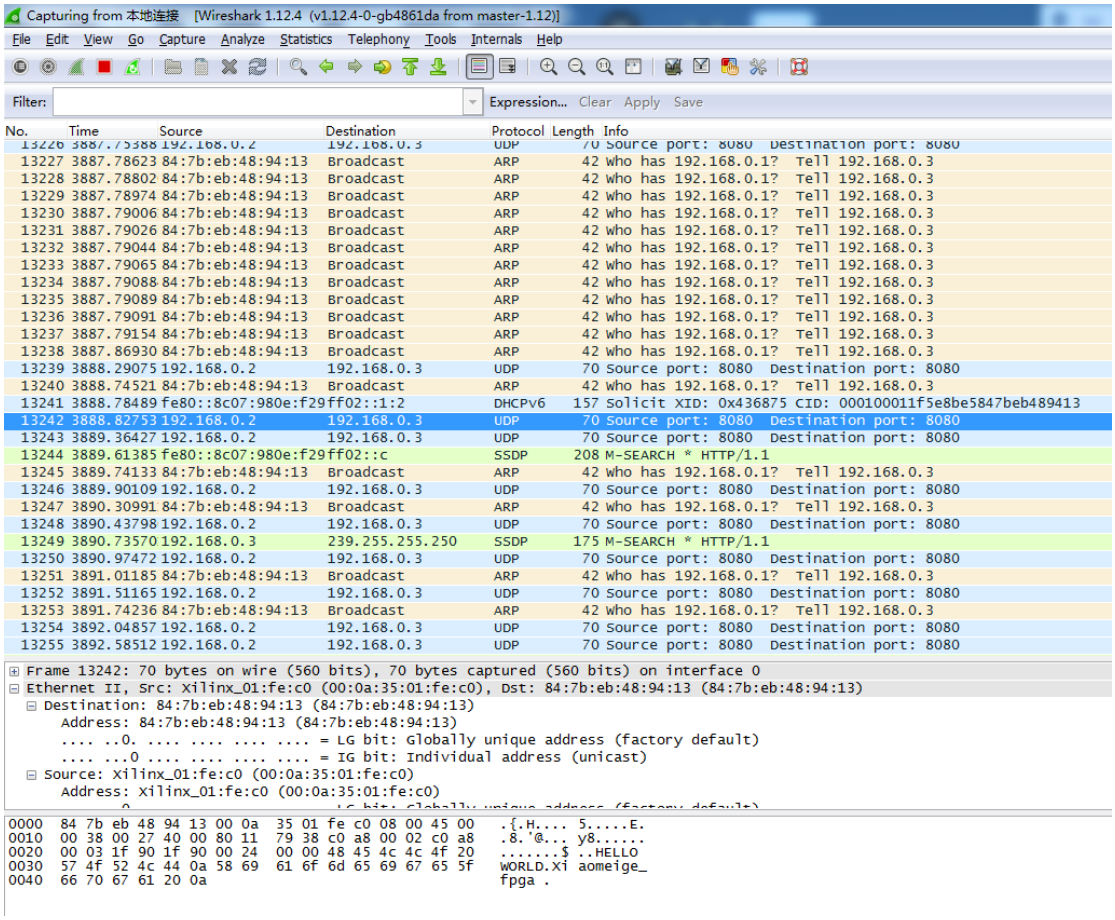
9、安装网络抓包工具 Wireshark，我们在实验的时候可以用这工具来查看 PC 网口发送的数据和接收到的数据。

10、打开 Quartus II 的 Programer，选择下载器和需要下载的文件，然后点击下载以开始下载 Ethernet.sof 文件到开发板中。

11、打开安装好的 wireshark 抓包工具。在软件界面选择您 PC 的千兆网卡，按开始按钮开始抓包。



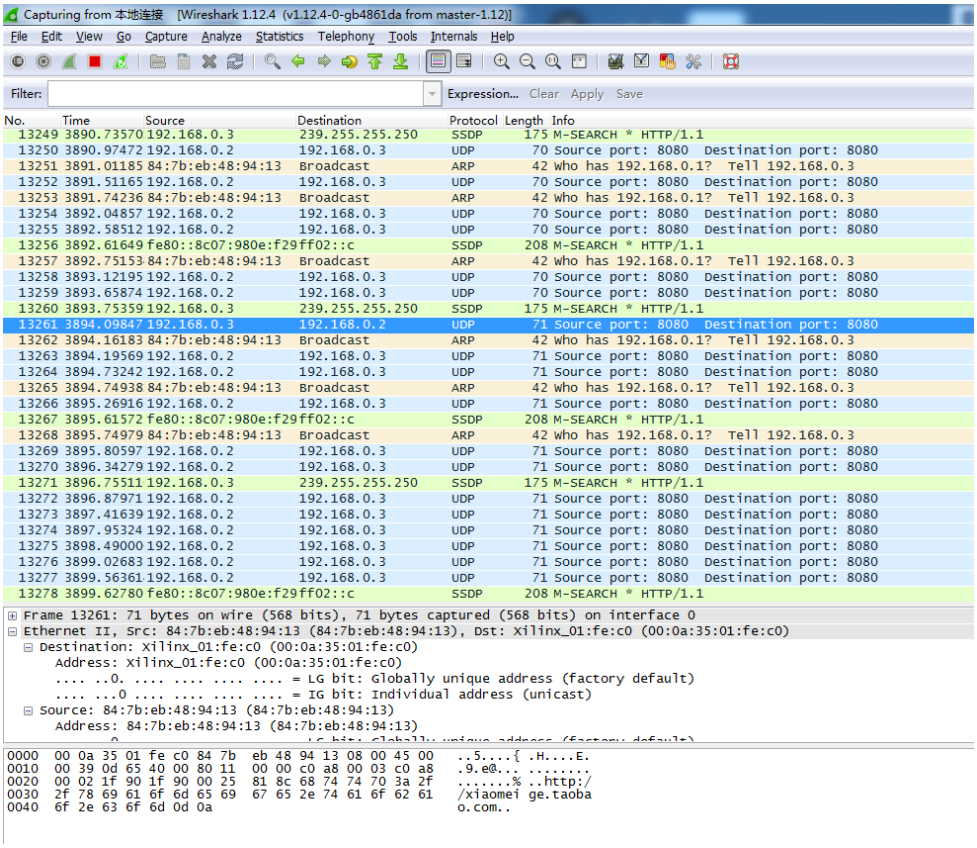
在 wireshark 抓包窗口我们可以看到开发板(192.168.0.2)向 PC 网口(192.168.0.3)发来的数据包。



12、打开网络调试助手并按照如图所示设置各项参数，再按连接按钮(这里的本地的 IP 地址为 PC 的 IP Address，本地端口需要跟 FPGA 程序中的一致，为 8080)

点就连接后，目标主机(192. 168. 0. 2)和目标端口(8080)都是默认值。

13、再在网络调试助手的发送窗口发送一大串字符，在网络的数据接收窗口我们可以看到从 FPGA 返回的数据也变成刚发送字符串。



注意：以太网的数据帧的传输有包长的要求，一般在 46—1500 字节。所以在发送以太网数据包的时候，数据帧的长度不能太短，不然会导致 PC 数据包发送而 FPGA 收不到数据包的情况。

小梅哥
2017 年 12 月 15 日