以太网（ENC28J60）实验

作者	fire
E-Mail	firestm32@foxmail.com
QQ	313303034
博客	firestm32.blog.chinaunix.net
硬件平台	野火 STM32 开发板
库版本	ST3.0.0

实验描述：在浏览器上创建一个 web 服务器，通过 web 里面的命令来控制开发板上的 LED 的亮灭。

应用->

1:在 PC 机的 DOS 界面输入： ping 192.168.1.15 ，看能否 ping 通。

2:在 IE 浏览器中输入：http://192.168.1.15/123456 则会出现一个网页，通过网页中的命令可以控制开发板中的 LED 的亮灭。

硬件连接：PB13 ：ENC28J60-INT

PA6-SPI1-MISO ：ENC28J60-SO

PA7-SPI1-MOSI ：ENC28J60-SI

PA5-SPI1-SCK ：ENC28J60-SCK

PA4-SPI1-NSS ：ENC28J60-CS

PE1 ：ENC28J60-RST

库文件 ：startup/start_stm32f10x_hd.c

CMSIS/core_cm3.c

CMSIS/system_stm32f10x.c

FWlib/stm32f10x_gpio.c

FWlib/stm32f10x_rcc.c

FWlib/stm32f10x_usart.c

FWlib/stm32f10x_spi.c

用户文件：USER/main.c

USER/stm32f10x_it.c

USER/led.c

USER/usart.c

USER/spi_enc28j60.c

USER/enc28j60.c

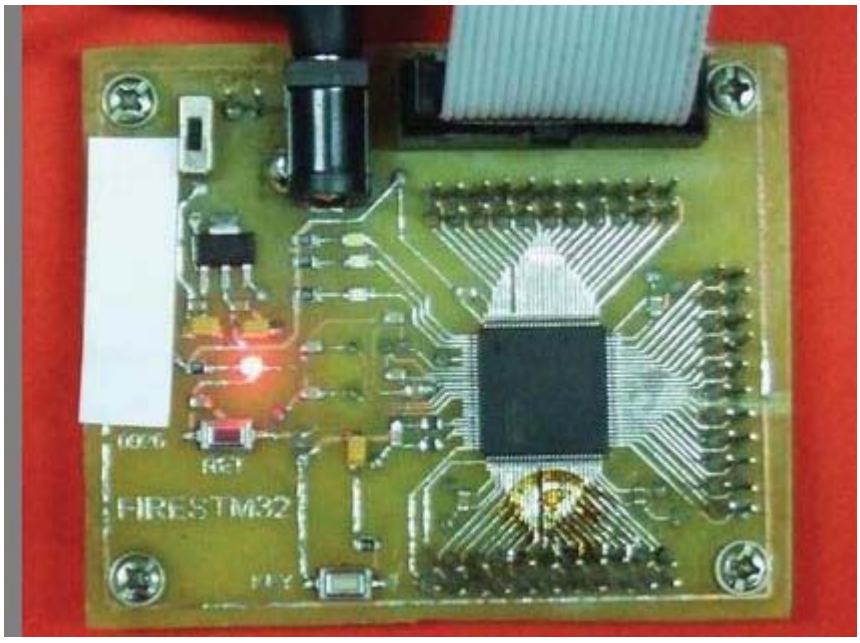
USER/ip_arp_udp_tcp.c

USER/web_server.c

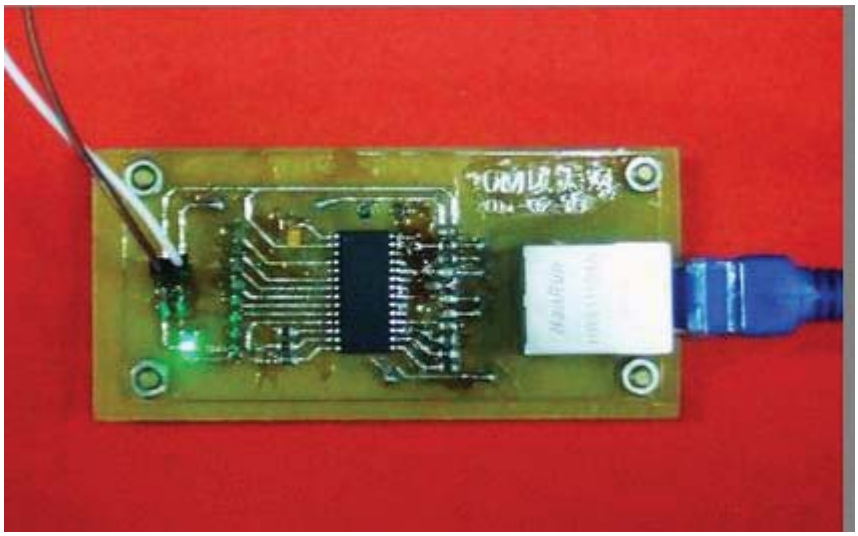
注：野火 **stm32** 开发板中板载的 **10M** 以太网 跟这里的接口是一样的。

纯手工野火 **stm32** 最小系统**+10M** 以太网**+MAX3232** 串口调试模块。

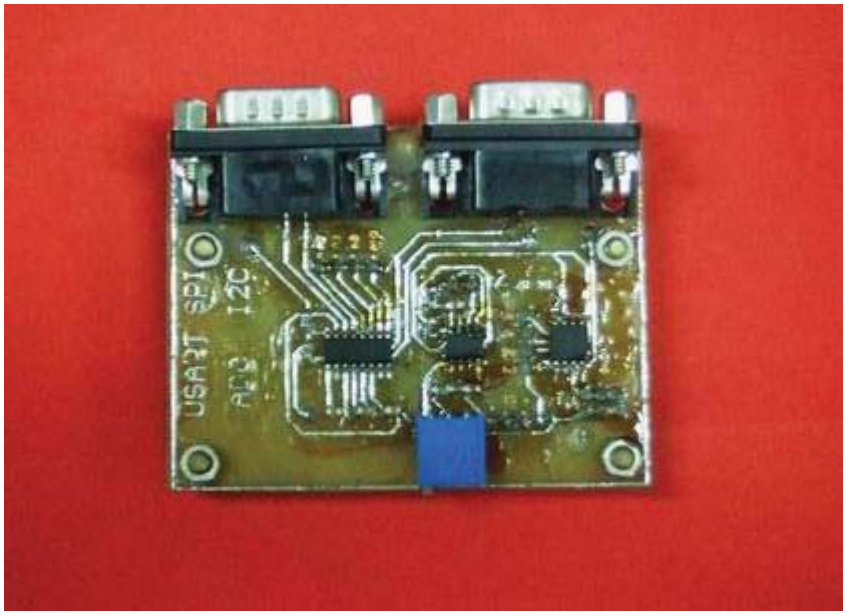
野火 **stm32** 最小系统



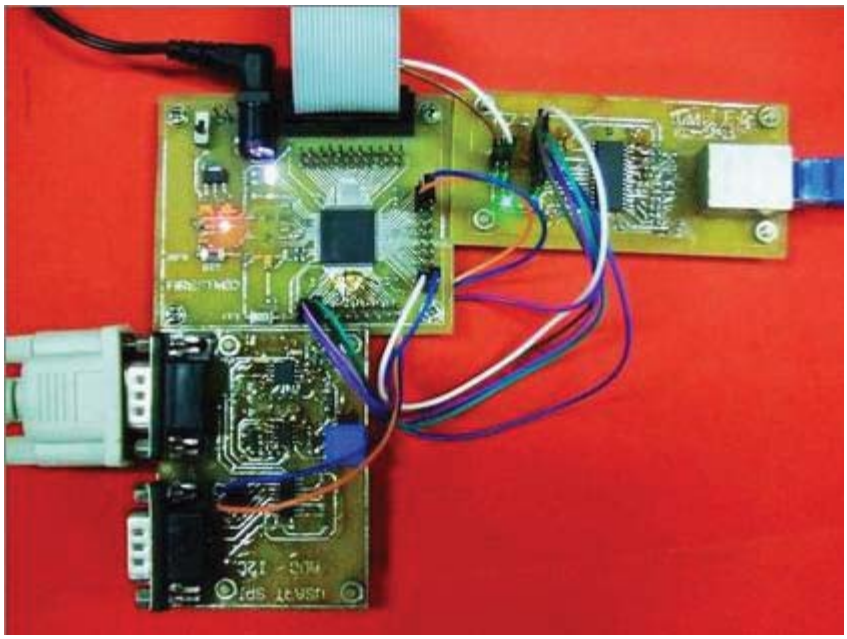
10M 以太网 ENC28J60



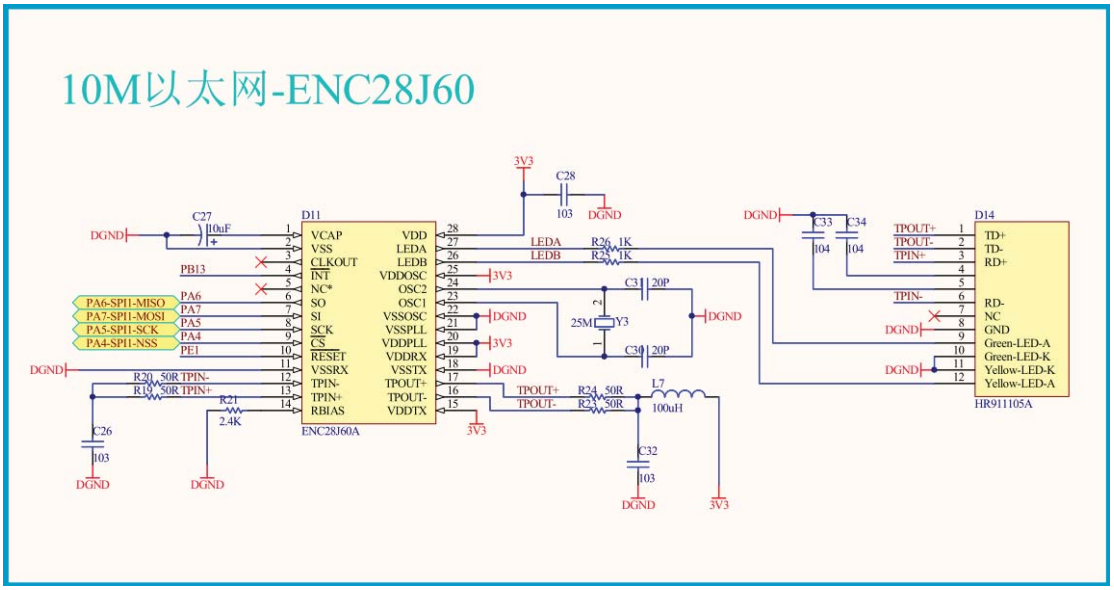
MAX3232（3.3V）串口调试模块



全家福



野火 **STM32** 开发板中 **10M** 以太网 **ENC28J60** 的硬件原理图：



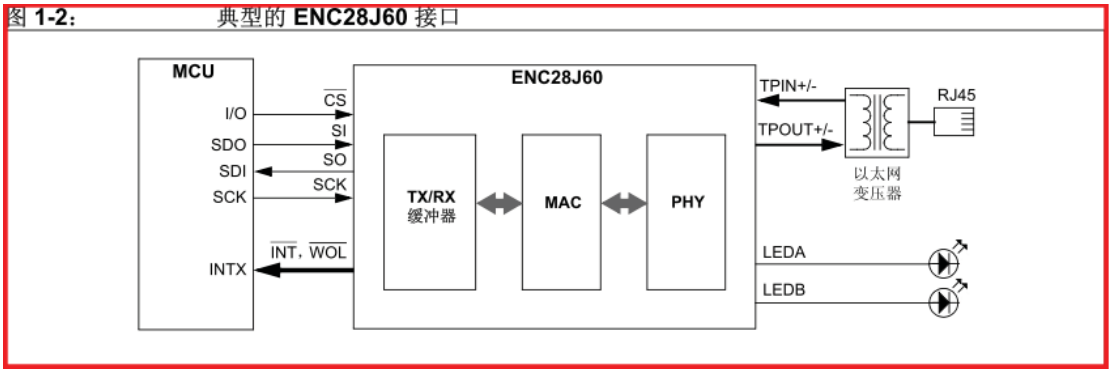
ENC28J60（以太网芯片）简介->

ENC28J60 是带有行业标准串行外设接口 （SerialPeripheral Interface，SPI）的独立以太网控制器。 它可作为任何配备有 SPI 的控制器的以太网接口。

ENC28J60 符合 IEEE 802.3 的全部规范，采用了一系列包过滤机制以对传入数据包进行限制。 它还提供了一个内部 DMA 模块，以实现快速数据吞吐和硬件支持的 IP 校验和计算。 与主控制器的通信通过两个中断引脚和 SPI 实现，数据传输速率高达 10 Mb/s。两个专用的引脚用于连接 LED，进行网络活动状态指示。

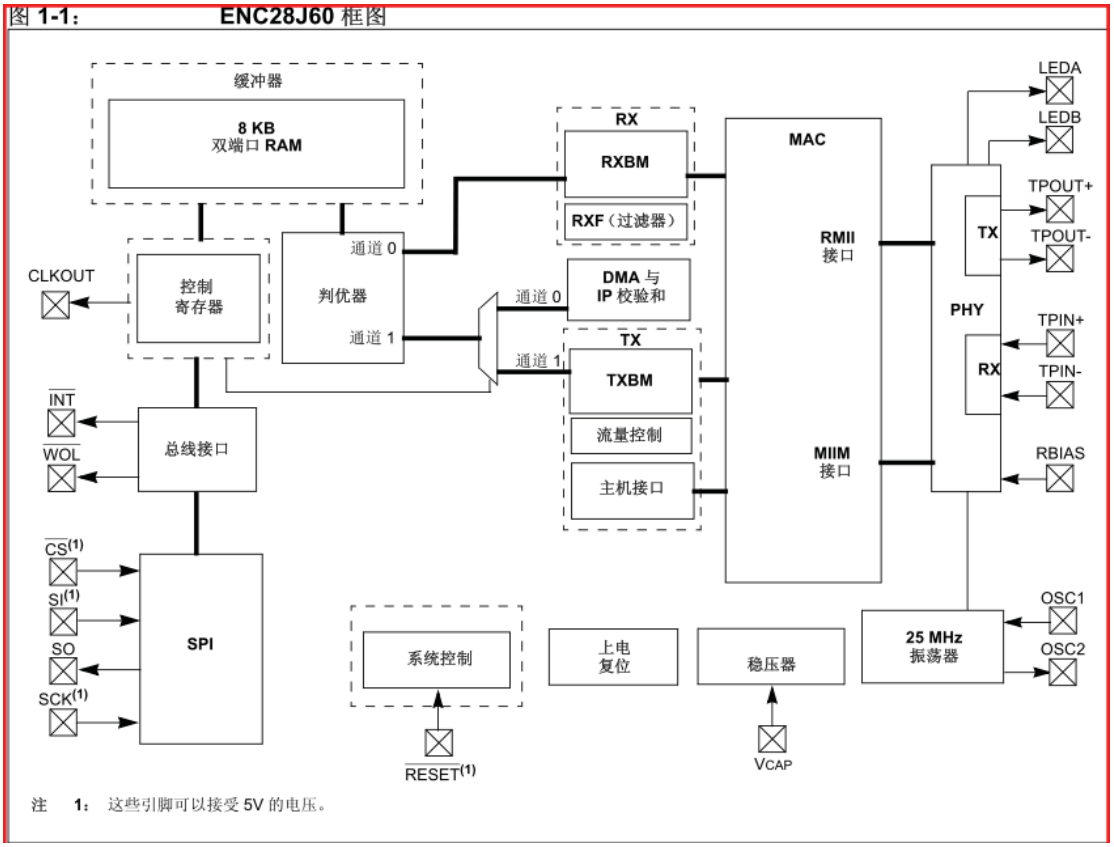
图 1-1 所示为 ENC28J60 的简化框图。 图 1-2 所示为使用该器件的典型应用电路。 要将单片机连接到速率为 10 Mbps 的以太网，只需 ENC28J60、两个脉冲变压器和一些无源元件即可。

图 1-2：　典型的 ENC28J60 接口



本开发板中用的网络变压器的型号为 **HR911105A**。

图 1-1：　ENC28J60 框图



ENC28J60 由七个主要功能模块组成：

1. SPI 接口——充当主控制器和 ENC28J60 之间通信通道。
2. 控制寄存器——用于控制和监视 ENC28J60。
3. 双端口 RAM 缓冲器——用于接收和发送数据包。
4. 判优器——当 DMA、发送和接收模块发出请求时对 RAM 缓冲器的访问进行控制。
5. 总线接口——对通过 SPI 接收的数据和命令进行解析。
6. MAC （Medium Access Control）模块——实现符合 IEEE 802.3 标准的 MAC 逻辑。
7. PHY（物理层）模块——对双绞线上的模拟数据进行编码和译码。该器件还包括其他支持模块，诸如振荡器、片内稳压器、电平变换器（提供可以接受 5V 电压的 I/O 引脚）和系统控制逻辑。

实验讲解开始->

建议阅读程序的顺序为：spi_enc28j60.c -> enc28j60.c -> ip_arp_udp_tcp.c -> web_server.c 。

spi_enc28j60.c ： ENC28J60(以太网芯片) SPI 接口应用函数库。

enc28j60.c ： Microchip ENC28J60 Ethernet Interface Driver。

ip_arp_udp_tcp.c ： IP, Arp, UDP and TCP functions（这部分野火仍在学习）。

web_server.c ： web 服务程序应用函数库。

其中 enc28j60.c 、ip_arp_udp_tcp.c web_server.c 是从 ATMEGA88 with ENC28J60 移植过来的，这是 AVR 的一块 ATMEGA88 评估板，源文件基本上没有做修改。spi_enc28j60.c 是由我们用户实现的底层函数接口，还有我们修改了 web_server.c 这个文件中网页命令控制部分的服务程序。

在配置好需要用的库文件之后，下面我们从 **main** 函数开始讲解，有关库函数是如何添加的情参考前面的教程，这里不再详述。

```
1. /*
2.  * 函数名：main
3.  * 描述  ：主函数
4.  * 输入  ：无
5.  * 输出  ：无
6.  */
7. int main (void)
8. {
9.   /* 配置系统时钟为 72M */
10.    SystemInit();
11.
12.    /* 配置 LED */
13.  LED_GPIO_Config();
14.
15.    /* ENC28J60 SPI 接口初始化 */
16.  SPI_Enc28j60_Init();
17.
18.    /* ENC28J60 WEB 服务程序 */
19.  Web_Server();
20.
21.    //return 0;
22. }
```

在进入 main 函数代码段后，我们首先调用系统库函数 `SystemInit();`将我们的系统时钟配置为 72MHZ。

`LED_GPIO_Config();`用于初始化 LED，因为我们我们在我们的 web 服务器中要控制的就是 LED，所以在这里要先把 LED 配置好，好让它接下来能工作。

`SPI_Enc28j60_Init();`用于配置以太网芯片 ENC28J60 所用到的数据通信口 SPI2 和其他控制 I/O。这是我们用户在 spi_enc28j60.c 中实现的底层程序。

```
1. /*
2.  * 函数名：SPI1_Init
3.  * 描述  ：ENC28J60 SPI 接口初始化
4.  * 输入  ：无
5.  * 输出  ：无
6.  * 返回  ：无
7.  */
8. void SPI_Enc28j60_Init(void)
9. {
10.    GPIO_InitTypeDef GPIO_InitStructure;
11.    SPI_InitTypeDef  SPI_InitStructure;
12.
13.    /* 使能 SPI1 时钟 */
14.    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA | RCC_APB2Periph_SPI1,
    ENABLE);
15.
16.   /*
17.    * PA5-SPI1-SCK :ENC28J60_SCK
18.    * PA6-SPI1-MISO:ENC28J60_SO
```

```
19.     * PA7-SPI1-MOSI:ENC28J60_SI
20.     */
21.     GPIO_InitStructure.GPIO_Pin = GPIO_Pin_5 | GPIO_Pin_6 | GPIO_Pin_7
    ;
22.     GPIO_InitStructure.GPIO_Speed = GPIO_Speed_10MHz;
23.     GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;       // 复用输出
24.     GPIO_Init(GPIOA, &GPIO_InitStructure);
25.
26.     /* PA4-SPI1-NSS:ENC28J60_CS */ // 片选
27.
28.     GPIO_InitStructure.GPIO_Pin = GPIO_Pin_4;
29.     GPIO_InitStructure.GPIO_Speed = GPIO_Speed_10MHz;
30.     GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;        // 推免输
    出
31.     GPIO_Init(GPIOA, &GPIO_InitStructure);
32.     GPIO_SetBits(GPIOA, GPIO_Pin_4);
33.
34.     /* PB13:ENC28J60_INT */ // 中断引脚没用到
35.     /* PE1:ENC28J60_RST*/    // 复位似乎不用也可以
36.
37.
38.     /* SPI1 配置 */
39.     SPI_InitStructure.SPI_Direction = SPI_Direction_2Lines_FullDuplex;

40.     SPI_InitStructure.SPI_Mode = SPI_Mode_Master;
41.     SPI_InitStructure.SPI_DataSize = SPI_DataSize_8b;
42.     SPI_InitStructure.SPI_CPOL = SPI_CPOL_Low;
43.     SPI_InitStructure.SPI_CPHA = SPI_CPHA_1Edge;
44.     SPI_InitStructure.SPI_NSS = SPI_NSS_Soft;
45.     SPI_InitStructure.SPI_BaudRatePrescaler = SPI_BaudRatePrescaler_8;

46.     SPI_InitStructure.SPI_FirstBit = SPI_FirstBit_MSB;
47.     SPI_InitStructure.SPI_CRCPolynomial = 7;
48.     SPI_Init(SPI1, &SPI_InitStructure);
49.
50.     /* 使能 SPI1  */
51.     SPI_Cmd(SPI1, ENABLE);
52. }
```

在这个函数中不知大家有没注意到没有这两条注释：

```
1.      /* PB13:ENC28J60_INT */ // 中断引脚没用到

2.

3.      /* PE1:ENC28J60_RST*/    // 复位似乎不用也可以
```

enc28j60 的中断引脚没用到很正常，但是复位引脚也没用到，这我就很纳闷了。我想原因可能是 enc28j60 有个上电自动复位的功能，这里它的复位引脚只能暂时没有用到而已，也或许是我们的开发板中引脚 PE1（接 enc28j60 的复位脚）收到什么信号的干扰，产生了类似复位的信号。这里我们先把这问题搁一边先，毕竟程序还是工作了。至于具体的原因我以后有时间再深究下。

`Web_Server();`函数实现的功能是创建一个网页服务器，在这个网页服务器上我可以点击我们设定好的命令按钮来控制我们开发板上 LED 的亮灭。其实，从这里面我们就可以看到有点智能家居的味道了，所谓智能家居就是通过网络来控制我们家电的状

态，如开和断。举个例子：我们可以在遥远的地方可以通过网络来控制家里的电视、电冰箱等，是不是很神奇哩？只要你学会了这个实验，再经过自己的再深入学习，这些都是小菜一碟。^_^，本实验旨在引导大家入门。

`Web_Server();`在 web_server.c 中实现。在 web_server.c 的开头包含了头文件：

```
1. #include "enc28j60.h"
2. #include "ip_arp_udp_tcp.h"
3. #include "net.h"
```

enc28j60.h ： Microchip ENC28J60 Ethernet Interface Driver Header file

ip_arp_udp_tcp.h： IP, Arp, UDP and TCP functions Header file . For more Infermation Please See http://www.gnu.org/licenses/gpl.html

net.h：Based on the net.h file from the AVRlib library by Pascal Stang.
For AVRlib See http://www.procyonengineering.com/
Used with explicit permission of Pascal Stang.

有关这三个头文件对应的 C 文件的功能，请大家阅读源码。

接下来我们具体看看 `Web_Server();`这个函数具体做了什么，由于这个函数的源码较多，在这里我就不贴出来了。

enc28j60Init(mymac); 这个函数用来初始化 enc28j60 的 MAC 地址(物理地址),这个函数必须要调用一次。mymac 在 web_server.c 的开头定义：

static unsigned char mymac[6] = {0x54,0x55,0x58,0x10,0x00,0x24}; 这里要注意的是：mac 地址在局域网内必须唯一，否则将与其他主机冲突，导致连接不成功。mymac 数组里面的数值可随便初始化，但万万不可与局域网内的 mac 地址冲突，否则当另外一部主机在上网时，你是上不了网的。

enc28j60PhyWrite(PHLCON,0x476);这个函数用于设定网络变压器中 LED 的颜色，不同的颜色指示不同的状态。网络变压器在没有工作的情况下，这两个 LED 是不会被点亮的，当网络变压器工作正常时，绿色 LED 表示 link 状态，黄色 LED 表示通信状态。本实验中，我们的程序工作正常时，绿色 LED 常亮，黄色 LED 是一闪一闪的。

init_ip_arp_udp_tcp(mymac,myip,mywwwport); 这个函数用于初始化以太网的 IP 层。这里面涉及到三个参数：mymac、myip、mywwwport。其中 mymac 在前面已经讲解过。ip 指的是我们开发板的 ip 地址，要想通过网页来访问我们的服务器(即开发板)，则必须要有 ip。ip 地址跟 mac 地址一样，在局域网能要保持唯一，不能够与其他主机的 ip 地址产生冲突。还要注意的一点就是：开发板的 ip 与我们电脑的 ip 必须保持在同一个网段，即 ip 地址的前三段要保持一致，后面一段不同。在本地链接中可查看到我们电脑的 ip 地址。我的电脑的 ip 地址是：192.168.1.106，如下截图所示：

所以，在此设置我们开发板的 ip 为 static unsigned char myip[4] = {192,168,1,15}; ip 的最后一段的值 15 可改为其他值，但要注意不要产生冲突。

接下来的是一个无限循环 while （ 1 ），在这里面我们创建了一个网页，在网页中注入了我们自己的信息，并随时监控我们命令状态的改变，好实时地控制我们的 LED，有关这些功能的实现请大家自行阅读源码，这里不再详述。我们仅来看看 LED 控制的部分：

```
1. if (cmd==1)      // 用户程序
2. {
3.     LED1(ON);
4.     i=1;         // 命令 = 1
5. }
6. if (cmd==0)      // 用户程序
7. {
8.     LED1(OFF);
9.     i=0;           // 命令 = 0
10.}
```

当我们在网页上点击 点亮 LED 这个按钮时，网页发送命令 1 给开发板，这是开发板中的 LED 被点亮，反之，LED 则被关闭。

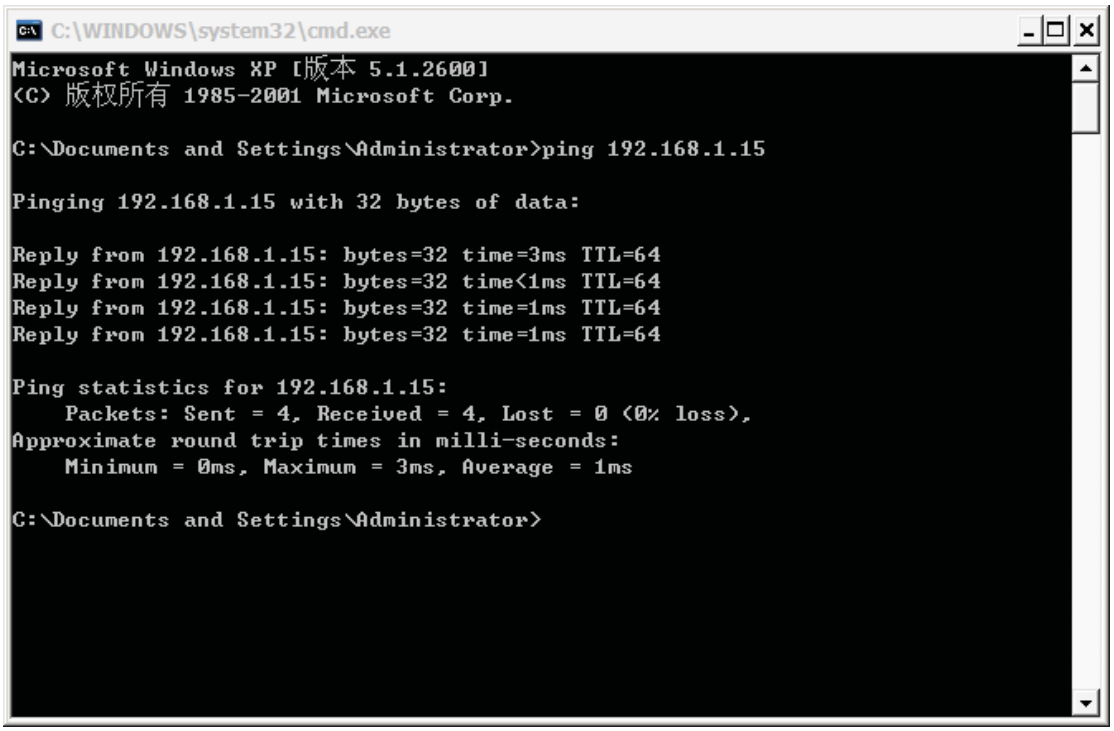
这里我们仅提供简单的 web 服务程序，如果要实现更加复杂的功能，仍需大家的努力。当然，我们工作室也会跟大家一起奋斗，开发出更多的应用程序跟大家分享。

这里面的操作涉及到很多 enc28j60 的知识，特别是寄存器的操作，具体的请大家参考 enc28j60 的 datasheet，大家一定要看看，而且是要认认真真、仔仔细细地看，直到弄懂为止。

实验现象->

将野火 STM32 开发板供电(DC5V)，插上 JLINK，插上网口线，网口线一端连接路由器，一端连开发板。注意：电脑跟开发板的网线连接的路由器要同在一个局域网中。将编译好的程序下载到开发板。

1：打开电脑的 DOS 界面，输入： ping 192.168.1.15 ，看看能否 ping 通。打印出如下信息则表示 ping 通，其中 192.168.1.15 是我们开发板的 ip。



2：打开 IE 浏览器，在地址栏输入：http://192.168.1.15/123456 ，其中 123456 是我们自己设置的密码。然后按 Enter 键进入，如果成功则会出现一个如下网页，通过网页中的命令按钮 点亮 LED /关闭 LED 就可以控制我们 开发板中 LED 的亮灭，这里控制的是开发板中的 LED1。

LED1 灭

LED1 亮