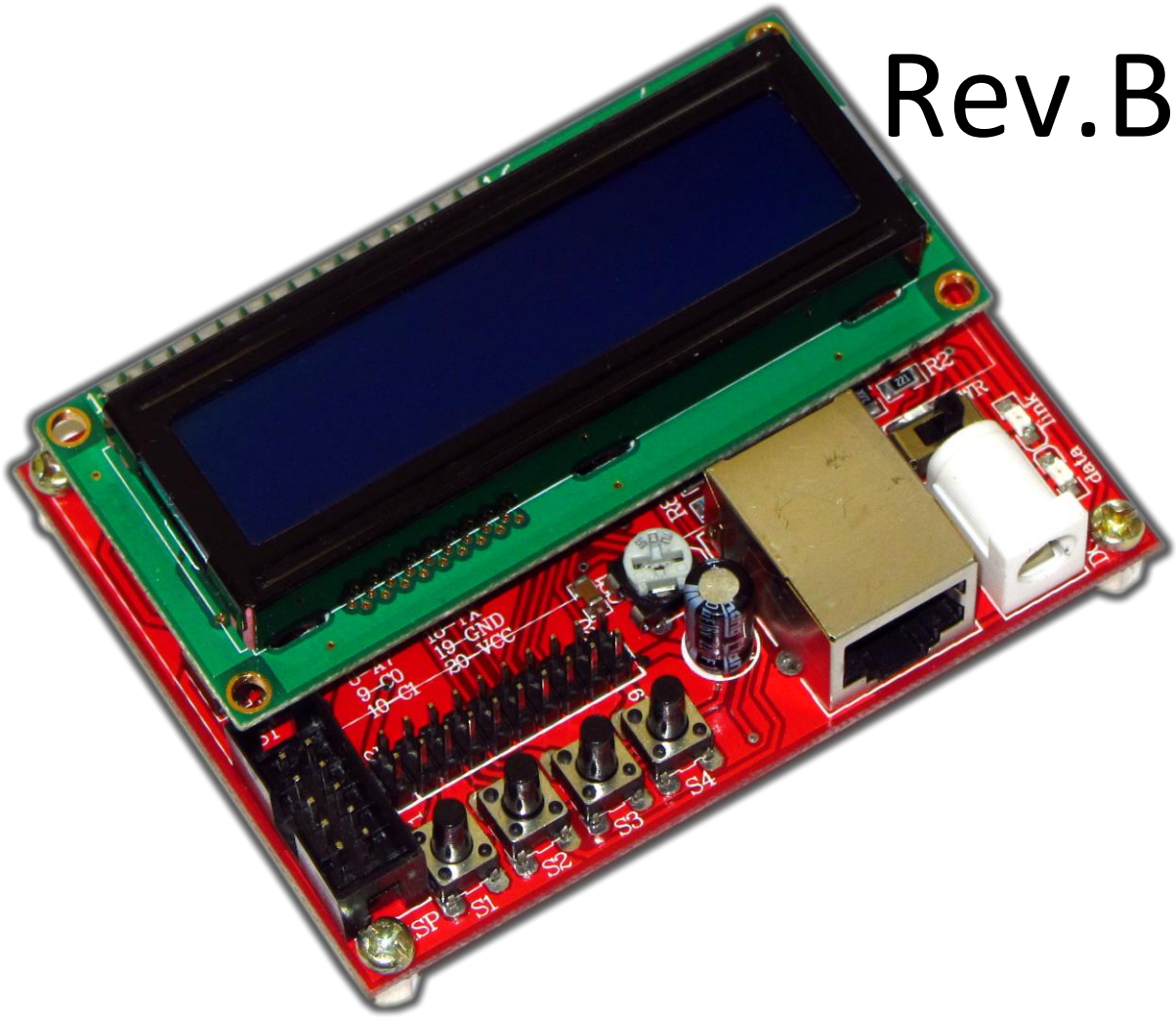
AVR Ethernet Board

Rev.B

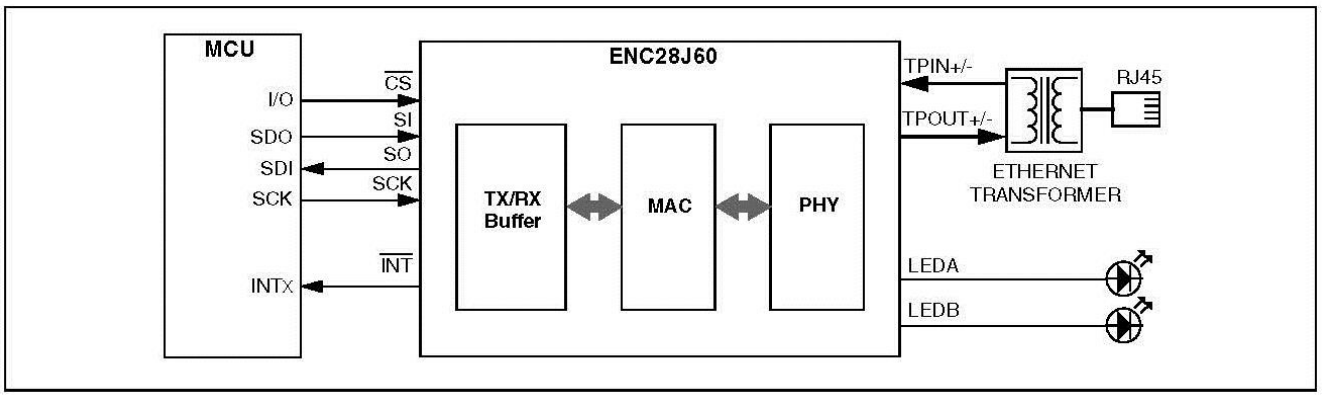


ENC28J60 یک کنترلر اترنت مستقل با رابط استاندارد صنعتی SPI می باشد. این کنترلر به عنوان یک رابط شبکه اترنت برای تمام میکروکنترلرهای مجهز به SPI طراحی شده است.

ENC28J60 با تمام ویژگی های IEEE 802.3 سازگار می باشد و همچنین قابلیت فیلتر کردن بسته های ورودی را برای محدود کردن آنها دارد. این کنترلر همچنین شامل حافظه DMA برای افزایش سرعت کار با داده ها و انجام محاسبات checksum می باشد. ارتباط با میکروکنترلر از طریق از طریق رابط SPI با حداکثر سرعت 10 Mbit/s می باشد.

فروشگاه تخصصی برق و الکترونیک www.eShop.ECA.ir ایمیل: eShop.ECA@Gmail.com
شماره تماس: ۰۴۱۱۵۵۳۳۷۳۲ فکس: ۰۴۱۱۵۵۳۹۷۶۹

بلوک دیاگرام :



برای ارتباط با ENC28J60 از میکروکنترلر Mega32A استفاده شده است.

برای اینکه بتوانیم با شبکه اترنت ارتباط برقرار کنیم باید مدل چهار لایه ای TCP/IP را روی میکروکنترلر AVR پیاده سازی کنیم. در پروژه ضمیمه شده لایه ها و پروتکل های لازم برای ایجاد یک وب سرور ساده در قالب هدرفایل قرار دارند. در بخش زیر با توضیح مختصر وظایف هر لایه TCP/IP فایلهای موبوط به هر لایه را مشخص می کنیم.

بررسی لایه های مدل TCP/IP :

- لایه ی اول : لایه ی واسط شبکه

در این لایه استانداردهای سخت افزار، و نرم افزارهای راه انداز (Device Driver) و پروتکل های شبکه تعریف می شود. این لایه درگیر با مسائل فزیکی، الکتریکی و مخابراتی کانال انتقال، نوع کنترلر شبکه و راه اندازهای لازم برای کنترلر شبکه می باشد.

در این پروژه کنترلر شبکه ENC28J60 بوده و توابع لازم برای راه اندازی این آی سی و ارسال و دریافت بسته های داده در این لایه در فایل enc28j60.c تعریف شده اند.

- لایه ی دوم : لایه ی شبکه

این لایه در ساده ترین عبارت وظیفه دارد بسته های اطلاعاتی را که از این به بعد آنها را بسته های IP می نامیم، روی شبکه هدایت کرده و از مبداء تا مقصد به پیش ببرد. در این لایه چندین پروتکل در کنار هم وظیفه ی مسیریابی و تحویل بسته های اطلاعاتی از مبداء تا مقصد را انجام می دهند. کلیدی ترین پروتکل در این لایه، پروتکل IP نام دارد. برخی از پروتکل های مهم که یک سری وظایف جانبی بر عهده دارند عبارتند از : IGMP – RARP – RIP – ICMP – BOOTP – ARP و ... .

وظیفه پروتکل ARP تبدیل IP به MAC Adress می باشد. پروتکل ICMP نیز در کنار پروتکل **IP**، برای بررسی انواع خطا و ارسال پیام برای مبدا بسته در هنگام بروز اشکالات ناخواسته استفاده می شود. در حقیقت ICMP یک سیستم گزارش خطا است که بر روی پروتکل IP نصب می شود تا در صورت بروز هر گونه خطا به فرستنده بسته پیام مناسب را بدهد تا آن خطا تکرار نشود.

فایل های ip.c ، arp.c و icmp.c برای پیاده سازی این لایه می باشند.

- لایه ی سوم : لایه انتقال

این لایه ارتباط ماشین های انتهایی (ماشینهای میزبان) را در شبکه برقرار می کند، یعنی می تواند بر اساس سرویسی که لایه دوم ارائه می کند یک ارتباط اتصال گرا و مطمئن برقرار کند. البته در این لایه برای عملیاتی نظیر ارسال صوت و تصویر که سرعت، مهمتر از دقت و خطا است سرویس های بدون اتصال سریع و نامطمئن نیز فراهم شده است.

در سرویس مطمئنی که در این لایه ارائه می شود، مکانیزمی اتخاذ شده است که فرستنده از رسیدن و یا عدم رسید صحیح بسته به مقصد با خبر شود.

این لایه از یک طرف با لایه ی شبکه و از طرف دیگر با لایه ی کاربرد در ارتباط است.

پروتکل های TCP و UDP در این لایه تعریف شده است و فایلهای tcp.c و udp.c برای پیاده سازی این پروتکل ها می باشند.

- لایه ی چهارم : لایه ی کاربرد

در این لایه بر اساس خدمات لایه های زیرین، سرویس سطح بالایی برای خلق برنامه های کاربردی ویژه و پیچیده ارائه می شود. این خدمات در قالب پروتکل های استانداردی همانند موارد زیر به کاربر ارائه می شود :

- شبیه سازی ترمینال (TELNET / Terminal Emulation)
- انتقال فایل یا FTP (File Transfer Protocol)
- مدیریت پست الکترونیکی
- خدمات انتقال صفحات ابرمتنی HTML
- ...

فایل http.c وظیفه پیاده سازی این لایه برای پروتکل HTTP و ایجاد یک صفحه وب را بر عهده دارد و تغییرات مورد نیاز برای سفارشی کردن صفحه وب در این فایل انجام می شود.

تابع http_home موجود در فایل http جهت ایجاد فایل HTML می باشد. با برنامه نمونه ای که گذاشته شده است امکان کنترل دو LED متصل به پین های ۲ و ۳ از PORTA وجود دارد. همچنین دو ورودی متن نیز در صفحه HTML ایجاد میشود که متن lcd کاراکتری متصل به میکرو را با استفاده از آنها می توان تغییر داد.

کدهای HTML به صورت سطر به سطر و توسط تابع tcp_puts_data_p به درخواست کننده ارسال می شوند.

به عنوان مثال در صورتیکه بخواهیم یک صفحه ساده HTML که تنها شامل یک لینک می باشد ایجاد کنیم تابع http_home باید به صورت زیر تغییر کند :

```
uint16_t http_home( uint8_t *rxtx_buffer ) {

uint16_t dlen;

dlen = tcp_puts_data_p ( rxtx_buffer, PSTR ( "HTTP/1.0 200 OK\r\nContent-Type:
text/html\r\n\r\n" ), 0 );

dlen = tcp_puts_data_p ( rxtx_buffer, PSTR ( "<title>" ), dlen );

dlen = tcp_puts_data_p ( rxtx_buffer, (PGM_P)web_title, dlen );

dlen = tcp_puts_data_p ( rxtx_buffer, PSTR ( "</title>" ), dlen );

dlen = tcp_puts_data_p ( rxtx_buffer, PSTR ( "<a href=\"http://www.ECA.ir/\"
target=\"_blank\"><b><font color=\"#000099\" size=\"+1\">" ), dlen );

return(dlen);

}
```

اگر بخواهیم لینک هایی ایجاد کنیم که عملکرد میکرو را کنترل کنند (مثلا روشن و خاموش کردن LED) باید از حالت زیر در تعریف لینک ها استفاده کنیم :

```
dlen = tcp_puts_data_p ( rxtx_buffer, PSTR ( "<a href=\"./?l1=" ), dlen );
```

کلید l1 که در این لینک تعریف شده است برای مشخص کردن یک عملکرد خاص می باشد و بعدا برای تشخیص عملکردهای مختلف که توسط لینک ها ایجاد می شوند استفاده خواهد شد. بعد از کاراکتر مساوی عدد یا رشته ای قرار می گیرد که نشان خواهد داد مثلا آن LED باید روشن یا خاموش شود. مثلا برای روشن کردن LED کد بالا به صورت زیر تغییر می کند:

```
dlen = tcp_puts_data_p ( rxtx_buffer, PSTR ( "<a href=\"./?l1=۱" ), dlen );
```

بررسی درخواست های HTTP در تابع http_webserver_process موجود در فایل http.c انجام می شود. به عنوان مثال برای کنترل LED1 از کدهای زیر در تابع استفاده شده است :

```
if ( http_get_variable ( rxtx_buffer, dlength, PSTR( "l1" ), generic_buf ) )
        {
if ( generic_buf[0] == '0' )
                LED_PORT |= _BV ( LED_PIN1 );
        else
                LED_PORT &= ~_BV ( LED_PIN1 );
}
```

همانطوریکه مشاهده می کنید با استفاده از شرط if وجود کلید l1 در درخواست http بررسی می شود و اگر درخواست مربوط به کلید l1 باشد عملکرد مورد نظر روی پین میکروی متصل به LED اعمال می شود.

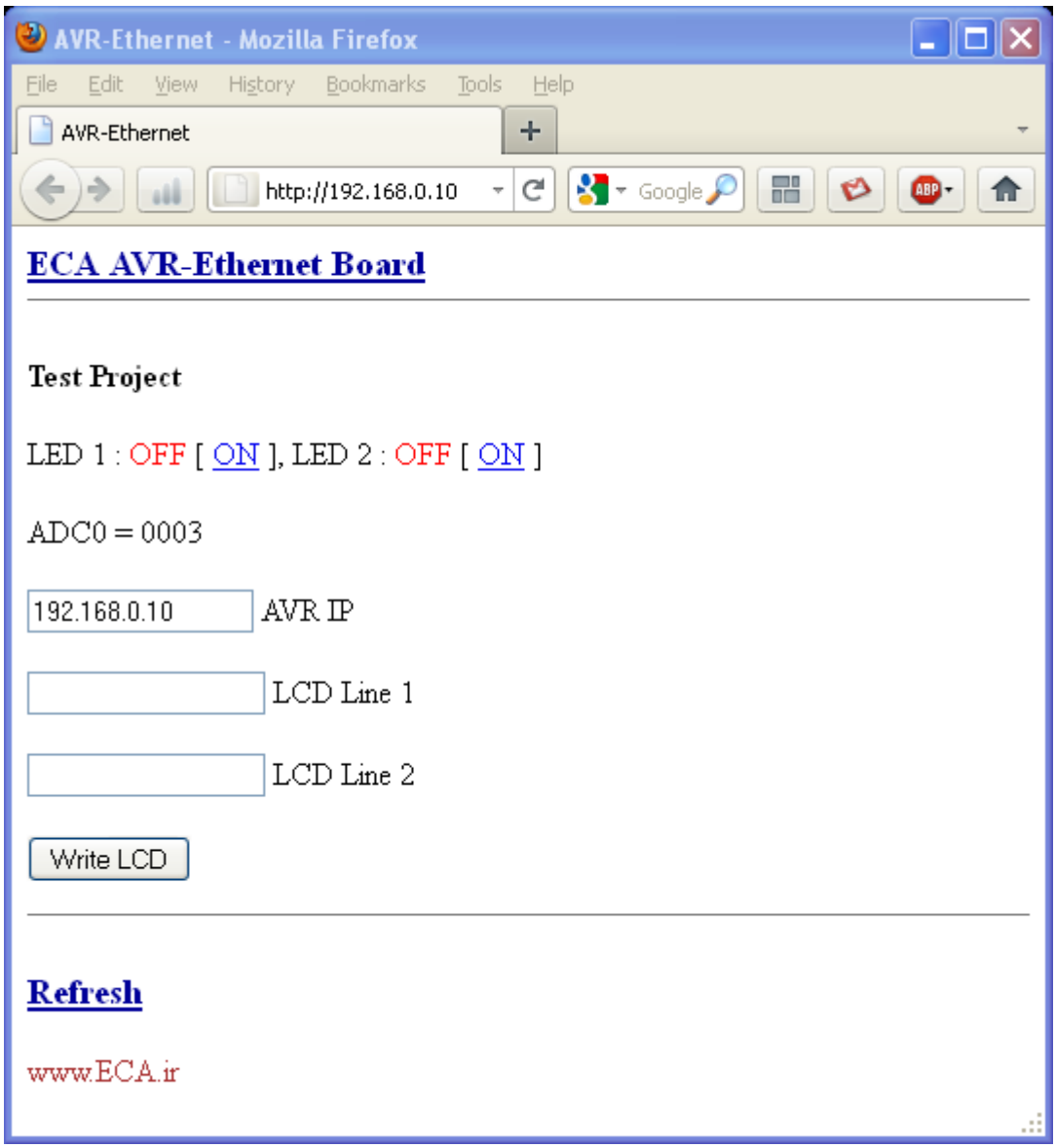
در تابع http_home برای ایجاد یک جعبه متن و گرفتن یک رشته از کد زیر استفاده میکنیم :

```
dlen = tcp_puts_data_p ( rxtx_buffer, (PGM_P) tag_form, dlen );

dlen = tcp_puts_data_p ( rxtx_buffer, PSTR ( "<input name=\"lcd1\" type=\"text\"
size=\"16\" maxlength=\"16\"> LCD Line 1<br><br>" ), dlen );

dlen = tcp_puts_data_p ( rxtx_buffer, PSTR ( "<input type=\"submit\" value=\"Write
LCD\"></form>" ), dlen );
```

کلید submit برای ارسال محتویات متن به سرور می باشد. کدهای مورد نیاز برای پاسخ دادن به این درخواست و نمایش رشته گرفته شده روی LCD به صورت زیر می باشد :

```
if ( http_get_variable ( rxtx_buffer, dlength, PSTR( "lcd1" ), generic_buf))
}
        urldecode ( generic_buf );
        lcd_putc ( '\f;( '
        lcd_print ( generic_buf );
        flag1.bits.lcd_busy = 1;
{
```

برای تست مدار ابتدا سورس برنامه را توسط کامپایلر WinAVR کامپایل نموده و توسط پروگرامر روی میکرو پروگرام نمائید. تنظیمات فیوز بیت را روی کریستال 8MHz خارجی قرار دهید. مدار را توسط کابل شبکه به پورت LAN کامپیوتر وصل کرده و روشن کنید. با استفاده از کلیدهای جهت نمای بالا وارد منوی تنظیمات و بخش AVR IP Config شوید و IP را طبق رنج IP شبکه خود تنظیم کنید. برای مثال اگر رنج IP شبکه 192.168.0.xx باشد IP را روی ۱۹۲,۱۶۸,۰,۱۰ تنظیم کنید. مروگر اینترنتی خود را باز کنید و IP برد را وارد کنید. حال باید صفحه ای شبیه تصویر زیر را روی مرورگر مشاهده کنید.


AVR-Ethernet - Mozilla Firefox
File Edit View History Bookmarks Tools Help
AVR-Ethernet
http://192.168.0.10
Google
ECA AVR-Ethernet Board
Test Project
LED 1 : OFF [ON], LED 2 : OFF [ON]
ADC0 = 0003
192.168.0.10
AVR IP
LCD Line 1
LCD Line 2
Write LCD
Refresh
www.ECA.ir