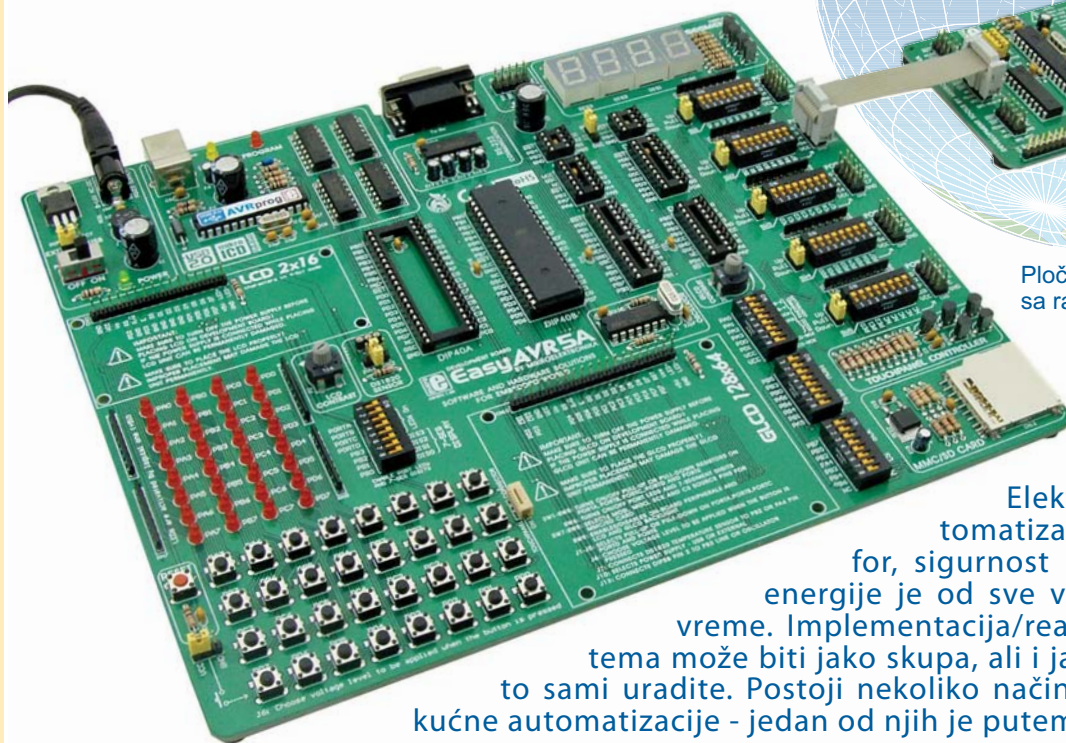


# OK. Vama je potreban ... ETHERNET



Ploča serijskog etneta povezana sa razvojnim sistemom EasyAVR5A

Elektronski sistem kućne automatizacije je sinonim za komfor, sigurnost i uštedu energije. Ušteda energije je od sve većeg značaja u poslednje vreme. Implementacija/realizacija/ugradnja ovih sistema može biti jako skupa, ali i jako jeftina ako odlučite da to sami uradite. Postoji nekoliko načina da upravljate sistemom kućne automatizacije - jedan od njih je putem Eterneta.

Srdan Tomić  
MikroElektronika - Software Department

Sve što je potrebno je jedan mikrokontroler Atmega 32 i serijski ethernet čip ENC28J60. Serijska komunikacija čini ovaj čip odličnim rešenjem i za mikrokontrolere iz drugih familija kao što su dsPIC, PIC itd. Za povezivanje na ethernet mrežu korišćen je RJ-45 konektor firme CviLux pod oznakom CJC8 A8HF1Y0. LED dioda povezana na PORTA.0 mikrokontrolera predstavljaće uređaj u kući kojim želimo da upravljamo.

Kompajler *mikroC PRO for AVR* sadrži ethernet biblioteku koja će nam uveliko olakšati pravljenje samog programa za mikrokontroler. Uz pomoć svega par rutina iz ove biblioteke, napravićemo program koji će nam omogućiti paljenje i gašenje uređaja u kući putem web browser-a.

U programu je potrebno uraditi sledeće:

- Korak 1.** Napraviti html stranicu preko koje će se upravljati mikrokontrolerom i ubaciti je u kod u vidu string-a.
- Korak 2.** Setovati IP, DNS, Gateway adrese i Subnet masku dobijenu od strane internet provajdera.

Primeru radi, parametri naše lokalne mreže su:

**IP:** 192.168.20.60 (adresa kontrolnog sistema)  
**DNS:** 192.168.20.1 (adresa Domain Name Sistema)  
**GATEWAY:** 192.168.20.6 (adresa Gateway-a)  
**SUBNET:** 255.255.255.0 (subnet maska)

**Korak 3.** Ugasiti analogne ulaze na portu A mikrokontrolera. Postaviti nulu na pin PORTA.0 i setovati ga kao izlazni.

**Korak 4.** Inicijalizovati SPI modul mikrokontrolera Atmega 32.

**Korak 5.** Inicijalizovati serijski ethernet modul ENC28J60.

**Korak 6.** Unutar funkcije `Spi_Ethernet_userTCP` napisati kod koji će po prijemu komande poslate putem web browser-a upaliti/ugasiti LED diodu.

**Korak 7.** U neprekidnoj petlji iščitavati primljene podatke.

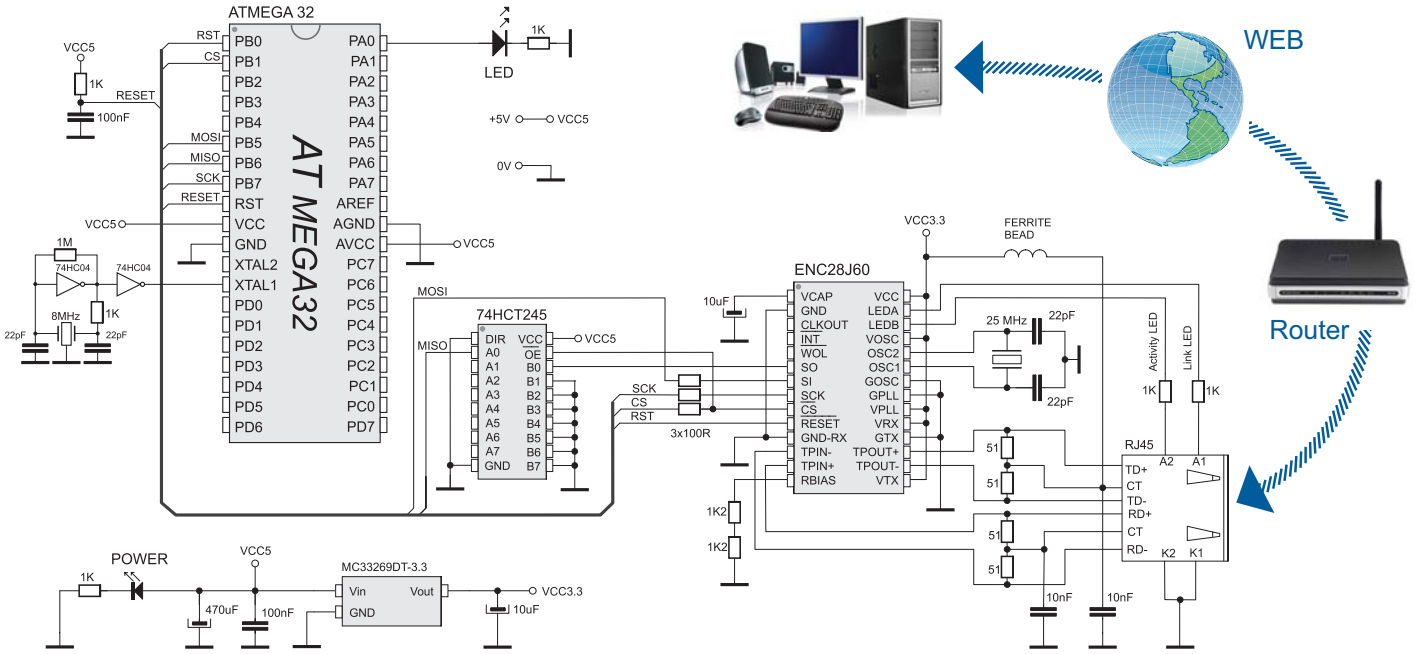
Najvažnija funkcija je `Spi_Ethernet_userTCP` u kojoj se odvija procesiranje svih primljenih komandi. Nakon prijema 'GET' zahteva poslatog sa vašeg računara putem web browser-a na IP adresu

kontrolnog sistema, mikrokontroler će browser-u vratiti web stranicu koja se nalazi u njegovoj memoriji. To je upravo ono što će biti prikazano na monitoru vašeg računara. Po prijemu ON komande LED dioda se pali, a po prijemu OFF komande ista se gasi. Ukoliko se umesto diode stavi relej može se ostvariti kontrola bilo kog uređaja kao što je svetlo, alarm, grejanje itd.

Sama kontrola se svodi na kucanje IP adrese kontrolnog sistema koji upravlja uređajima u kući u web browser i zadanje željenih komandi. Naravno, kako se kontroliše jedan pin mikrokontrolera tako se može kontrolisati i više pinova čime se dolazi do kompleksne kontrole većeg broja kućnih uređaja, odnosno, do sistema kućne automatizacije.

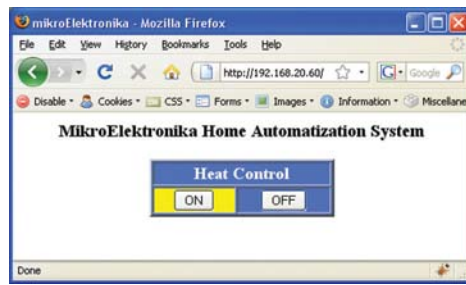


Slika 1. Pločica serijskog etneta sa ENC28J60 čipom



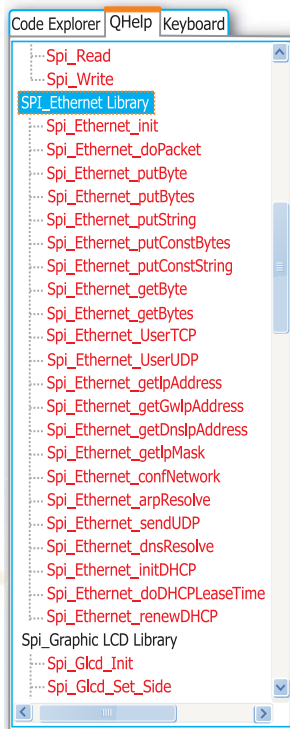
Šema 1. Povezivanje pločice serijskog ethernet i mikrokontrolera Atmega 32

Na slici 2 je prikazana stranica koja se dobija u web browser-u nakon unošenja IP adrese kontrolnog sistema. U opisanom primeru, klikom na tastere ON i OFF dolazi do paljenja i gašenja LED diode čime se simulira upravljanje sistemom za grejanje.



Slika 2. Stranica web browser-a

SPI Ethernet Library biblioteka sa gotovim funkcijama sadržana u kompajleru mikroC PRO for AVR.



<b>Spi_Ethernet_Init()</b>	Inicijalizuje ENC28J60 kontroler
<b>Spi_Ethernet_Enable()</b>	Omoгуčuje mrežni saobraćaj
<b>Spi_Ethernet_Disable()</b>	Omoгуčuje mrežni saobraćaj
<b>Spi_Ethernet_doPacket()</b>	Obraduje dobijeni paket podataka
<b>Spi_Ethernet_putByte()</b>	Upisuje bajt
<b>Spi_Ethernet_putBytes()</b>	Upisuje zahtevane bajtove
<b>Spi_Ethernet_putConstBytes()</b>	Upisuje zahtevane const bajtove
<b>Spi_Ethernet_putString()</b>	Upisuje string
<b>Spi_Ethernet_putConstString()</b>	Upisuje const string
<b>Spi_Ethernet_getByte()</b>	Uzima bajt
<b>Spi_Ethernet_getBytes()</b>	Uzima zahtevane bajtove
<b>Spi_Ethernet_UserTCP()</b>	Obraduje TCP
<b>Spi_Ethernet_UserUDP()</b>	Obraduje UDP
<b>Spi_Ethernet_getIpAddress()</b>	Uzimanje IP adrese
<b>Spi_Ethernet_getGwIpAddress()</b>	Uzimanje Gateway adrese
<b>Spi_Ethernet_getDnsIpAddress()</b>	Uzimanje DNS adrese
<b>Spi_Ethernet_getIpMask()</b>	Uzimanje IP maske
<b>Spi_Ethernet_confNetwork()</b>	Postavlja mrežne parametre
<b>Spi_Ethernet_arpResolve()</b>	Šalje zahtev za ARP
<b>Spi_Ethernet_sendUDP()</b>	Šalje UDP
<b>Spi_Ethernet_dnsResolve()</b>	Šalje zahtev za DNS
<b>Spi_Ethernet_initDHCP()</b>	Šalje zahtev za DHCP
<b>Spi_Ethernet_doDHCPLeaseTime()</b>	Obrada DHCP validnosti
<b>Spi_Ethernet_renewDHCP()</b>	Zahtev za DHCP obnovu

Spisak dodatnih funkcija korišćenih u programu:

<b>Spi_Init()</b>	Inicijalizacija modula na mikrokontroleru
<b>memcpy()</b>	Kopira RAM lokacije mikrokontrolera
<b>memcmp()</b>	Upoređuje RAM lokacije mikrokontrolera

Primer 1: Program demonstracije rada serijskog ethernet

```
// duplex config flags
#define Spi_Ethernet_HALFDUPLEX 0x00 // half duplex
#define Spi_Ethernet_FULLDUPLEX 0x01 // full duplex

// mE ethernet NIC pinout
sfr sbit SPI_Ethernet_Rst at PORTB.B0; // reset pin
sfr sbit SPI_Ethernet_CS at PORTB.B1; // chip select pin
sfr sbit SPI_Ethernet_Rst_Direction at DDRB.B0; // reset pin direction
sfr sbit SPI_Ethernet_CS_Direction at DDRB.B1; // chip select pin direction
// end ethernet NIC definitions

const char httpHeader[] = "HTTP/1.1 200 OK\r\nContent-type: "; // HTTP header
const char httpMimeTypeHTML[] = "text/html\r\n"; // HTML MIME type
const char httpMimeTypeScript[] = "text/plain\r\n"; // TEXT MIME type

// default html page
char indexPage[] =
"<html><head><title>mikroElektronika</title></head><body>\r\n"
"<h3 align=center>MikroElektronika Home Automatization System</h3>\r\n"
"<form name='input' action='\"/\"' method='\"get\"'\r\n"
"<table align=center width=200 bgcolor=#4974E2 border=2><tr>\r\n"
"<td align=center colspan=2><font size=4 color=white><b>Heat Control</b></font>\r\n"
"</td></tr><tr><td align=center bgcolor=#4974E2><input name='tst1' width=60\r\n"
" type='\"submit\"' value='\"ON\"'\r\n"
"<td align=center bgcolor=#4974E2><input name='tst2' type='\"submit\"' value='\"OFF\"'\r\n"
"</td></tr></table>\r\n"
"</form></body></html>";

// network parameters
char myMacAddr[6] = {0x00, 0x14, 0xA5, 0x76, 0x19, 0x3f}; // my MAC address
char myIpAddr[4] = {192, 168, 20, 60}; // my IP address
char gwIpAddr[4] = {192, 168, 20, 6}; // gateway IP address
char dnsIpAddr[4] = {192, 168, 20, 1}; // dns IP address
char ipMask[4] = {255, 255, 255, 0}; // subnet mask
// end network parameters

unsigned char getRequest[20]; // HTTP request buffer

unsigned int SPI_Ethernet_UserTCP( char *remoteHost, unsigned int remotePort,
unsigned int localPort, unsigned int reqLength)
{
    unsigned int len; // my reply length
    if(localPort != 80) return(0); // I listen only to web request on port 80

    // get 10 first bytes only of the request, the rest does not matter here
    for(len = 0; len < 15; len++) getRequest[len] = SPI_Ethernet_getByte();
    getrequest[len] = 0;

    if(memcmp(getRequest, "GET /", 5)) return(0); // only GET method

    if(!memcmp(getRequest+11, "ON", 2)) // do we have ON command
        PORTA.F0 = 1; // set PORTA bit0
    else // do we have OFF command
        if(!memcmp(getRequest+11, "OFF", 3)) // clear PORTA bit0
            PORTA.F0 = 0;

    if(PORTA.F0)
    {
        memcpy(indexPage+340, "#FFFF00", 6); // highlight (yellow) ON
        memcpy(indexPage+431, "#4974E2", 6); // clear OFF
    }
    else
    {
        memcpy(indexPage+340, "#4974E2", 6); // clear ON
        memcpy(indexPage+431, "#FFFF00", 6); // highlight (yellow) OFF
    }

    len = SPI_Ethernet_putConstString(httpHeader); // HTTP header
    len += SPI_Ethernet_putConstString(httpMimeTypeHTML); // with HTML MIME type
    len += SPI_Ethernet_putString(indexPage); // HTML page first part
    return len; // return to the library with the number of bytes to transmit
}

unsigned int SPI_Ethernet_UserUDP( char *remoteHost, unsigned int remotePort,
unsigned int destPort, unsigned int reqLength)
{
    return 0; // back to the library with the length of the UDP reply
}

void main()
{
    // set PORTA as output
    PORTA0_bit = 0; // clear PORTA.B0
    DDRA.F0 = 1; // set PORTA.B0 as output (rele control pin)

    // starts ENC28J60 with: reset bit on PORTB.F0, CS bit on PORTB.F1,
    // my MAC & IP address, full duplex
    SPI1_Init_Advanced(SPI_MASTER, SPI_FCY_DIV4, SPI_CLK_LO_LEADING);
    SPI_Rd_Ptr = SPI_Read; // pass SPI Read function of used SPI module
    // full duplex, CRC + MAC Unicast + MAC Broadcast filtering
    SPI_Ethernet_Init(myMacAddr, myIpAddr, SPI_Ethernet_FULLDUPLEX);

    // dhcp will not be used here, so use preconfigured addresses
    SPI_Ethernet_confNetwork(ipMask, gwIpAddr, dnsIpAddr);

    while(1) { // do forever
        SPI_Ethernet_doPacket(); // process incoming Ethernet packets
    }
}
```

NOTE: Pored proširene verzije ovog programa koji je napisan za AVR mikrokontrolere, sa našeg sajta možete preuzeti i verzije pisane za dsPIC i PIC mikrokontrolere. [www.mikroe.com/en/article/](http://www.mikroe.com/en/article/)

