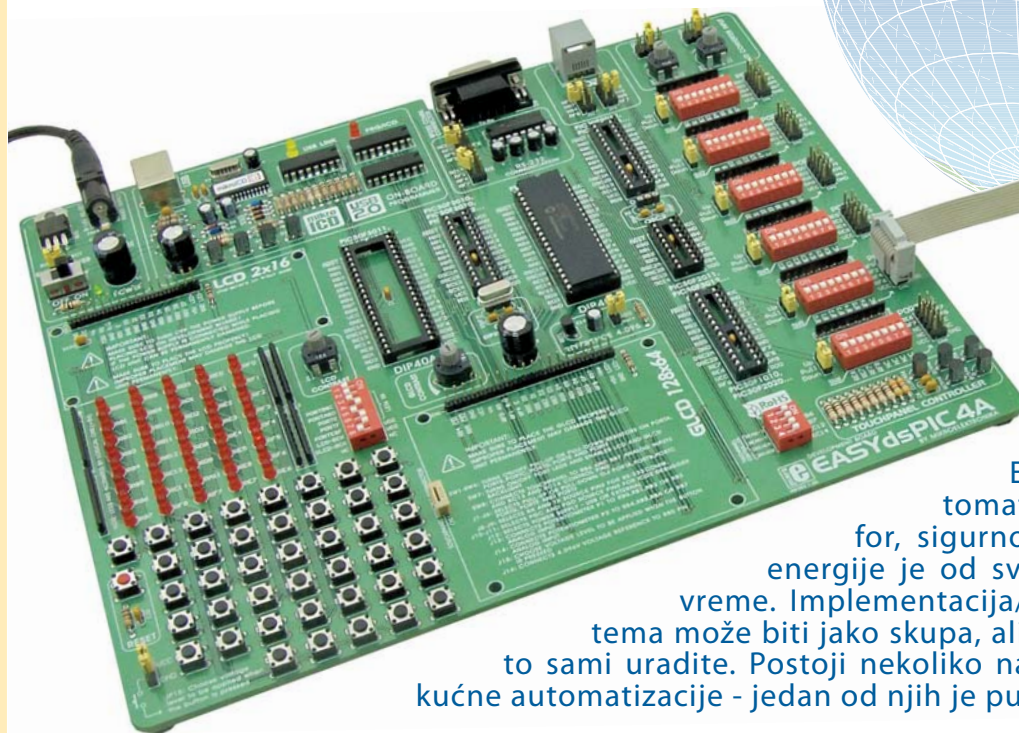


OK. Vama je potreban ... ETHERNET



Ploča serijskog eterneta povezana sa razvojnim sistemom EasydsPIC4A

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.

Srđan Tomić
MikroElektronika - Software Department

Sve što je potrebno je jedan mikrokontroler dsPIC30F4013 i serijski ethernet čip ENC28J60. Serijska komunikacija čini ovaj čip odličnim rešenjem i za mikrokontrolere iz drugih familija kao što su AVR, PIC itd. Za povezivanje na ethernet mrežu korišćen je RJ-45 konektor firme CviLux pod oznakom CJCBA8HF1Y0. LED dioda povezana na PORTD.0 mikrokontrolera predstavljaće uređaj u kući kojim želimo da upravljamo.

Kompajler *mikroC for dsPIC* 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 D mikrokontrolera. Postaviti nulu na pin PORTD.0 i setovati ga kao izlazni.

Korak 4. Inicijalizovati SPI modul mikrokontrolera dsPIC30F4013.

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.

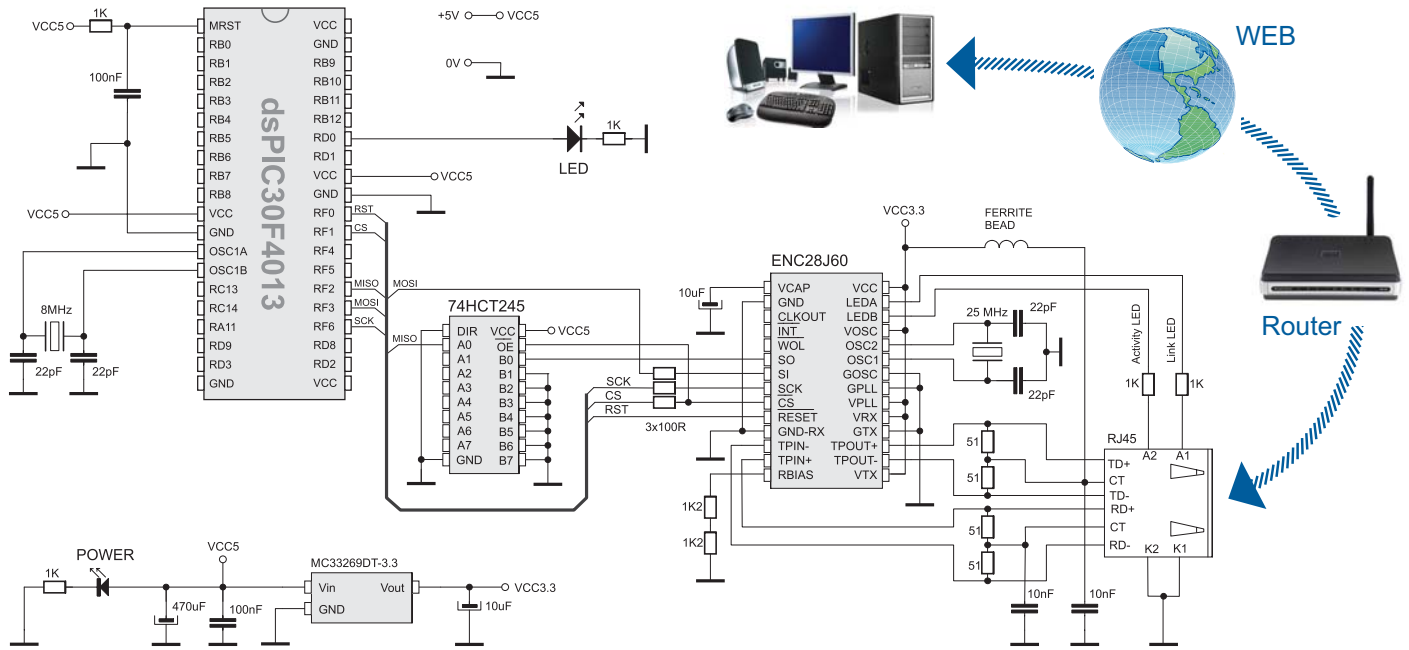
Najvažnija funkcija je Spi_Ethernet_userTCP u kojoj se odvija procesiranje svih primljenih komandi. Nakon prijema 'GET' zahteva poslatog sa vašeg računara putem web browser-a na IP adresu kontrolnog sistema, mikrokontroler će

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.



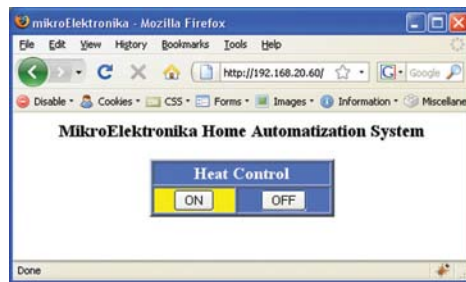
Slika 1. Pločica serijskog eterneta sa ENC28J60 čipom



Šema 1. Povezivanje pločice serijskog ethernet i mikrokontrolera dsPIC30F4013

Primer 1: Program demonstracije rada serijskog ethernet

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

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

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>\n"
"<h3 align=center>MikroElektronika Home Automation System</h3>\n"
"<form name='input' action='/' method='get'>\n"
"<td align=center colspan=2><font size=4 color=white><b>Heat Control</b></font>\n"
"</td></tr><tr><td align=center bgcolor=#4974E2><input name='tst1' width=60\n"
" type='submit' value='ON'></td><td align=center bgcolor=#FFFFFF00>\n"
"<input name='tst2' type='submit' value='OFF'></td></tr></table>\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
// end network parameters

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

unsigned int putConstString(const char *s) {
    unsigned int ctr = 0;
    while(*s) Spi_Ethernet_putByte(*s++), ctr++;
    return(ctr);
}

unsigned int putString(char *s) {
    unsigned int ctr = 0;
    while(*s) Spi_Ethernet_putByte(*s++), ctr++;
    return(ctr);
}

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

    Delay_1us();

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

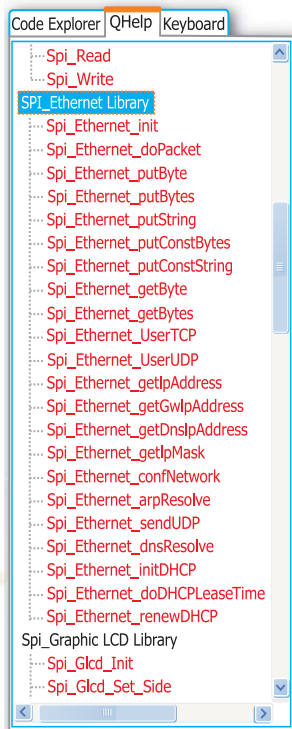
unsigned int SPI_Ethernet_UserUDP( char *remoteHost, unsigned int remotePort,
    unsigned int destPort, unsigned int reqLength)
{
    return 0; // back to the library with the length of the UDP reply
}

void main()
{
    ADPCFG |= 0xFFFF; // no analog inputs
    PORTD.F0 = 0; // set PORTD.B0 as output (rele control pin)
    TRISD.F0 = 0; // set PORTD.B0 as output (rele control pin)

    // starts ENC28J60 with: reset bit on PORTF.F0, CS bit on PORTF.F1,
    Spi_Init(); // my MAC & IP address, full duplex
    // full duplex, CRC + MAC Unicast + MAC Broadcast filtering
    Spi_Ethernet_Init(&PORTF, 0, &PORTF, 1, // HTML page first part
        myMacAddr, myIpAddr, Spi_Ethernet_FULLDUPLEX);

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

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



Spi_Ethernet_Init()	Inicijalizacija ENC28J60 kontroler
Spi_Ethernet_Enable()	Omoгуuje mrežni saobraćaj
Spi_Ethernet_Disable()	Onemogućuje mrežni saobraćaj
Spi_Ethernet_doPacket()	Obraduje dobijeni paket podataka
Spi_Ethernet_putByte()	Upisuje bajt
Spi_Ethernet_putBytes()	Upisuje zahtevane bajtove
Spi_Ethernet_putConstBytes()	Upisuje zahtevane const bajtove
Spi_Ethernet_putString()	Upisuje string
Spi_Ethernet_putConstString()	Upisuje const string
Spi_Ethernet_getByte()	Uzima bajt
Spi_Ethernet_getBytes()	Uzima zahtevane bajtove
Spi_Ethernet_UserTCP()	Obraduje TCP
Spi_Ethernet_UserUDP()	Obraduje UDP
Spi_Ethernet_getIpAdress()	Uzimanje IP adrese
Spi_Ethernet_getGwIpAdress()	Uzimanje Gateway adrese
Spi_Ethernet_getDnsIpAdress()	Uzimanje DNS adrese
Spi_Ethernet_getIpMask()	Uzimanje IP maske
Spi_Ethernet_confNetwork()	Postavlja mrežne parametre
Spi_Ethernet_arpResolve()	Šalje zahtev za ARP
Spi_Ethernet_sendUDP()	Šalje UDP
Spi_Ethernet_dnsResolve()	Šalje zahtev za DNS
Spi_Ethernet_initDHCP()	Šalje zahtev za DHCP
Spi_Ethernet_doDHCPLeaseTime()	Obrada DHCP validnosti
Spi_Ethernet_renewDHCP()	Zahtev za DHCP obnovu

Spisak dodatnih funkcija korišćenih u programu:

- Spi_Init()** Inicijalizacija modula na mikrokontroleru
- memcpy()** Kopira RAM lokacije mikrokontrolera
- memcmp()** Upoređuje RAM lokacije mikrokontrolera

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

