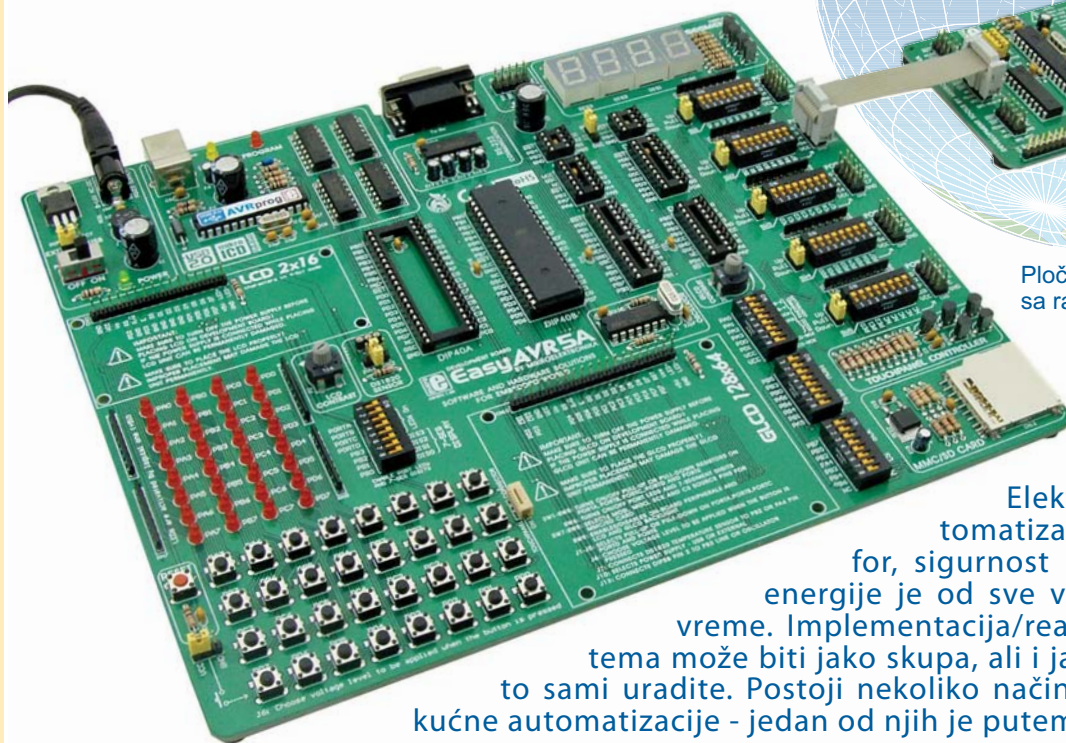


OK. Vama je potreban ... ETHERNET



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 *mikroBASIC 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.

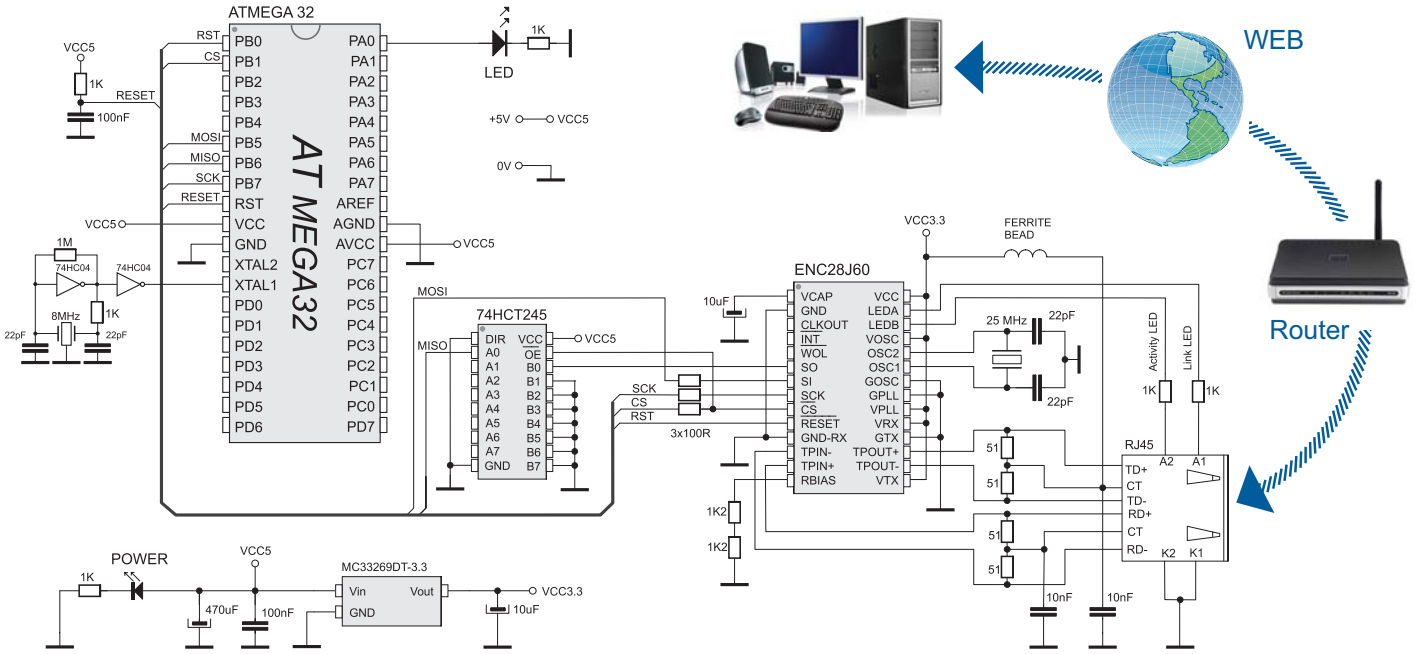
Najvažnija funkcija je `Spi_Ethernet_userTCP` u kojoj se odvija procesiranje svih primljenih komandi. Nakon prijema 'GET' zahteva poslatog sa vašeg računara putem web browser-a na IP adresu

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.

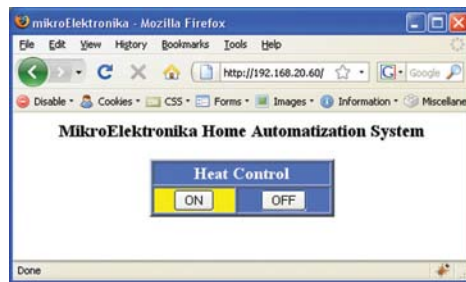


Slika 1. Pločica serijskog etneta sa ENC28J60 čipom



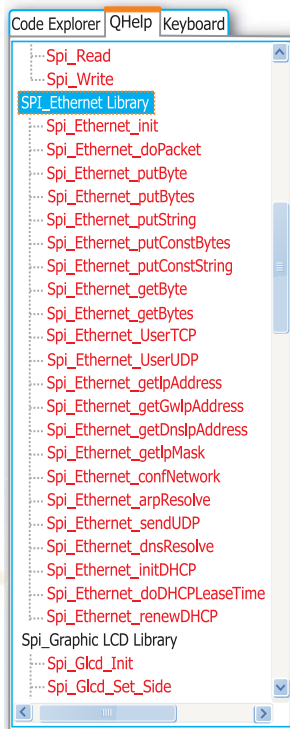
Šema 1. Povezivanje pločice serijskog etneta i mikrokontrolera Atmega 32

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 mikroBASIC for AVR.



Spi_Ethernet_Init()	Inicijalizuje 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_GetGwIpAddress()	Uzimanje IP adrese
Spi_Ethernet_GetGwIpAddress()	Uzimanje Gateway adrese
Spi_Ethernet_GetDnsIpAddress()	Uzimanje DNS adrese
Spi_Ethernet_GetGwIpMask()	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

* SPI Ethernet biblioteka korišćena u programu:
 Spisak dodatnih funkcija korišćenih u programu:
 Spi_Init() Inicijalizacija modula na mikrokontroleru
 memcpy() Kopira RAM lokacije mikrokontrolera
 memcpy() Upoređuje RAM lokacije mikrokontrolera

Primer 1: Program demonstracije rada serijskog etneta

```

program enc_ethernet
dim myMacAddr as byte(6)          ' my MAC address
myIpAddr as byte(4)              ' my IP address
gwIpAddr as byte(4)              ' gateway (router) IP address
ipMask as byte(4)                ' network mask (for example : 255.255.255.0)
dnsIpAddr as byte(4)             ' DNS server IP address
indexPage as string(523)         ' default html page
getRequest as byte(20)           ' HTTP request buffer
httpHeader as string(30)         ' HTTP header
httpMimeTypeHTML as string(13)  ' HTML MIME type
httpMimeTypeScript as string(14)' TEXT MIME type

'mE ethernet NIC pinout
SPI_Ethernet_Rst as sbit at PORTB.B0
SPI_Ethernet_CS as sbit at PORTB.B1
SPI_Ethernet_Rst_Direction as sbit at DDRB.B0
SPI_Ethernet_CS_Direction as sbit at DDRB.B1
'end ethernet NIC definitions

sub function putString(dim s as ^byte) as word
result = 0
while(s^ < 0)
  Spi_Ethernet_putByte(s^a)
  Inc(s)
  Inc(result)
wend
end sub

sub function SPI_Ethernet_UserTCP(dim byref remoteHost as byte(4),
dim remotePort, localPort, reqLength as word) as word
my reply length
dim cnt as word
tmp as string(10)

if(localPort <> 80) then          ' I listen only to web request on port 80
result = 0
exit
endif

' get 15 first bytes only of the request, the rest does not matter here
for cnt = 0 to 14 getReq[ cnt ] = SPI_Ethernet_getByte() next cnt
getReq[ cnt ] = 0

tmp = "GET/"
if(memcpy(@getReq, @tmp, 5) > 0) then          ' only GET method
result = 0
exit
endif

tmp = "ON"
if(memcpy(@getReq+11, @tmp, 2) = 0) then       ' do we have ON command
PORTA.B0 = 1
else
tmp = "OFF"
if(memcpy(@getReq+11, @tmp, 3) = 0) then       ' do we have OFF command
clear PORTA.B0 = 0
endif
endif

if (PORTA.B0) then
tmp = "#FFFF00" memcpy(@indexPage+340, @tmp, 6) ' highlight (yellow) ON
tmp = "#4974E2" memcpy(@indexPage+431, @tmp, 6) ' clear OFF
else
tmp = "#4974E2" memcpy(@indexPage+340, @tmp, 6) ' clear ON
tmp = "#FFFF00" memcpy(@indexPage+431, @tmp, 6) ' highlight (yellow) OFF
endif
cnt = putString(@httpHeader)                  ' HTTP header
cnt = cnt + putString(@httpMimeTypeHTML)     ' with HTML MIME type
cnt = cnt + putString(@indexPage)            ' HTML page first part
result = cnt                                  ' return to the library with the number of bytes to transmit
end sub

sub function SPI_Ethernet_UserUDP(dim byref remoteHost as byte(4),
dim remotePort, destPort, reqLength as word) as word
result = 0
' back to the library with the length of the UDP reply
end sub

main:
PORTA.0_bit = 0                          ' set PORTA as output
DDRA.B0 = 1                              ' clear PORTA.B0
' set PORTA.B0 as output (rele control pin)

httpHeader = "HTTP/1.1 200 OK" + chr(10) + "Content-type:"
httpMimeTypeHTML = "text/html" + chr(10) + chr(10)
httpMimeTypeScript = "text/plain" + chr(10) + chr(10)
indexPage =
"<html><head><title>mikroElektronika</title></head><body>"+
"<h3 align=center>MikroElektronika Home Automation System</h3>"+
"<form name="+chr(34)+"input"+chr(34)+" action="+chr(34)+"?"+chr(34)+" method="+
chr(34)+" get"+chr(34)+"><table align=center width=200 bgcolor=#4974E2 border=2><tr>"+
"<td align=center colspan=2><font size=4 color=white>cb-Heat Control</td><td align=center bgcolor=#4974E2><input name="+chr(34)+"tst1"+
chr(34)+" width=60 type="+chr(34)+" submit="+chr(34)+" value="+chr(34)+" ON"+
chr(34)+"></td><td align=center bgcolor=#FFFF00><input name="+chr(34)+"tst2"+
chr(34)+" type="+chr(34)+" submit="+chr(34)+" value="+chr(34)+" OFF"+chr(34)+"></td></tr></table></form></body></html>"

myMacAddr[0] = 0x00 myMacAddr[1] = 0x14 myMacAddr[2] = 0xA5
myMacAddr[3] = 0x76 myMacAddr[4] = 0x19 myMacAddr[5] = 0x3F
ipMask[0] = 255 ipMask[1] = 255 ipMask[2] = 255 ipMask[3] = 0
gwIpAddr[0] = 192 gwIpAddr[1] = 168 gwIpAddr[2] = 20 gwIpAddr[3] = 60
gwIpAddr[0] = 192 gwIpAddr[1] = 168 gwIpAddr[2] = 20 gwIpAddr[3] = 6
dnsIpAddr[0] = 192 dnsIpAddr[1] = 168 dnsIpAddr[2] = 20 dnsIpAddr[3] = 1

' starts ENC28J60 with: reset bit on PORTB.F0, CS bit on PORTB.F1,
my MAC & IP address, full duplex
SPI_Init Advanced_SPI_MASTER, SPI_FCY_DIV4, SPI_CLK_LO_LEADING)
SPI_Read Ptr = @SPI_Read
SPI_Ethernet_UserTCP_Ptr = @SPI_Ethernet_UserTCP
SPI_Ethernet_UserUDP_Ptr = @SPI_Ethernet_UserUDP
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 true
  SPI_Ethernet_DoPacket()
wend
end.

```

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/

