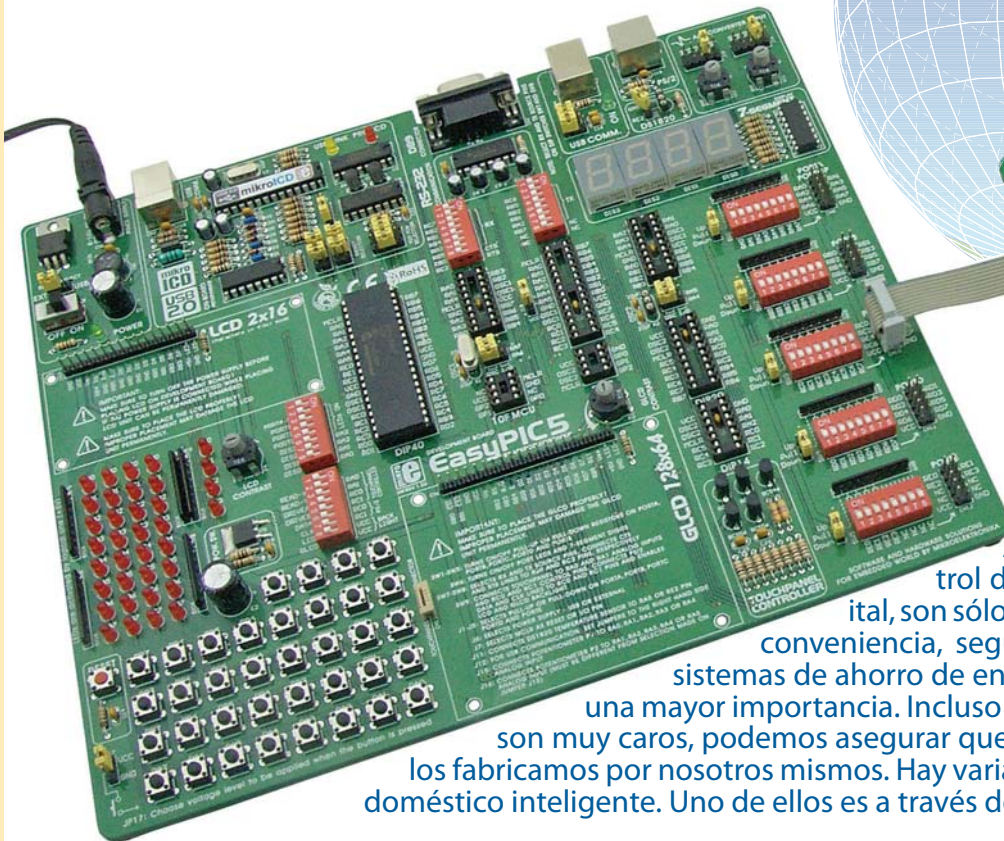


Si. ETHERNET

Se necesita...



Módulo Serie Ethernet conectado al sistema de desarrollo EasyPIC5

Automatización doméstica, control doméstico, casa inteligente o digital, son sólo diferentes nombres para confort, conveniencia, seguridad y ahorro de energía. Los sistemas de ahorro de energía están adquiriendo, hoy día, una mayor importancia. Incluso si pensamos que dichos sistemas son muy caros, podemos asegurar que también son bastante baratos si los fabricamos por nosotros mismos. Hay varias formas de controlar un sistema doméstico inteligente. Uno de ellos es a través de Ethernet.

Por Srdjan Tomic
MikroElektronika – Departamento de Software

Todo lo que necesitamos es un microcontrolador PIC18F4520 y un circuito integrado Ethernet serie ENC28J60. Este circuito integrado es una gran solución para otras familias de microcontroladores, tales como AVR, dsPIC etc. El conector RJ-45 CJCBA8HF1Y0 de Cvilux se usa para la conexión a la red Ethernet. Un diodo LED conectado al PORTB.0 del microcontrolador, simulan una aplicación doméstica que quiere el control.

El compilador *mikroBASIC for PIC* contiene la librería `SPI_Ethernet` que simplificará considerablemente el proceso de escritura de un programa para el microcontrolador. Usando unas pocas rutinas de esta librería, es posible crear el programa que nos permitirá controlar aplicaciones eléctricas en nuestra casa a través de un explorador web.

Para ello, es necesario realizar las siguientes operaciones dentro del programa:

Paso 1. Crear una página html a través de la cual arrancar el microcontrolador. Importar el código como un bloque de texto ("string").

Paso 2. Configurar las direcciones IP, DNS, Gateway y máscaras de Subred proporcionadas por nuestro proveedor de Internet.

Por ejemplo, nuestros parámetros locales de red son los siguientes:

IP: 192.168.20.60 (dirección del Sistema de Control)

DNS: 192.168.20.1 (dirección del Domain Name System o Sistema de Nombres de Dominio)

GATEWAY: 192.168.20.6 (dirección de la pasarela o Gateway)

SUBNET: 255.255.255.0 (máscara de Subred)

Paso 3. Deshabilitar las entradas analógicas de PORTB. El terminal del microcontrolador debe ser borrado y configurado como una salida.

Paso 4. Inicializar el módulo SPI del microcontrolador PIC18F4520.

Paso 5. Inicializar el módulo Serie Ethernet del circuito integrado ENC28J60.

Paso 6. Escribir el código dentro de la función `SPI_Ethernet_userTCP` que, después de recibir el comando a través del explorador web, encenderá/apagará el diodo LED conectado al PORTB.0.

Paso 7. Leer los datos recibidos en un bucle sin fin.

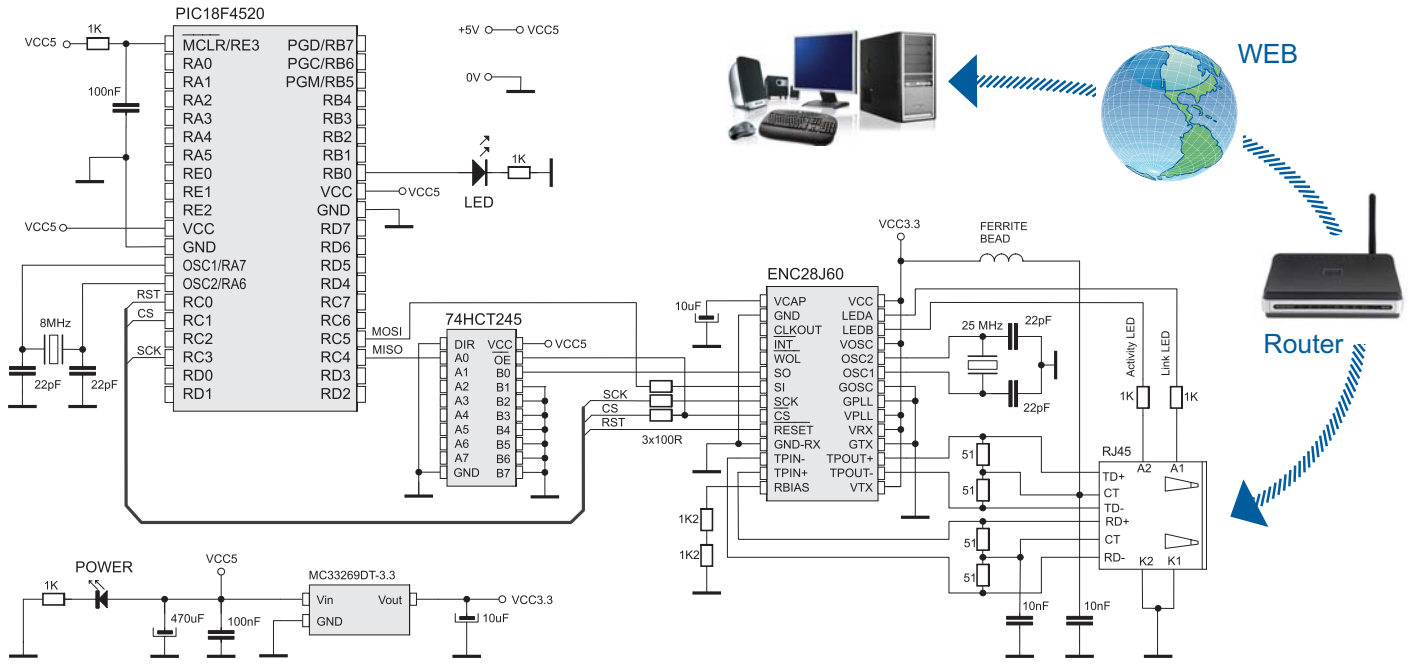
La parte más importante del programa es la función `SPI_Ethernet_userTCP`, que procesa todos los comandos recibidos.

Después de recibir la petición "GET" del navegador web, enviada desde nuestro ordenador a la dirección IP del sistema de control, el microcontrolador responderá con una página web almacenada en su memoria. Esta página será mostrada automáticamente en la pantalla del ordenador por el navegador web. Cuando se recibe el comando ON, el diodo LED conectado a PORTB.0 se encenderá.

Del mismo modo, cuando se recibe el comando OFF, el diodo LED se apaga. Si en lugar de un diodo LED tenemos un relé, es posible controlar cualquier aplicación como una lámpara, un sistema de seguridad, un sistema de calefacción, etc.

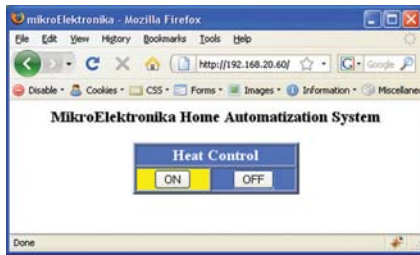


Figura 1. MikroElektronika Módulo Serie Ethernet con ENC28J60



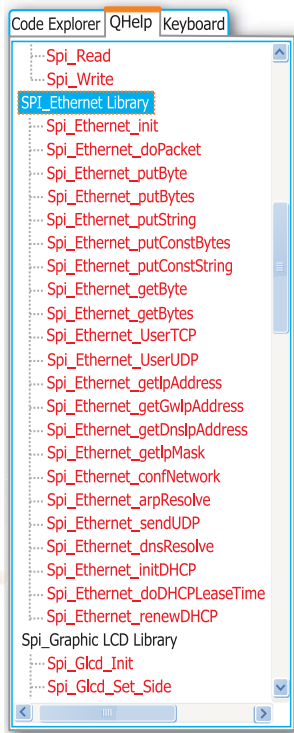
Esquema eléctrico 1. Conexión del módulo Serie Ethernet al PIC18F4520

El control de cualquier aplicación doméstica consiste en la introducción de la dirección IP del sistema de control en el navegador web y especificar los comandos deseados. Por supuesto, es posible controlar más de un terminal del microcontrolador, lo cual nos permite gobernar un gran número de aplicaciones o un sistema de automatización doméstico completo.



La captura de pantalla ilustra la página web mostrada por el navegador después de introducir la dirección IP del sistema de control. En nuestro ejemplo, al pulsar sobre los botones ON y OFF provocaremos que el diodo LED se encienda y se apague, simulando el control de un sistema de calefacción.

Mas abajo está la lista de las funciones, ya creadas, contenidas en la librería SPI Ethernet Library. Esta librería esta integrada en el compilador mikroBASIC for PIC.



| | |
|---------------------------------------|--------------------------------------|
| Spi_Ethernet_Init() | Inicia el controlador ENC28J60 |
| Spi_Ethernet_Enable() | Habilita de tráfico de la red |
| Spi_Ethernet_Disable() | Deshabilitar el tráfico de la red |
| Spi_Ethernet_doPacket() | Paquetes de procesos recibido |
| Spi_Ethernet_putByte() | Almacena un byte |
| Spi_Ethernet_putBytes() | Almacena bytes |
| Spi_Ethernet_putConstBytes() | Almacena bytes como constantes |
| Spi_Ethernet_putString() | Almacena string |
| Spi_Ethernet_putConstString() | Almacena string como constante |
| Spi_Ethernet_getByte() | Obtener un byte |
| Spi_Ethernet_getBytes() | Obtener bytes |
| Spi_Ethernet_UserTCP() | Código de manejo TCP |
| Spi_Ethernet_UserUDP() | Código de manejo UDP |
| Spi_Ethernet_getIpAddress() | Obtiene dirección IP |
| Spi_Ethernet_getGwIpAddress() | Obtiene dirección Gateway o pasarela |
| Spi_Ethernet_getDnsIpAddress() | Obtiene dirección DNS |
| Spi_Ethernet_getIpMask() | Obtiene máscara de dirección IP |
| Spi_Ethernet_confNetwork() | Establece los parámetros de red |
| Spi_Ethernet_arpResolve() | Envía una petición ARP |
| Spi_Ethernet_sendUDP() | Envía un paquete UDP |
| Spi_Ethernet_dnsResolve() | Envía una petición DNS |
| Spi_Ethernet_initDHCP() | Envía una petición DHCP |
| Spi_Ethernet_doDHCPLeaseTime() | Tiempo de proceso |
| Spi_Ethernet_renewDHCP() | Petición de renovar DHCP |

Otras funciones mikroBASIC for PIC usadas en el programa:
memcpy() Copia posiciones de la memoria RAM del microcontrolador
memset() Copia posiciones de la memoria RAM del microcontrolador

Ejemplo 1: Programa que demuestra el control a través de Ethernet (two files)

```

program home_auto
include "home_auto_utils"

dim myMacAddr as byte[6]
myIpAddr as byte[4]
gwIpAddr as byte[4]
ipMask as byte[4]
dnsIpAddr as byte[4]

main:
ADCON1 = 0x0F
CMCON = 0x07

PORTB.0 = 0
TRISB.0 = 0

indexPage =
"<html><head><title>mikroElektronika</title></head><body>"+
"<h3 align=center>MikroElektronika Home Automatization System</h3>"+
"<form name="+Chr(34)+"input"+Chr(34)+" action="+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><b>Heat Control</b></font>"+
"</td></tr><tr><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
myIpAddr[0]=192 myIpAddr[1]=168 myIpAddr[2]=20 myIpAddr[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 PORTC.F0, CS bit on PORTC.F1,
myMac & IP address, full duplex
Spi_Init()
' full duplex, CRC + MAC Unicast + MAC Broadcast filtering
Spi_Ethernet_Init(PORTC.0, PORTC.1,
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

module home_auto_utils
const httpHeader as string[31] = "HTTP/1.1 200 OK"+chr(10)+"Content-type:"
const httpMimeTypeHTML as string[13] = "text/html"+chr(10)+"chr(10)
const httpMimeTypeScript as string[14] = "text/plain"+chr(10)+"chr(10)

' default html page
dim indexPage as string[523]
dim getRequest as byte[20] ' HTTP request buffer
implements
sub function Spi_Ethernet_UserTCP(dim byref remoteHost as byte[4],
dim remotePort, localPort, reqlength as word) as word
dim cnt as word
tmp as string[10]
' my reply length

if(localPort < 80) then
' I listen only to web request on port 80
exit
end if

' get 10 first bytes only of the request, the rest does not matter here
for cnt = 0 to 14
getRequest[cnt] = Spi_Ethernet_getByte()
next cnt

tmp = "GET "
if(memcmp@getRequest, @tmp, 5) < 0 then
' only GET method
result = 0
exit
end if

tmp = "ON"
if(memcmp@getRequest+11, @tmp, 2) = 0 then
' do we have ON command
PORTB.0 = 1
set PORTB bit 0
else
tmp = "OFF"
if(memcmp@getRequest+11, @tmp, 3) = 0 then
' do we have OFF command
PORTB.0 = 0
clear PORTB bit 0
end if
end if

if(PORTB.0) 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
end if

cnt = Spi_Ethernet_putConstString(@httpHeader) ' HTTP header
cnt = cnt + Spi_Ethernet_putConstString(@httpMimeTypeHTML) ' HTML MIME type
cnt = cnt + Spi_Ethernet_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
end

```

NOTA: El código para este ejemplo ha sido escrito para microcontroladores PIC® en lenguaje C, Basic y Pascal, del mismo modo que los programas han sido escritos para microcontroladores dsPIC® y AVR®. Todo ello lo pueden encontrar en nuestra página web: www.mikroe.com/en/article/

