# David and Jacqui's Wiki

electronics:micromite_wiznet_library

# Micromite Wiznet Library

## Hardware Connections

| Wiznet | Micromite | Notes |
|---|---|---|
| Ground | Pin 19 | |
| +5V | +5V | |
| RST | Pin 23 | Wiznet W5100 Reset |
| SS | Pin 15 | SPI Slave Select |
| CLK | Pin 25 | SPI Clock |
| MO | Pin 3 | SPI MOSI - Master Out, Slave In |
| MI | Pin 14 | SPI MISO - Master In, Slave Out |
| Gnd | | not used |
| PoE+ | | not used |
| PoE- | | not used |

## Required Code

Note: The wiznet module may not work when connected to a PoE switch - symptoms seen: gets Power LED but never establishes an ethernet connection to the Local Area Network (LAN).

The following code snippet (plus the library) is the minimum you require to get a ping response from the Micromite with Wiznet:

```
Option base 1

CONST optionBase=1       ' Set to match option base above (if changed)
CONST wiznetResetPin%=23 ' Set to pin used for SPI Slave Select of Wiznet

dim ip1%, ip2%, ip3%, ip4%

wiznetReset wiznetResetPin%

' Set static MAC address/IP address/subnet mask/gateway here

ip1%=192
ip2%=168
ip3%=1
ip4%=65
SetMAC &H00,&HFE,ip1%,ip2%,ip3%,ip4%
SetIP ip1%,ip2%,ip3%,ip4%
SetMask 255,255,255,0
SetGateway ip1%,ip2%,ip3%,1
```

```
wiznetInitialize    ' Sets various internal registers for Wiznet
```

## Notes:

1. If you change to “Option base 0” (which starts array numbering at 0, not 1), set “CONST optionBase=0” also or all SPI receives will fail.
2. You need to make sure the MAC address is unique to every host on your local network. This sample code sets the MAC address to &H00FEwwxxyyzz where ww.xx.yy.zz represent the IP Address. That way the code will work across multiple devices where you change only the IP address by setting ip1%, ip2%, ip3% and ip4%. If you produce a commercial product, you will need to apply for your own MAC address allocation.

## When you can ping your Micromite

From here, up to four sockets (0, 1, 2, and 3) can be initialized and used for either TCP, UDP, IP-Raw or MAC-Raw (only TCP implemented in Micromite library at present) concurrently. In other words, you could have a webserver “Listening” on one TCP server socket on the Micromite that gives users access to a small web page showing analog pin values or temperature/humidity data. You could have another socket establishing a TCP client socket connection to a remote webserver data logging your temperature/humidity every 60 seconds.

The Micromite library allocates 2048 bytes to each send and 2048 bytes to each receive buffer on the Wiznet module (in wiznetInitialize subroutine).

### Sockets

| Library Support | Protocol | Title | Notes |
|---|---|---|---|
| Yes | TCP | Transmission Control Protocol | Guaranteed delivery, eg: Telnet to a server; HTTP - accessing (client) or hosting (server) web sites; etc. |
| No | UDP | User Datagram Protocol | Very basic IP protocol - can send and receive packets but no guarantee they will go through or will arrive in order. Great if you want a simple one-packet reply quickly or for data logging applications where it doesn't matter if you miss a packet every so often. |
| No | IP-Raw | Raw TCP/IP Packet | Write your own IP sub-protocol! This is what you'd use to implement DHCP |
| No | MAC-Raw | Raw MAC level Packet | Write your own ethernet protocol! You'd need to come up with a routing protocol to get past your local network. Unlikely anybody would ever want to use this in a Micromite |

When initialized, the Wiznet module sets all four sockets to “Closed”. From there, they can be “Initialized” to TCP or “Opened” to UDP, IP Raw or MAC Raw. When a socket is “Open”, data can be sent and received.

### TCP server

```
'If socket closed, start listening for incoming connections
if (SockClosed(0)=1) then
  if (SockTCPInit(0, 23)=1) then
    print:print:print "Socket initialized on local port 23"
```

```
  else
    print:print:error "Failed to initialize socket"
  end if
```

#### SockClosed Function

SockClosed(<socket number 0-4>)

Check Socket Closed Status

Call with the socket number.

Returns 1 if socket is Closed or -1 if socket is Open.

#### SockTCPInit Function

SockTCPInit(<socket number 0-4>, <local port number 0-65535>)

Initialize Socket

Call with the socket number and local port number to use. Port 23 is used for Telnet, port 80 for HTTP, etc. Here are some more:

https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml [https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml]

You can use any port for testing, but typically keep them to below 1024 for TCP servers.

Returns 1 if socket inialized or -1 if socket is socket fails to initialize.

```
  if (SockTCPListen(0)=1) then
    print "Socket listening for incoming connections - telnet to IP address"
  else
    error "Failed to listen on socket"
  end if
```

#### SockTCPListen Function

TCP Socket Listen

SockTCPListen(<socket number 0-4>)

Call with the socket number.

Returns 1 if socket is Listening or -1 if socket fails (Note: if the Listen fails, you need to reinitialize the Socket with SockTCPInit function before retrying the SockTCPListen).

```
do while sockData(0)
  print chr$(sockRead(0));
loop

'Send some data to the client
if SockEstablished(0)=1 then
  txstring$="0123456789"
  SockWrite 0, txstring$
end if
```

#### sockData Function

sockData(<socket number 0-4>)

Check if there is data available in a socket buffer on the Wiznet module.

Call with the socket number.

Returns the number of bytes available in the receive buffer for the socket. There may be data remaining in a socket even after a TCP connection is “Established”.

#### sockRead Function

sockRead(<socket number 0-4>)

Read next byte from socket buffer on the Wiznet module.

Call with the socket number.

Returns the next data byte as an integer from the buffer for the socket. Do not call unless there is data waiting or corruption will occur - check sockData() function first.

#### SockEstablished Function

SockEstablished(<socket number 0-4>)

Checks if a TCP socket is established - if it is, data can be sent. This function also finalises a disconnection if the remote party has requested disconnection, so good practice dictates that it should be called in a loop while connections are established.

Call with the socket number.

Returns 1 if TCP socket established or -1 if socket is not established (in other words, 1 if there is a current connection to another device for this socket).

#### SockWrite Subroutine

SockWrite <socket number 0-4>, <string to send>

Send data over an established socket.

Call with socket number and a string of characters to send.

#### To do

Add Function to disconnect established TCP connection.

### TCP client

```
if SockTCPConnect(0,192,168,1,14,80)=1 then ' http://192.168.1.14
  print "Successful connection to webserver on port 80"
else
  error "Failed to connect to webserver on port 80"
end if
```

#### SockTCPConnect Function

SockTCPConnect(<socket number 0-4>, <Remote IP Octet 1>, <Remote IP Octet 1>, <Remote IP Octet

1>, <Remote IP Octet 1>, <Remote Port 0-65535>)

Attempt to establish a TCP connection to a remote host. Function may take up to 4 seconds to return as it waits for a connection.

Call with the socket number, the remote IP address octets and the remote port number to connect to (eg. port 80 for a standard webserver running HTTP). Note: If using client connection, initialize the socket with a high source port number, say between 63999 and 65536, eg. SockTCPInit(0, 64000).

Returns 1 if TCP client connection successful or -1 if TCP client connection fails.

```
do while sockData(0)
  print chr$(sockRead(0));
loop
```

Receiving data is the same as TCP server - refer to SockData() and SockRead() functions above.

```
SockWrite 0, "GET / HTTP/1.1"+chr$(&H0D)+CHR$(&H0A)
SockWrite 0, "Host: 192.168.1.14"+chr$(&H0D)+CHR$(&H0A)
SockWrite 0, "User-Agent: Micromite!"+chr$(&H0D)+CHR$(&H0A)
SockWrite 0, "Connection: close"+chr$(&H0D)+CHR$(&H0A)
SockWrite 0, chr$(&H0D)+CHR$(&H0A)
```

Similarly, sending data is the same as TCP server - refer to SockEstablished(0) function and SockWrite() subroutine above.

# Links

Discussion (The Back Shed Forum):

http://www.thebackshed.com/forum/forum_posts.asp?TID=7727&PN=1 [http://www.thebackshed.com/forum/forum_posts.asp?TID=7727&PN=1]

Datasheet for Wiznet module:

https://www.sparkfun.com/datasheets/DevTools/Arduino/W5100_Datasheet_v1_1_6.pdf [https://www.sparkfun.com/datasheets/DevTools/Arduino/W5100_Datasheet_v1_1_6.pdf]

eBay link for module:

http://www.ebay.com.au/itm/Durable-TOP-Mini-W5100-Ethernet-Shield-Network-Expansion-Module-for-Arduino-/121594616164 [http://www.ebay.com.au/itm/Durable-TOP-Mini-W5100-Ethernet-Shield-Network-Expansion-Module-for-Arduino-/121594616164]

# To do

1. SockEstablished needs to return -1 if DISCON has been processed - at the moment it doesn't allocate a return value
2. Add the ability to disconnect a TCP socket connection
3. Add support for UDP packets
4. Possibly add string read function to read more than one byte at a time for better performance. Problem is the string variable only has 255 characters in the Micromite and the buffer may be much larger (up to 2048 bytes). One line at a time won't work either - a line may well be longer than 255 characters.

electronics/micromite_wiznet_library.txt · Last modified: 2015/06/08 07:18 by david