Using Visual Basic (VB) to communicate with the DS100 and EM100

Please, register at www.tibbo.com to receive update notifications

Version: 2.0a

Table of Contents

1. Introduction 3
   - 1.1. What is WinSock? 3
   - 1.2. Required tools 3
2. Preparing the DS100/EM100SK 4
   - 2.1. Connecting cables 4
   - 2.2. Choosing an IP-address 5
   - 2.3. Installing the *HyperTerminal* 7
   - 2.4. Setting up the *HyperTerminal* 8
   - 2.5. Setting up the DS100/EM100SK 9
   - 2.6. Testing the LAN link 10
3. Creating the *WinSock Demo* 12
   - *3.1.* *WinSock Demo* action plan 12
   - 3.2. Step 1: creating a simple UDP/IP *Demo* 12
   - 3.3. Step 2: learning to control the DS100/EM100 over the network 15
   - 3.4. Step 3: TCP/IP communications 18

1. Introduction

1.1. What is WinSock?

WinSock ActiveX control is a part of a standard *Visual Basic 6 (VB6)* distribution. *WinSock* dramatically simplifies the development of TCP/IP-based applications. Since the **DS100 Serial Device Server** and the **EM100 Ethernet Module** from Tibbo Technology support standard UDP and TCP protocols the *WinSock* can also be used to build applications that communicate with and control the DS100 and EM100.

This *Tutorial* provides step-by-step instructions on using *VB* to create three rudimentary programs (called "*WinSock Demos*") that communicate with and program the DS100/EM100 over the network. The *Demos* are not a complete fully-functional programs but rather a "templates" that show you the basic steps of building your own applications with the *WinSock*. Complete *VB* projects for each *WinSock Demo* (packed into a ZIP archives and named "*WinSock Demo Step1*", "*WinSock Demo Step2*", and "*WinSock Demo Step3*".) can be downloaded from www.tibbo.com/downloads.htm.

1.2. Required tools

- DS100 Serial Device Server or EM100SK Starter Kit
- WAS-1455 or similar serial cable to connect the DS100/EM100 to the PC*
- WAS-1498 or WAS-1499 Ethernet cable may be needed to connect the DS100/EM100 to the LAN*
- Power adaptor (220V or 120V depending on your locality)*
- *Windows* PC with one unused Serial port and an Ethernet port
- *Visual Basic 6.0* from *Microsoft*
- *HyperTerminal* for *Windows* (part of a standard *Windows* distribution)

**These accessories are supplied with the DS100-KIT and EM100SK Products.*

2. Preparing the DS100/EM100SK

2.1. Connecting cables

To test out the *WinSock Demos* you will need to connect the Ethernet port of the DS100/EM100SK to the LAN (to which your test PC is also connected), and the RS232 port- to the PC's COM port. Our *WinSock Demos* will communicate with the DS100/EM100SK directly via the Ethernet (which is a TCP/IP network). RS232 side of the link will be served by the *HyperTerminal*- a standard *Windows* software used to send and receive the serial data.

Assembly the network as follows (*Fig. 1*):

- Connect the RS232 port of the DS100/EM100SK to the unused Serial port of your PC with the WAS-1455 serial cable
- Connect the Ethernet port of the DS100/EM100SK to your office LAN or directly to your PC using the WAS-1499 or WAS-1498 cable (see below for more information)
- Power the DS100/EM100SK up using the power adaptor.

There are two ways to connect the Ethernet side of the DS100/EM100SK to the PC:

- **Via LAN**. If you have a LAN installed in your office and your PC is already connected to this LAN, then you will need a spare cable (coming out of the network hub) to connect the DS100/EM100SK. If no spare cable is readily available, then our WAS-1499 Device-to-Hub cable can be used. Plug one side of the cable into the unused 10BaseT connector on your hub, another one- into the DS100/EM100SK Ethernet port
- **Directly**. You can bypass the LAN and the Hub using our WAS-1498 Device-to-Device cable. Plug one side of the cable into your PC's Ethernet port (yes, you need to disconnect your PC from the office network first), another side- into the Ethernet port on the DS100/EM100SK.

Note: The Ethernet port of the DS100/EM100SK is of 10BaseT type, not 100BaseT. You can use the DS100/EM100SK on “10” or “Auto 10/100” LANs, but not on “100” LANs.

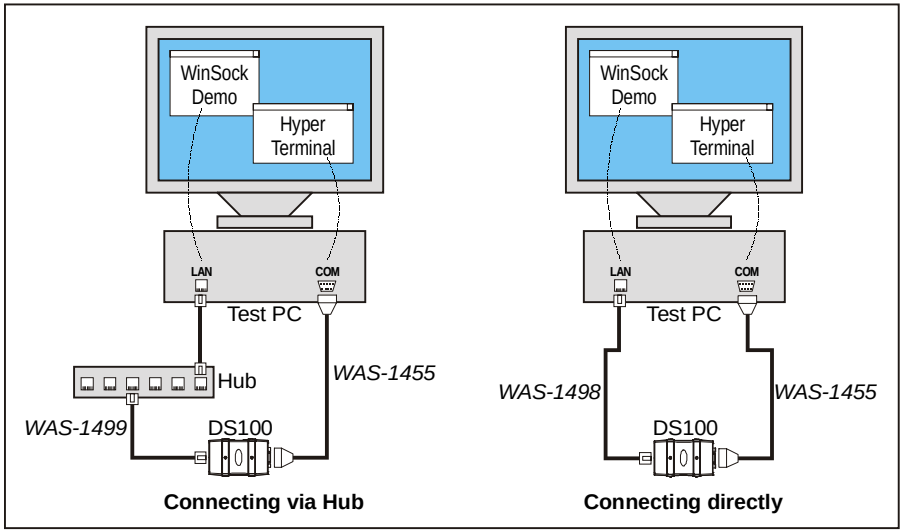

Fig. 1. Test arrangement

2.2. Choosing an IP-address

Each DS100/EM100 you use must be assigned a network-unique IP-address. Most LANs have restrictions on the range of permitted IP-addresses, so you cannot just use any random address. One reliable way to obtain a “good” IP-address is to ask your IT guy ☺. If there is no such person in your building then you can follow this (many times proven) method:

- Use our WAS-1498 cable to directly connect your PC to the DS100/EM100SK. This way you will disconnect from the rest of the LAN (a lot of possible problems will be cut out…)
- Use the *WINIPCFG* program to find out the IP-address of your PC (see explanation below). In 99% of cases, choosing the address for the DS100/EM100SK that differs slightly from your PC's IP-

- From a drop-down box at the top of the dialog select a network adaptor that is used to connect your PC to the Ethernet network. All available adaptors are listed by their actual model number. Usually there is only one Ethernet adaptor in the PC so you need to find some line with the word “Ethernet” in it. For example if the drop-down box contains two entries: “PPP Adapter” and “D-Link DFE-530TX PCI Fast Ethernet Adapter” then you know which one you need to choose, right ? (just in case, it is the second one).
- After you have selected correct adapter note an *IP-address* displayed below and click *OK*- the dialog will close
- Choose an IP-address that is close to the one set for your PC. Procedures below assume that this new IP-address is 192.168.100.40.

2.3. Installing the *HyperTerminal*

One side of the test link we are building is served by (as yet to be created) a *WinSock Demo* program, another one- by a standard serial communications software called *HyperTerminal*. This program is also used here to setup the DS100/EM100SK. Actually, there is a setup software called *DS Manager* (part of *Tibbo Device Server Toolkit* available for download from www.tibbo.com/downloads.htm) that can be used to setup the DS100/EM100SK. However, we are deliberately choosing to make you type all the setup commands manually. This way you will familiarize yourself with the structure and use of actual commands sent to the DS100/EM100SK. Later in this Tutorial we will show you how to setup the DS100/EM100SK through the network so the knowledge of programming commands will be quite handy!

The *HyperTerminal* is normally found in the *Start→ Programs→ Accessories→ Communications→ HyperTerminal* folder. If it is not there, then you must have opted it out when installing *Windows* on your PC. Follow the instructions below to add *HyperTerminal* to your system (be sure to have your *Windows* distribution CD handy!):

- Go to the *Control Panel (Start→ Settings→ Control Panel)* and double-click on the *Add/Remove Programs* icon- the *Add/Remove Programs* dialog will open

- Click on *Windows Setup* tab to view the list of optional installation components
- Choose *Communications* in the *Components* list and click *Details*
- In the *Communications* window, select the *HyperTerminal* (it must be "checked")
- Press *OK* to close *Communications* window, press *OK* again to close *Add/Remove Programs*
- You will possibly be asked to insert the *Windows* CD at this point. Do this and follow the instructions on the screen.

2.4. Setting up the *HyperTerminal*

Once the *HyperTerminal* is installed (or found ☺), launch it and follow the procedure below:

- When the *Connection Description* dialog opens, type any descriptive string (like "DS100") and press OK
- When the *Connect to* dialog opens, select an appropriate COM port from the *Connect Using* drop-down box (for example, "*Direct to COM1*")
- When the *COM Properties* dialog appears, set communications parameters as follows: *Bits per second*: 38400, *Data bits*: 8, *Parity*: None, *Stop bits*: 1, *Flow control*: None. Click *OK* when done- the *HyperTerminal's* main window will appear
- Choose *File→Properties* from the *Main* menu- the *Properties* dialog will open
- Click on the *Settings* tab and press the *ASCII Setup* button- the *ASCII Setup* dialog will open
- "Check" (enable) three options: *Echo typed characters locally*, *Send line feeds with line ends*, and *Append line feeds to incoming line ends*
- Click *OK* twice to close both dialogs
- Optional (but strongly recommended): you may want to *Save* (*File→Save* in *Main* menu) this *HyperTerminal* configuration so that you don't have to set all the parameters again in the future!

After the initialization the DS100 is set to work in the Slave Routing Mode, with the UDP/IP communications protocol. The data exchange is to be effected with Data Port #1001.

OK, now disconnect the DS100's power and then connect it again- the unit will resume operation with the new functioning parameters.

Do not exit the *HyperTerminal* at this time- you will need it to exchange the data with the *WinSock Demo*.

2.6. Testing the LAN link

Final step before we start creating the *Demo* is to make sure that your PC can "find" the DS100/EM100SK on the network. Use *PING* program to check this:

- Choose *Run…* from the *Start* menu of *Windows*- the *Run* dialog will open
- Type *ping 192.168.100.40* into the *Open* textbox and press OK
- The ping program will send four messages to the DS100/EM100SK. If there is no reply from the DS100/EM100SK then the program output will look like this:

```
Pinging 192.168.100.40 with 32 bytes of data
Request timed out.
Request timed out.
Request timed out.
Request timed out.
```

- If the DS100/EM100SK does reply to your PC's "pinging" then the output from the ping program will look somewhat like this:

```
Pinging 192.168.100.40 with 32 bytes of data
Reply from 192.168.100.40: bytes=32, time=9ms, TTL=255
Reply from 192.168.100.40: bytes=32, time=9ms, TTL=255
Reply from 192.168.100.40: bytes=32, time=9ms, TTL=255
```

```
Reply from 192.168.100.40: bytes=32, time=9ms, TTL=255
```

Note: “time” and “TTL” parameters are not important. The only important point is that the DS100/EM100SK does, indeed, reply to your PC’s “pinging”.

Obtaining a “good ping” from the DS100/EM100SK is a required step in setting up your test network. Nothing will work as expected unless your PC can find the DS100/EM100SK by “pinging” it.

3. Creating the *WinSock Demo*

3.1. *WinSock Demo* action plan

We will show you the basics of communicating with the DS100/EM100SK in three steps:

- **In step 1** we will create a very simple program that will be able to exchange the data with the Data Port of the DS100/EM100SK using an UDP/IP protocol
- **In step 2** we will show how to program (setup) the DS100/EM100 *over the network*. Same UDP/IP protocol will be used, but this time to communicate with the Command Port of the DS100/EM100SK
- **In step 3** we will show how to exchange data with the DS100/EM100 using the TCP/IP protocol. Supporting TCP/IP is slightly more difficult so we will do this at the end of the *Tutorial.*

You can follow our instructions step-by-step or you can download a completed project for each of the steps from www.tibbo.com/downloads.htm. The projects are archived (.zip) and named "*WinSock Demo Step1*", "*WinSock Demo Step2*", and "*WinSock Demo Step3*". **Important:** you may need to change the *RemoteHost* property of the *WinSock* control in each project to the actual IP-address you've set for your DS100/EM100.

3.2. Step 1: creating a simple UDP/IP *Demo*

Follow the instructions below to complete the first step of our action plan:

- Lunch the *Visual Basic (VB)*, choose the *Standard EXE* project
- Rename the default form (Form1) to *FormMain*, set its caption to "WinSock Demo"
- Save the project (and do use the *Save* button often!)
- Add the *WinSock* control to your Project:
  - Choose *Project→Components* from the main menu- *Components* dialog will open

- Scroll down to *Microsoft Winsock Control 6.0* and “check” it
- Click *OK*- the *Winsock* control will be added to your project’s *Toolbox*

- Add the following controls to your Project’s form (see *Fig. 3*):

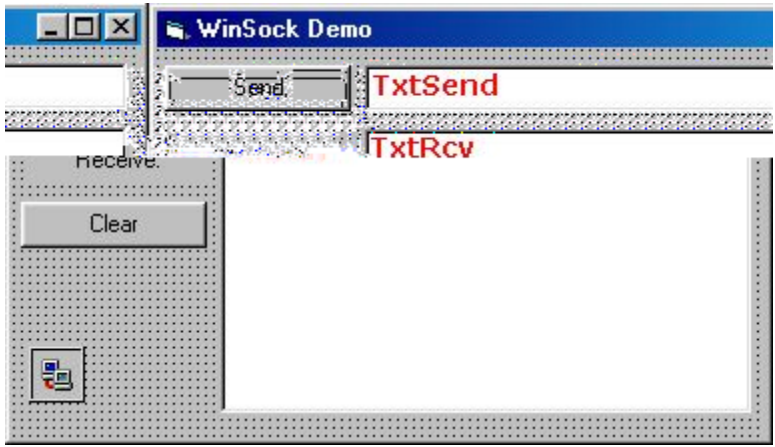

Fig. 3. Controls for the Demo, step 1

- *CmdSend* CommandButton (change its *Caption* property to “Send”)
- *TxtSend* TextBox (change its *Text* property to “”, i.e. empty string)
- *LblRcv* Label (change its *Caption* property to “Receive:”)
- *TxtRcv* TextBox (change its *Text* property to “”, i.e. empty string) and set its *MultiLine* property to “True”. Make this TextBox span several lines as shown on *Fig. 3*
- *CmdClear* CommandButton (change its *Caption* property to “Clear”)

- *WinSock* control, adjust its properties as follows:
- *Protocol:* 1-sckUDPProtocol
- *RemoteHost:* 192.168.100.40 (or whatever IP-address you've set for the DS100/EM100SK)
- *RemotePort:* 1001 (must be specified!)

- Add the following code to the project:
  - For the *Click* event of the *CmdSend* button:

```
Private Sub CmdSend_Click()
    WinSock.SendData TxtSend.Text
End Sub
```

  - For the *DataArrival* event of the *WinSock* control:

```
Private Sub WinSock_DataArrival(ByVal bytesTotal As Long)
    WinSock.GetData s$
    TxtRcv.Text = TxtRcv.Text + s$
End Sub
```

  - For the *Click* event of the *CmdClear* button:

```
Private Sub CmdClear_Click()
    TxtRcv.Text = ""
End Sub
```

That's it! The Project is done! *Save* it and proceed to testing!

- Run the *Demo* you have created, run the HyperTerminal and configure it as explained in Setting up the HyperTerminal if necessary
- Type any string in the *Send* textbox and press *Send*

At this point you will likely be shown… an error message saying some nonsense like "Address family is not supported" or just an error code without any explanation. This is a peculiar bug in the WinSock control. Manual way around this situation is to stop the program and simply click once on the WinSock control in your FormMain form (in the design mode). Next time you run the Demo this problem won't show up! This is not a solution for a "real" application you are probably starting to develop. One way to avoid this annoying message is to use the OnError statement to filter out the error.

- Well, again (if necessary ☺)… run the *Demo*, type any string and press *Send*- the string will appear in the *HyperTerminal's* window
- Type anything in the *HyperTerminal's* window- the same string will appear in the *Receive* textbox of your *Demo*.
- Congratulations! Step 1 successfully completed!

3.3. Step 2: learning to control the DS100/EM100 over the network

The DS100/EM100 can be programmed (setup) over the network (see *Section 4.3* of the *DS100 Tech Manual*

```
    WinSock.GetData s$
    TxtRcv.Text = TxtRcv.Text + s$ + Chr$(13) + Chr$(10)
End Sub
```

- Run the Demo again
- In the *Send* textbox type “V” and click *Send*, the DS100/EM100 will respond with its version string, i.e. “V2.20”

You can type most commands like this. Right now we will show you how to change the Transport Protocol (i.e. the protocol the DS100/EM100 uses to receive and send the data) from UDP/IP to TCP/IP.

- To change any Setting of the DS100/EM100 over the network you need to login. Login password of the freshly initialized DS100/EM100 is NULL (empty string) but login is still necessary. Type “L” and press Send. The DS100/EM100 will respond with “A”
- To select the TCP/IP transport protocol send this command: “STP1”. Again, the DS100/EM100 will respond with “A”
- End the Network Setup Session by sending “E”. The DS100/EM100SK will not respond to this command but simply reboot. The DS100/EM100SK is now using TCP/IP.

One last thing that remains to be demonstrated is the use of broadcast commands. Broadcast commands are sent using Ethernet broadcast packets. These packets reach all devices attached to the same network segment as your PC. Broadcast commands allow you to find all locally attached DS100/EM100s and also assign a new IP-address to any of these devices. The latter is very important- you can setup the DS100/EM100 even when its IP-address is not suitable for the current network and regular “IP” communications is not possible. Example below shows the use of the broadcast commands:

- Stop the *Demo*

- Change the *RemoteHost* property of the *WinSock* to 255.255.255.255. Such an IP-address is, in fact, invalid. For the *WinSock* control it means that all packets must be sent in the broadcast mode
- Modify the code (add a line) for the *Click* event of the *Send* button:

```
Private Sub CmdSend_Click()
    WinSock.RemoteHost = "255.255.255.255" 'takes care of the bug
    WinSock.SendData TxtSend.Text
End Sub
```

There is another bug in the WinSock control. Without the line that repeatedly assigns "255.255.255.255" each time you are about to send the datagram the WinSock will not always process all the replies from all the DS100/EM100s!

- Run the *Demo*
- Send "X" command. If you have several DS100/EM100s attached to the local network segment you will get a reply from each device! For example:

```
A0.2.3.4.12.250/1001
A.0.2.3.4.23.126/1000
A.1.2.3.4.23.127/1001
```

Each reply contains the Ethernet (MAC) address of the DS100/EM100 and its current Data Port number. The Ethernet address is important because it is unique for every DS100/EM100 (and, in fact, for every other Ethernet device connected to your network). You can use the Ethernet address to reference a specific DS100/EM100 whose IP-address you want to change!

- You can now assign a new IP-address to the DS100/EM100 you are playing with (make sure you choose correct new IP-address). For example, if the Ethernet address of this DS100/EM100SK is 1.2.3.4.23.127 and you want to change the IP-address to 192.168.100.41 then send this command: "A0.2.3.4.23.126//192.168.100.41" (the empty field in the middle is for login password which is now NULL). The DS100/EM100 will reply with "A" and start using the new IP-address

3.4. Step 3: TCP/IP communications

Finally, let's modify the Demo to communicate with the DS100/EM100 using the TCP/IP protocol. This protocol has many advantages over the UDP/IP. Naturally, this also implies a somewhat more complicated programming.

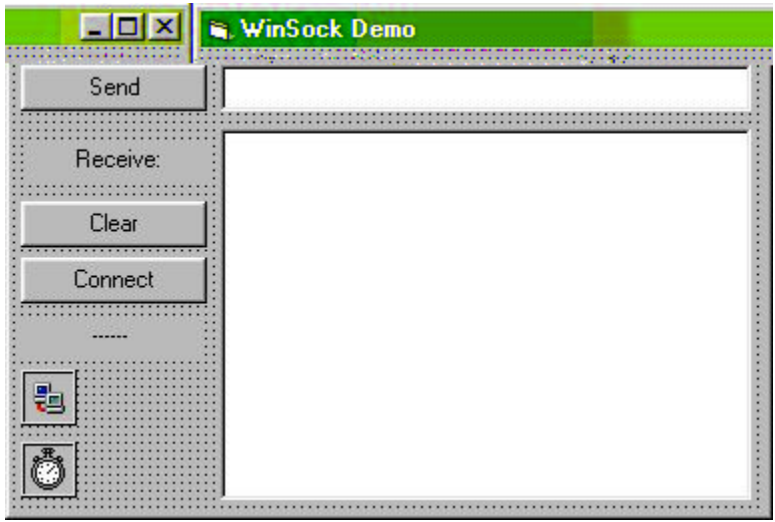

Fig. 4. Controls for the Demo, step 3

- Stop the *Demo*
- If you have assigned a new IP-address to the DS100/EM100 in the previous section then change the *RemoteHost* property of the *WinSock* control accordingly
- Change the *RemotePort* property of the *WinSock* back to 1001

- Change the *Protocol* property of the *WinSock* control to “0- sckTCPProtocol” (you have already switched the DS100/EM100 to TCP/IP earlier)
- Change the *Enabled* property of the CmdSend button to “Disabled” (because the data cannot be sent unless the connection is established)
- The TCP/IP is a connection-oriented protocol so we will need to add a button to connect to/ disconnect from the remote host. Add *CmdConnect* CommandButton to your form, set its *Caption* to “Connect” (see *Fig. 4*)
- Also add another Label named *LblState*- this will be used to track the change in TCP/IP connection states. Set this Label’s *Caption* to “-------”
- Add one Timer, *Timer1*, to the form too. Set the Timer’s *Interval* to 100 (i.e. 10 times/second)
- Add the code below to the *Timer* event of *Timer1* This code does three things. 1) It periodically updates the status of the TCP/IP connection. 2) Dynamically changes the Caption of the *CmdConnect* button: when the connection is closed (*WinSock.State=0*) the *Caption* reads “Connect”, when the connection is *not* closed the *Caption* reads “Disconnect”. This way the button can be used as a “toggle switch”. 3) The code also disables or enables the *CmdSend* button- the data can be sent only when connection is established

```
Private Sub Timer1_Timer()
    'display current connection state
    Select Case WinSock.State
        Case 0: LblState.Caption = "Closed"
        Case 1: LblState.Caption = "Open"
        Case 2: LblState.Caption = "Listening"
        Case 3: LblState.Caption = "Connection pending"
        Case 4: LblState.Caption = "Resolving host"
        Case 5: LblState.Caption = "Host resolved"
        Case 6: LblState.Caption = "Connecting"
        Case 7: LblState.Caption = "Connected"
```

```
        Case 8: LblState.Caption = "Peer closing"
        Case Else: LblState.Caption = "Error"
    End Select

    'set the caption of the "Connect" button
    If WinSock.State = 0 Then
        CmdConnect.Caption = "Connect"
    Else
        CmdConnect.Caption = "Disconnect"
    End If

    'enable or disable the "Send" button
    If WinSock.State = 7 Then
        CmdSend.Enabled = True
    Else
        CmdSend.Enabled = False
    End If
End Sub
```

- Add the code below to the *Click* event of the *CmdConnect* button. This code selects open the connection when it is closed and to close the connection when it is not closed:

```
Private Sub CmdConnect_Click()
    If WinSock.State = 0 Then
        WinSock.Connect
    Else
        WinSock.Close
    End If
End Sub
```

- "Comment out" disable the line in the *Click* event of the *Send* button (we have added this line to make the *WinSock* correctly display all the replies from the devices in the broadcast mode. In our current TCP/IP test this is not only unnecessary, but erroneous- in TCP/IP you cannot change the IP-address after the connection has been established!):

```
Private Sub CmdSend_Click()
    'WinSock.RemoteHost = "255.255.255.255" Comment out for TCP test
    WinSock.SendData TxtSend.Text
End Sub
```

- Finally, "comment out" the CR/LF addition to every packet received from the DS100/EM100. It is convenient for the Network Setup but will look strange for a "normal" data exchange:

```
Private Sub WinSock_DataArrival(ByVal bytesTotal As Long)
    WinSock.GetData s$
    TxtRcv.Text = TxtRcv.Text + s$ '+ Chr$(13) + Chr$(10) Comment out
End Sub
```

Now you can test your TCP/IP *Demo*!

- Run the *Demo*. The *Status* of the TCP/IP connection will be "Closed". The *Send* button will not be active
- Click *Connect* button. The *Status* will change to "Connecting", then to "Connected". If the program cannot connect to the DS100/EM100 (because the device is off, because you are connecting to a wrong IP-address, because somebody is already connected to this DS100/EM100, etc. etc.) then the *Status* will hang on "Connecting" for a while, then change to "Error"
- With TCP/IP connection established the *Send* button will be enabled and you can exchange the data between the *Demo* and the *HyperTerminal*

And this is the end of our Tutorial. Hope this wasn't too boring! ☺