

# ***Connecting to a Thin Web Server via a Direct SLIP Connection in Windows XP***

---

## **Introduction**

This Application Note explains how to connect a PC running Windows XP to a thin web server via a direct Serial Line Internet Protocol (SLIP) connection.

The thin web server hardware is from the CubeSat Kit (<http://www.cubesatkit.com/>), with a TI MSP430F149 single-chip microcontroller (<http://www.ti.com/>). The TCP/IP stack and thin web server software are from Adam Dunkel's uIP TCP/IP stack (<http://www.sics.se/~adam/uip/>). The uIP web server application runs as a pair of tasks in a Salvo (<http://www.pumpkininc.com/>) RTOS multitasking application on the CubeSat Kit's MSP430F149.

The direct SLIP connection is made via a simple RS-232 serial cable between the Windows PC and the thin web server.

Static IP addresses are used for the server and client.

For more information on the various components above, please see their respective manuals and documentation.

## **Before You Begin**

You will need a thin web server configured for SLIP, and a PC with a serial port running Windows XP. Connect them with a serial cable.

## SLIP Details

### What is SLIP?

SLIP is a means of transmitting Internet Protocol (IP) datagrams between two nodes on a network. It can be implemented over any kind of serial interface, whether it be wired, infrared, wireless, etc. Therefore any embedded system with software to support TCP/IP and SLIP can be connected to the Internet via a low-cost serial link. Assuming that there is a PC, router or other similar hardware to complete the SLIP connection, high-level functionality like web servers, telnet servers, etc. can be implemented on the embedded target and can be accessed via commonly available tools like browsers, telnet clients, etc.

### IP Addresses

Just like Ethernet networks, SLIP requires the use of IP addresses. In this example, we'll assign a private IP address of 172.16.1.2 to the Windows PC SLIP client, and 172.16.1.10 to the thin web server.

## Thin Web Server Specifics

### Memory Requirements

A uIP stack and thin web server application requires approximately 6-15KB of program memory and under 500 bytes of RAM on the embedded target.<sup>1</sup> An MSP430, with 55KB program memory and 2KB RAM affords plenty of room for the uIP TCP/IP stack, thin web server, Salvo multitasking RTOS, transmit and receive buffers, and other application-specific code.

### Processing Power & Buffer Sizes

The example thin web server runs at 7.372MHz with 256-byte transmit and receive buffers, and a 128-byte uIP buffer size.

The RS-232 serial link is configured for 9600,N,8,1 with no hardware or software handshaking.

## Windows XP SLIP Client Behavior

### Initial Connection

In order for the Windows XP SLIP client to be able to connect to the thin web server, the thin web server must issue the response string `CLIENTSERVER` after receiving the string `CLIENT` from the Windows XP client.

---

**Note** SLIP clients running under different operating systems may not have the same initial challenge/response behavior.

---

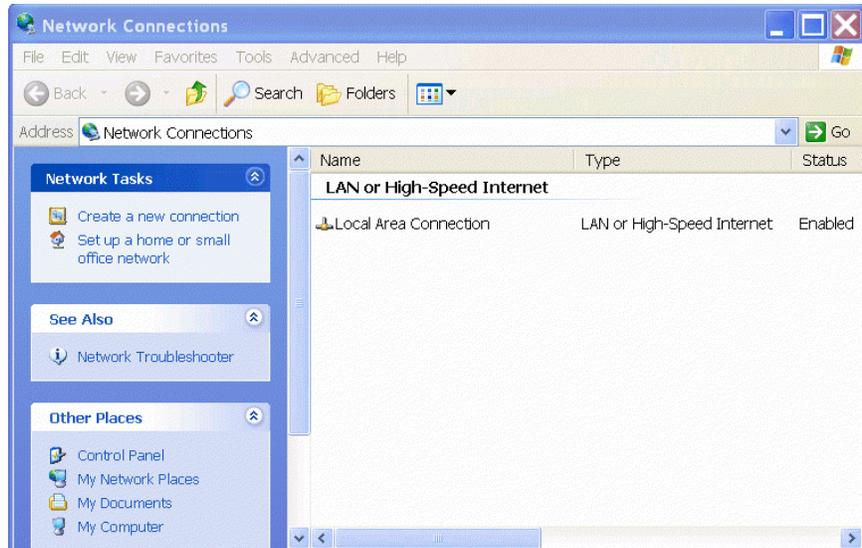
A very simple implementation used for the thin web server in the CubeSat Kit demonstration software is shown in Listing 1 in *Appendix A – Client/Server Discovery Code*.

### Once a Connection is Established

Even though a SLIP connection is a point-to-point connection, under Windows XP, the SLIP client will initially create a lot of TCP/IP traffic, not all of which is destined for the node at the other end of the SLIP link. It's important for the thin web server to be able to properly handle IP traffic that is not destined for it – i.e. traffic that does not have the thin web server's IP address as its destination address. Normally, such traffic is simply ignored by the thin web server.

## Configuring the Windows XP SLIP Client

Open the Network Connections window using Start → All Programs → Accessories → Communications → Network Connections:



**Figure 1: Network Connections Window**

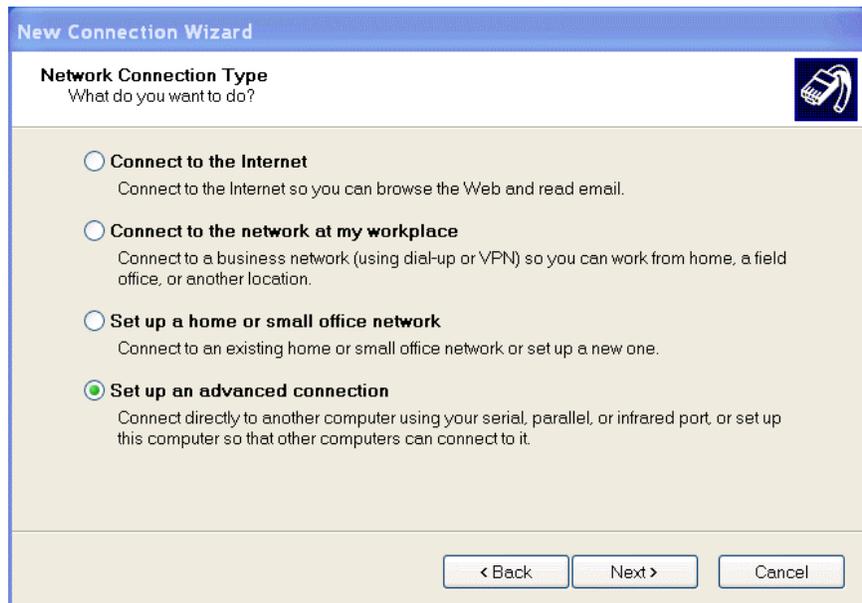
Click on Create a new connection and you will be greeted with the New Connection Wizard window:



**Figure 2: New Connection Wizard**

Click on Next.

Select Set up an advanced connection:



**Figure 3: Selecting an Advanced Connection**

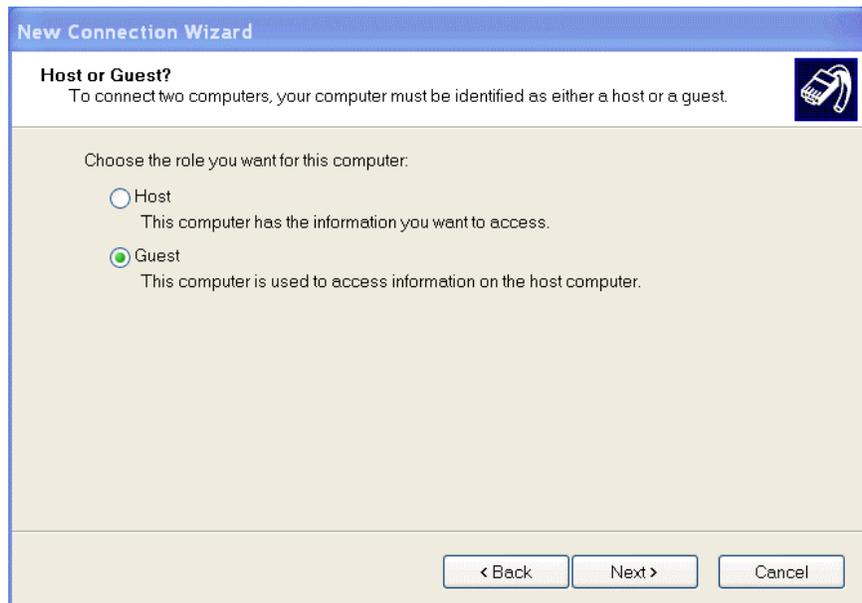
Click on Next. Click on Connect directly to another computer:



**Figure 4: Selecting a Direct Connection to Another Computer**

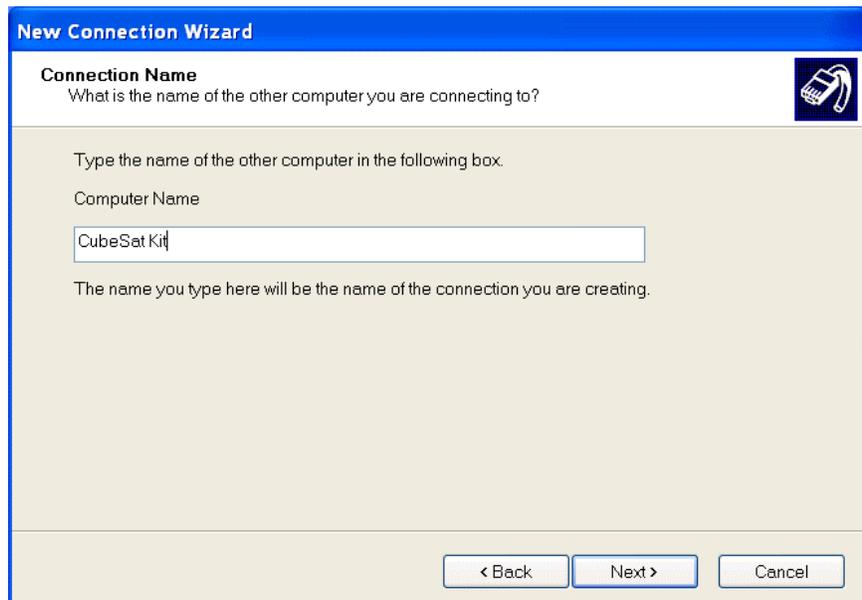
Click on Next.

Click on Guest:



**Figure 5: Identifying the PC as a Guest**

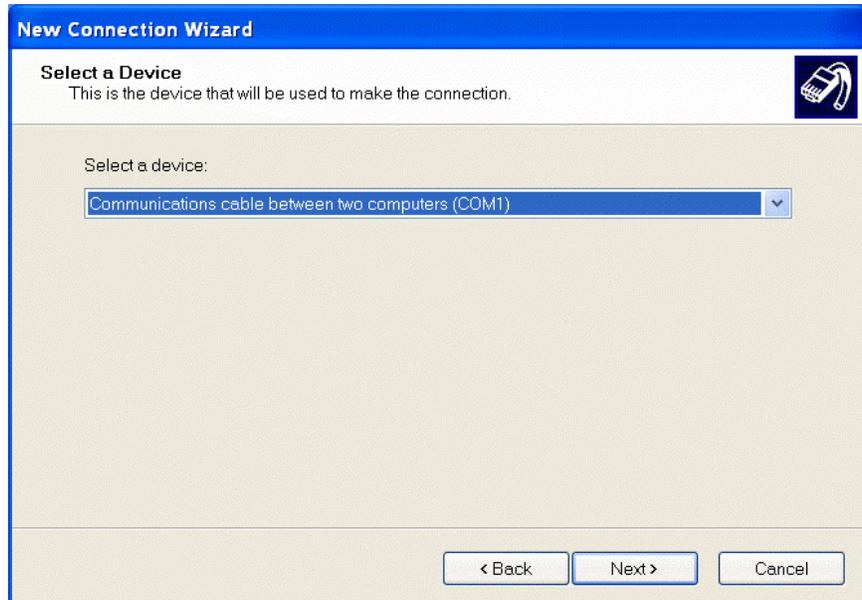
Click on Next. Enter a Connection Name:



**Figure 6: Naming the Connection**

Click on Next.

Under **Select a device**, select the serial interface between the Windows PC and the thin web server. Normally this will be a serial cable using one of the PC's COM ports:



**Figure 7: Selecting the Device**

Click on **Next**. The **New Connection Wizard** is complete:



**Figure 8: New Connection Wizard is Complete**

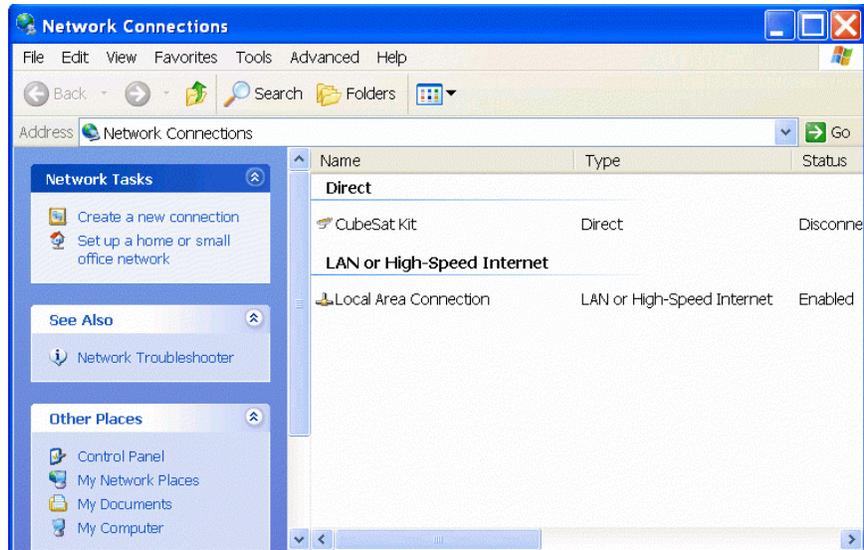
Click on **Finish**.

The named connection window will open:



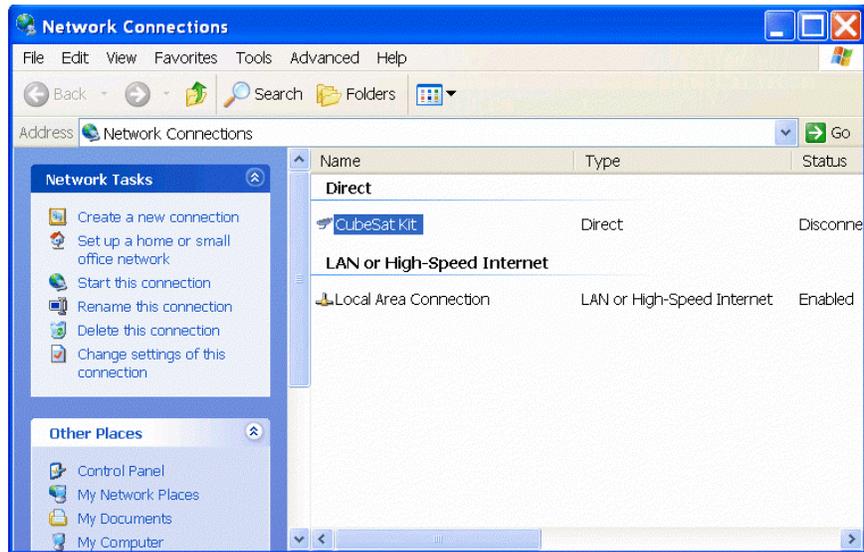
**Figure 9: Direct Connection Start Window**

The connection's properties are not yet configured for SLIP, so click on **Cancel**. In the **Network Connections** window you'll see that a new **Direct** connection is now available:



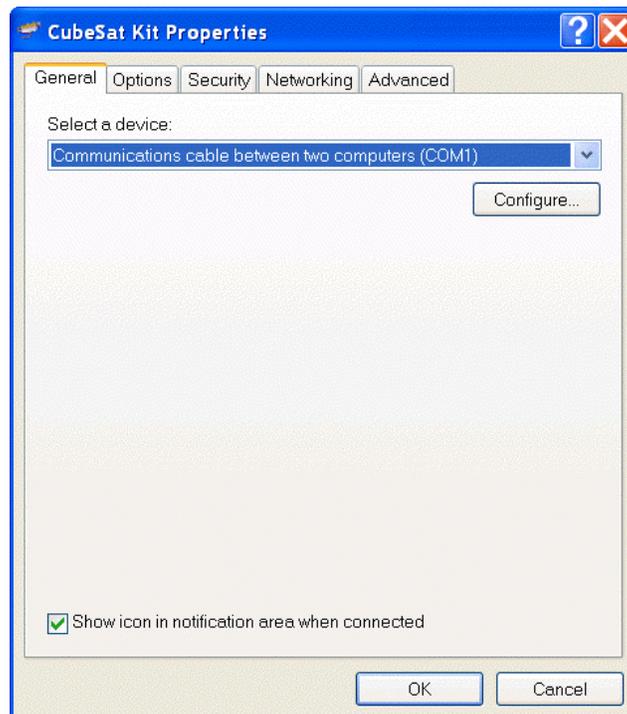
**Figure 10: New Direct Network Connection to Thin Web Server**

Select the direct connection to the thin web server:



**Figure 11: Selecting the Direct Connection**

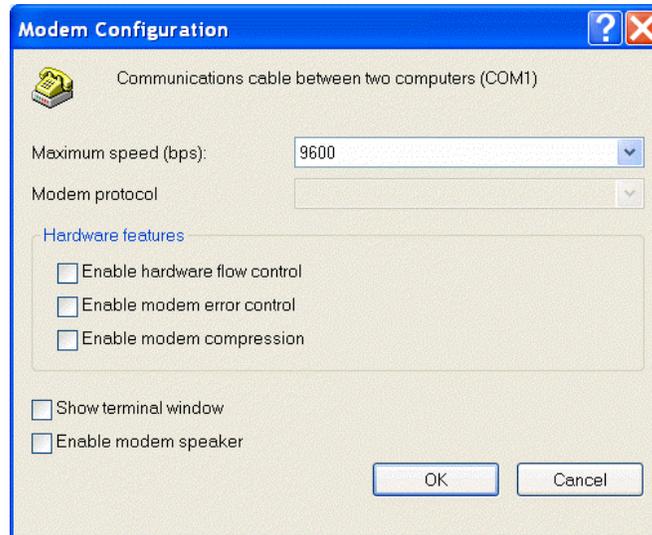
Click on Change settings of this connection:<sup>2</sup>



**Figure 12: General Properties of Direct Connection**

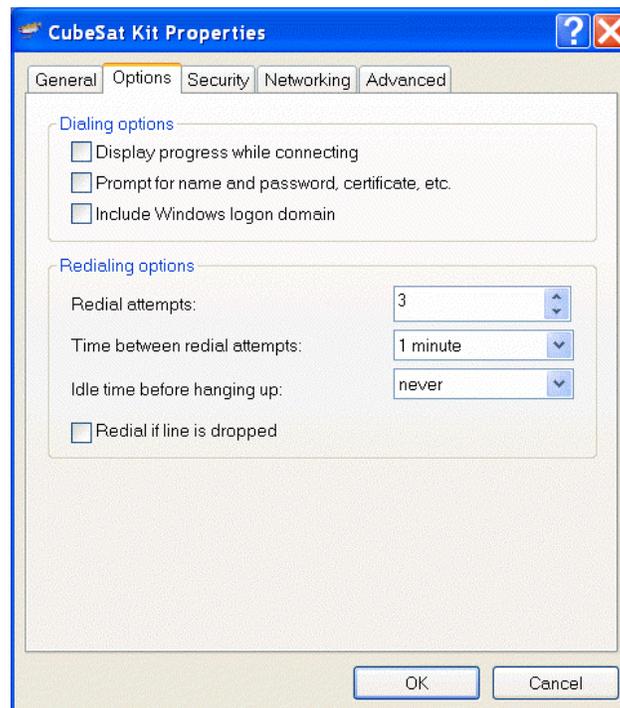
Click on Configure.

Under Maximum speed (bps), choose the connection speed and select the Hardware features that are supported by the thin web server:



**Figure 13: Modem Configuration**

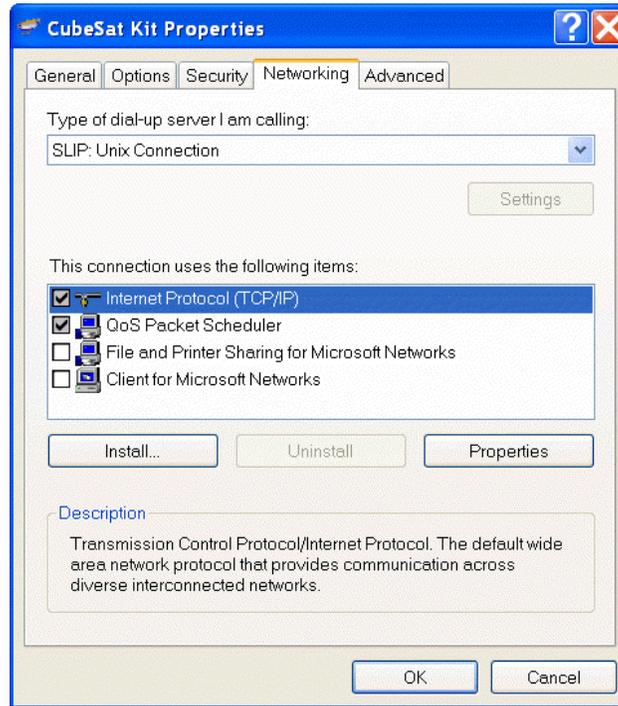
Click on OK, then Options:



**Figure 14: Disabling the Dialing Options**

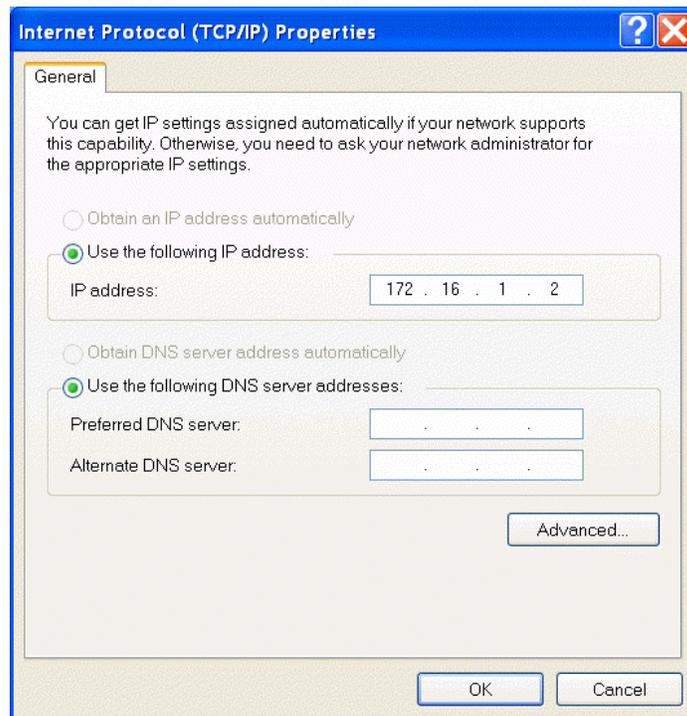
Disable all of the Dialing Options. Click on the Networking tab.<sup>3</sup>

Under Type of dial-up server I am calling, choose SLIP: Unix Connection. Only Internet Protocol (TCP/IP) is required:



**Figure 15: Networking Options**

Select Internet Protocol (TCP/IP), then click on Properties:



**Figure 16: TCP/IP Properties**

Under **Use the following IP address**, assign an IP address to the Windows PC SLIP client that is on the same subnet as the thin web server. Click on **OK**, then click on the **Advanced** tab.

The default settings are acceptable:

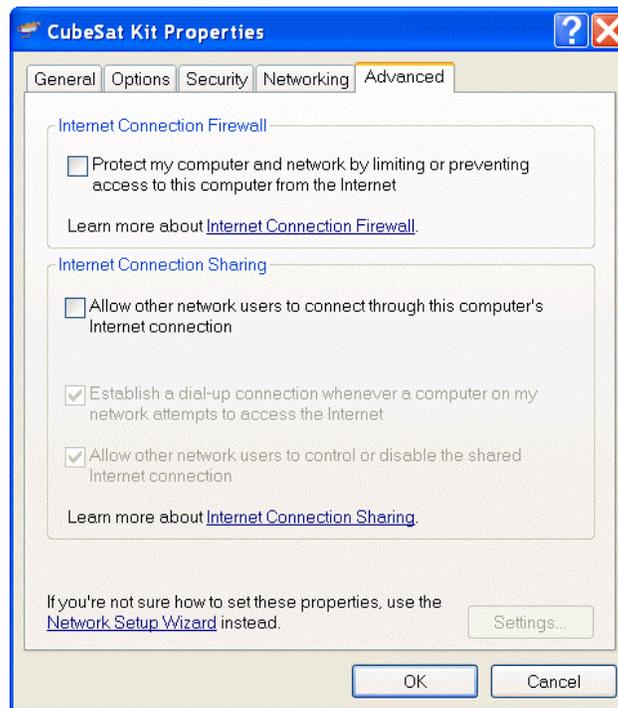


Figure 17: Advanced Settings

Click on OK. The direct SLIP connection is now properly configured.

## Using the SLIP Connection

### Starting the Connection

Turn on the thin web server and connect it to the Windows XP SLIP client via the serial cable. Start the SLIP connection by double-clicking on the direct connection in the Network Connections window.

## Verifying the Connection

The network connection icon for the SLIP connection will appear in the Windows XP toolbar once a connection is made. You can check it via its **Status** pop-up at any time:

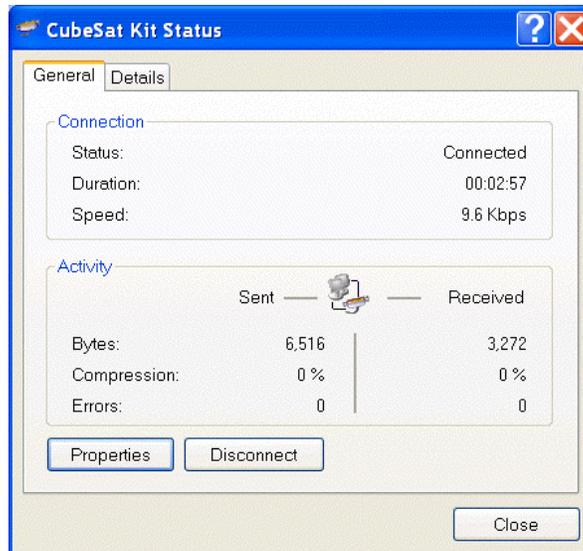


Figure 18: SLIP Connection Status Window

You can also verify the connection via the Windows `ipconfig.exe` command. Open the Windows XP Command Prompt window and type `ipconfig`:

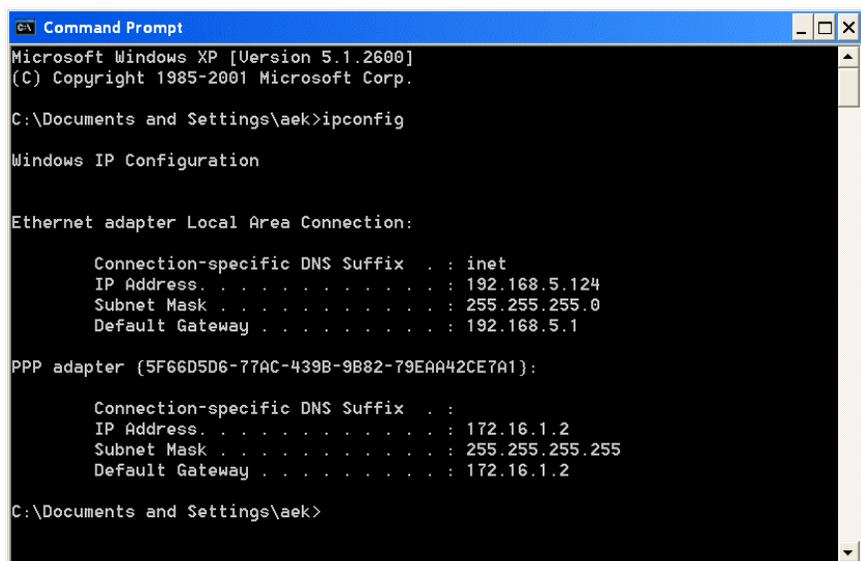
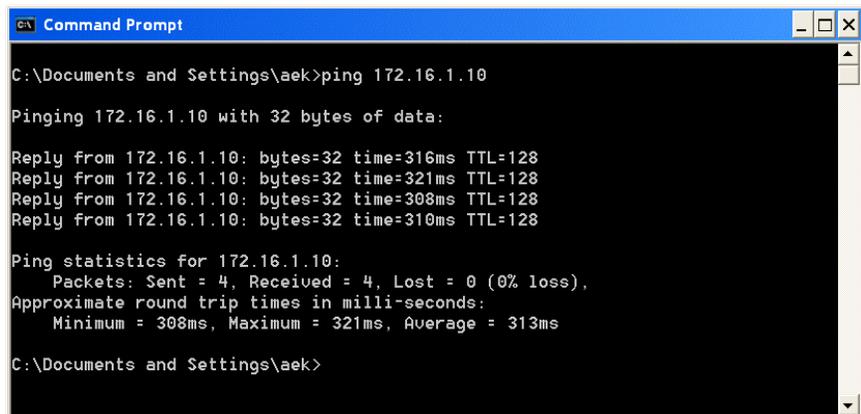


Figure 19: Ipconfig Results

When everything is working properly, `ipconfig` will report that a PPP adapter with the desired IP Address is present.

## Testing the SLIP connection

Assuming that your thin web server supports ICMP and the ubiquitous `ping` command, the simplest way to test the connection is by pinging the thin web server. From the Command Prompt window, ping the thin web server using its IP address, e.g. `ping 172.16.1.10`:



```
Command Prompt
C:\Documents and Settings\æk>ping 172.16.1.10

Pinging 172.16.1.10 with 32 bytes of data:

Reply from 172.16.1.10: bytes=32 time=316ms TTL=128
Reply from 172.16.1.10: bytes=32 time=321ms TTL=128
Reply from 172.16.1.10: bytes=32 time=308ms TTL=128
Reply from 172.16.1.10: bytes=32 time=310ms TTL=128

Ping statistics for 172.16.1.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 308ms, Maximum = 321ms, Average = 313ms

C:\Documents and Settings\æk>
```

Figure 20: Ping Results<sup>4</sup>

---

**Note** You can also ping the PC itself as an additional test, e.g. `ping 172.16.1.2`.

---

**Tip** You can ping continuously by using the `ping` command's `/t` command-line option. E.g. `ping 172.16.1.10 /t`. Use `Ctrl-Break` to exit.

---

## Closing the Connection

The SLIP connection can be closed by clicking on `Disconnect` in the direct connection's `Status` window.

## Serving Web Pages via SLIP

Once the SLIP connection has been established, you can browse them via your PC's browser. Just enter the IP address of the thin web server, e.g. `http://172.16.1.10/`:



Figure 21: Web Page served from Thin Web Server over SLIP

---

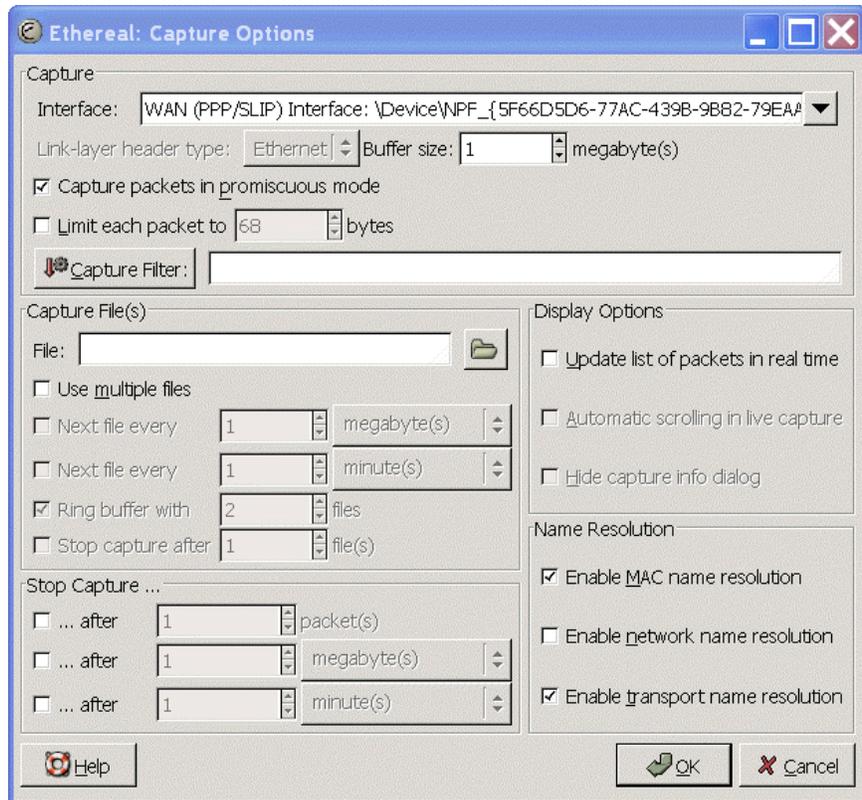
**Note** SLIP connections usually run at much slower speeds (e.g. 9600bps) than a typical Ethernet connection (at 10, 100 or even 1,000Mbps). Therefore web pages will take longer to load at the client end than you may be accustomed to.

---

## Protocol Analysis of the SLIP Connection

Since a SLIP connection simply encapsulates normal TCP/IP traffic in a form suitable for point-to-point serial transmission, any protocol analyzer that can handle SLIP can be used to analyze the traffic between the thin web server and the Windows XP SLIP client.

One such freely available multi-platform protocol analyzer is Ethereal (<http://www.ethereal.com/>). Once your SLIP connection is established, it's easy to capture the traffic on the SLIP link. Simply select the appropriate WAN (PPP/SLIP) interface detected by Ethereal and begin capture:



**Figure 22: Selecting the SLIP link for Capture in Ethereal**

**Note** A protocol analyzer like Ethereal can only analyze the TCP/IP traffic over the SLIP link *once the SLIP connection has been established*. It cannot be used to debug the initial exchange of information between the PC SLIP client and the thin web server.

Once Ethereal's capture is has begun, you can request web pages over the SLIP link and Ethereal will provide a human-readable analysis of each packet captured:

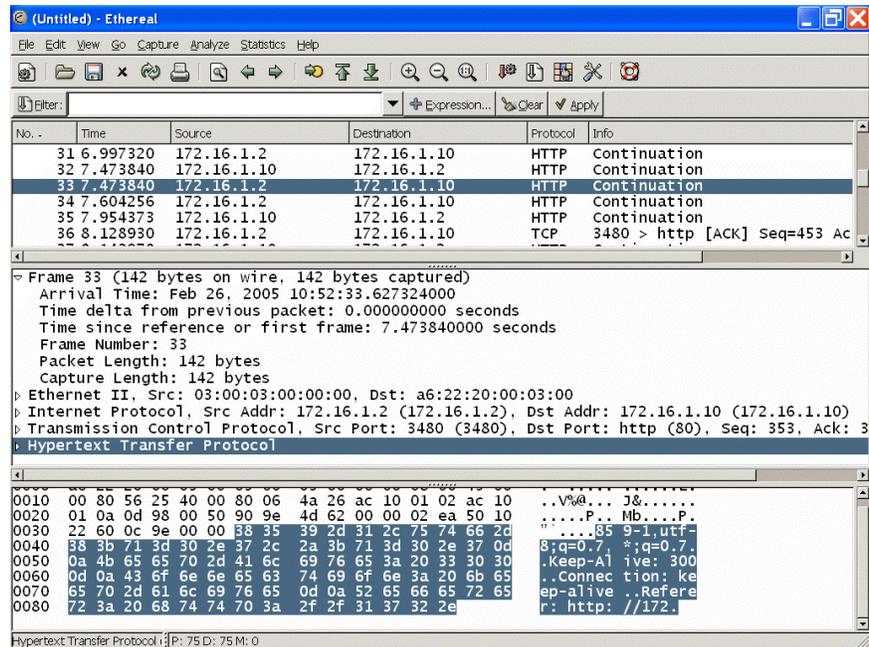


Figure 23: Ethereal SLIP Packet Analysis

## Other Operating Systems

Setting up a SLIP client under Windows 2000 is substantially similar to the situation in Windows XP.

Setting up a SLIP client in Windows 98 through a direct serial connection appears to be impossible. Windows 98 always expects a modem to be present, and requires responses to certain AT command sets when attempting to initialize the modem as part of establishing the serial link.

## Multiple-Node SLIP Connections

It is possible to establish a network of a single SLIP client and multiple thin web servers. That's because SLIP packets have source and destination IP addresses. As long as each thin web server on the network has a unique IP address on the network, then the SLIP client will be able to access unique web pages on each thin web server.

For example, using a wireless network where each device's serial port is connected to a wireless modem, then a Windows XP SLIP client with address 172.16.1.2 can access two thin web servers, one at 172.16.1.10 and one at 172.16.1.13. Tests with each thin web server running the uIP TCP/IP stack show that pages will be served even if there is continuous traffic on the network (e.g. constant `ping`ing of one or more nodes on the network).

## Troubleshooting

### Windows XP SLIP Client Won't Connect

For the Windows XP client to successfully create a SLIP connection, it must successfully communicate with the thin web server first. You'll need to verify that the thin web server has indeed received the `CLIENT` string from the SLIP client, and has transmitted the `CLIENTSERVER` response string.

### Windows XP SLIP Client Connects, Won't Talk

If a SLIP connection has been established but no response is received from the thin web server (e.g. via a `ping` test), then you'll need to check your IP addresses and your SLIP code on the thin web server.

### Browser Timeouts

When accessing complex web pages on the thin web servers, a browser may issue a timeout message or other error message. A simple test is to request a page that doesn't exist on the server. The response from the server is generally very small and involves little data. You may need to tweak your thin web server settings and/or reduce the size of your web pages if your serial link is so slow that such timeouts occur.

### Cabling

If implementing SLIP over an RS-232 connection, proper cabling must be used. If both devices are DTE devices (PC serial ports are always DTE), a null-modem cable will be required between the two devices. If the thin web server is a DCE device, then a pass-through cable will be required.

## Appendices

### Appendix A – Client/Server Discovery Code

The C code<sup>5</sup> in Listing 1 is used to establish a SLIP connection between a thin web server and a Windows XP PC SLIP client:

```
void TaskOpenSLIP(void)
{
    FM430status.SLIPstate = SLIP_DISCONNECTED;

    for (;;)
    {
        switch ( FM430status.SLIPstate )
        {
            case SLIP_DISCONNECTED:

                /* if Rx0 chars have been received ... */
                if ( SLIPBuffSize() != 0 )
                {
                    /* Did the client say "CLIENT"? */
                    if ( strstr(SLIPBuff,"CLIENT"))
                    {
                        /* Yes, so mark as connected, and */
                        /* start the uIP tasks. */
                        FM430status.SLIPstate = SLIP_CONNECTED;
                        OSStartTask(TASK_DO_UIP_P);
                        OSStartTask(TASK_DO_UIP_PERIODIC_P);

                        /* Tell client we're here. */
                        SLIPputs("CLIENTSERVER\n");

                        /* Flush buffer (i.e. remove non-SLIP */
                        /* stuff still in it). */
                        OS_Delay(100);
                        SLIPResetUSART();
                        OS_Stop();
                    }
                }

                /* check for "CLIENT\n" every 500ms */
                /* until connected */
                OS_Delay(50);
                break;

            case SLIP_CONNECTED:
                OS_Stop();
                break;

            default:
                break;

        } /* switch() */
    } /* for() */
}
```

**Listing 1: Challenge-Response Code for Thin Web Server**

`TaskOpenSLIP()` checks the receive buffer that holds incoming packets every 500ms for the character string `CLIENT`. Once such a string is detected, it launches the two uIP tasks that run the thin web server<sup>6</sup>, sends the character string `CLIENTSERVER`, and prepares the receive buffer for SLIP packets. Once the SLIP connection has been established, `TaskOpenSLIP()` simply stops.

---

**Note** This discovery code is active only once per SLIP session. I.e. `TaskOpenSLIP()` must start from reset, then the Windows XP SLIP client should be started. If the Windows XP SLIP client is stopped / killed and then restarted, a SLIP connection with the thin web server will not be re-established, because `TaskOpenSLIP()` is stopped. A restart of the thin web server will be required.

---

## Appendix B – uIP Tasks

The C code in Listing 2 is runs the uIP TCP/IP stack as two separate tasks in a Salvo RTOS multitasking operation.

```
void TaskDoUIP(void)
{
    slipdev_init(); // start SLIP
    httpd_init(); // start HTTP on port 80

    for (;;)
    {
        uip_len = slipdev_poll();

        if(uip_len > 0)
        {
            uip_input();

            if(uip_len > 0)
            {
                slipdev_send();
            }
        }

        OS_Delay(1);
    }
}

void TaskDoUIPPeriodic(void)
{
    unsigned char i;

    for (;;)
    {
        for(i = 0; i < UIP_CONNS; ++i)
        {
            uip_periodic(i);

            if(uip_len > 0)
            {
                slipdev_send();
            }
        }

        OS_Delay(50);
    }
}
```

**Listing 2: Salvo Tasks to Implement uIP TCP/IP Stack**

After initializing the SLIP link and starting the thin web server, TaskDoUIP() checks for new SLIP packets every 10ms. If a packet is present, it processes it, and if the result is that an outgoing packet is ready to be sent, it sends it out via SLIP.

`TaskDoUIPPeriodic()` checks all of the open connections every 500ms for activity, and sends out SLIP packets as required.

---

**Note** Both uIP tasks could be combined into a single task if so desired.

---

- 
- <sup>1</sup> All figures are approximate because of the high degree of configurability of the software involved.
  - <sup>2</sup> Alternately, you can select the direct connection and choose its Properties.
  - <sup>3</sup> There are no Security Settings for a SLIP connection.
  - <sup>4</sup> In this particular test, the SLIP interface is via a pair of 9600 baud 900MHz wireless modems. The modem link adds slightly to the ping times.
  - <sup>5</sup> And not particularly versatile, as it does not handle re-connections.
  - <sup>6</sup> These two tasks will not run until `TaskOpenSlip()` returns to the scheduler.