Freescale MQX RTOS Example Guide
telnet_to_serial example

This document explains the telnet_to_serial example, what to expect from the example and a brief introduction to the API.

The example

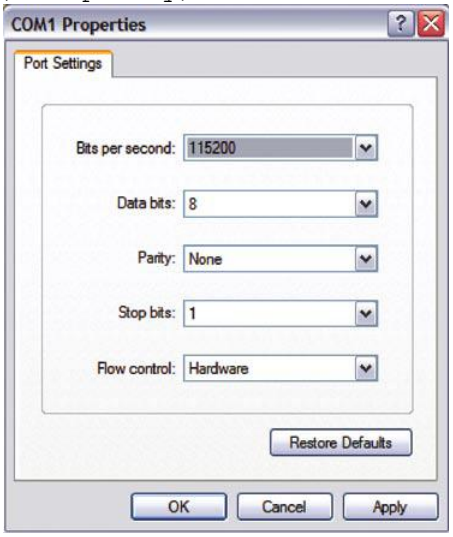
The telnet_to_serial example code is a simple character passing between UART console and telnet session. This shows custom "lightweight" telnet server. The application creates a telnet session in the port 23 where a TCP/IP connection can be established.

Running the example

Connect a serial cable from the UART0 port of the board to the PC. Connect an Ethernet cable from the RJ45 (Ethernet) connector from the board to the RJ45 connector in the PC.

Start HyperTerminal on the PC (Start menu->Programs->Accessories->Communications).
Make a connection to the serial port that is connected to the board (usually will be COM1).

Set it for 115200 baud, no parity, 8 bits and click OK.

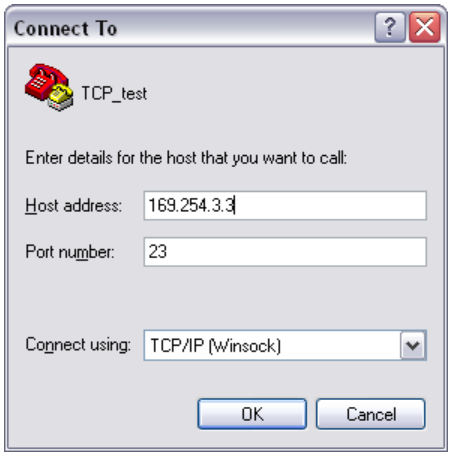


The default IP address of the board is 169.254.3.3. Typically, when you connect your computer directly to the board, the computer will default

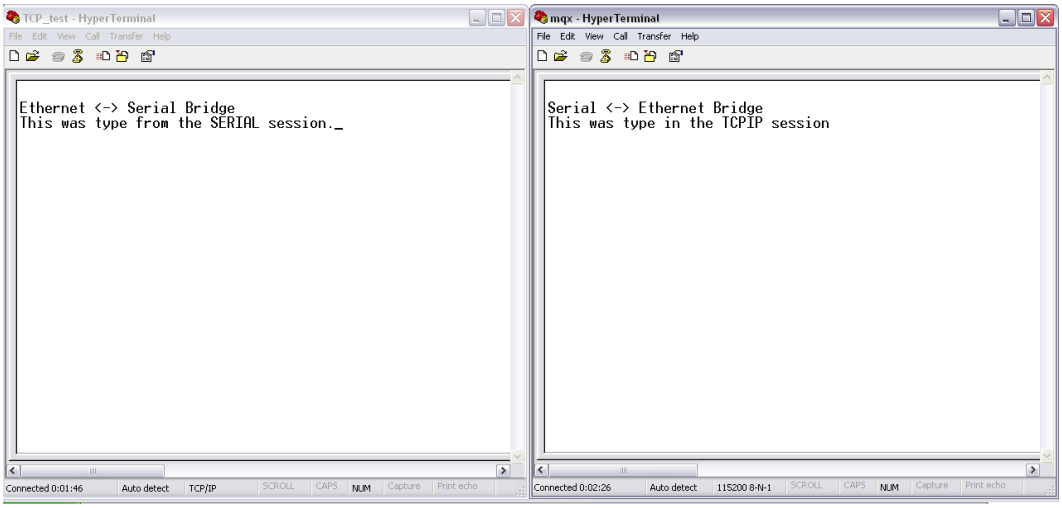
to an auto IP address on the same subnet as the board (169.254.x.x), therefore requiring no setup.

Note: The PC may take a few minutes to default to the auto IP address and make the connection. However, if you have trouble connecting, you may configure the IP address of the computer manually. Select Start > Settings > Network Connections > Local Area Connection. Note your original TCP/IP settings, and then set your IP address to 169.254.3.4 and your subnet mask to 255.255.0.0.

Start a second HyperTerminal on the PC (Start menu->Programs->Accessories->Communications).
Write a the TCP_test name.
In "Connecting using" parameter select "TCP/IP (Winsock)" option.
In the "Host address" introduce the IP address of the board. In this case the 169.254.3.3

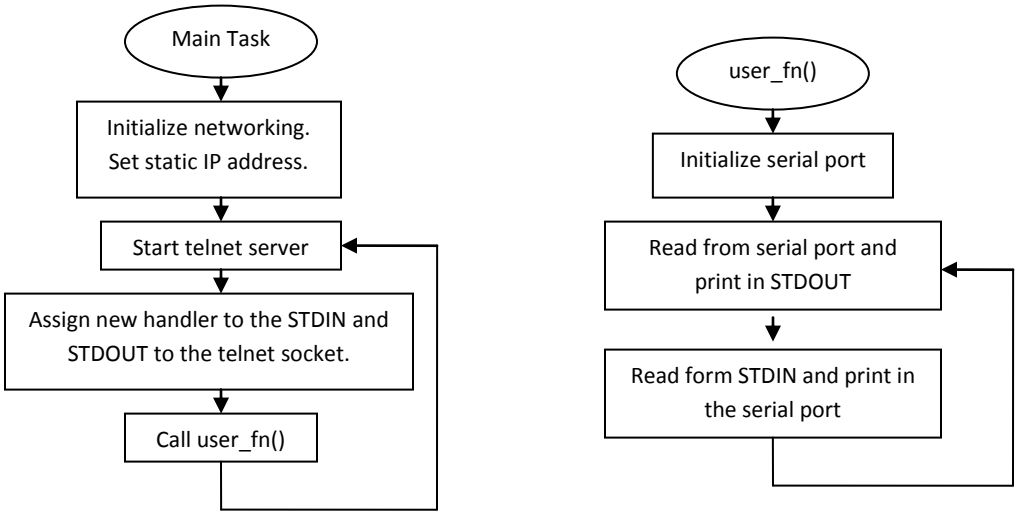


The session is now established. Whatever is typed in one window will appears in the other window. The TCPIP session shows the message "Ethernet <-> Serial Bridge" and the serial session shows "Serial <-> Ethernet Bridge".



Explaining the example

The application demo creates only one main task. The flow of the task is described in the next figure.



The main task starts with the networking initialization and start a telnet server in the port 23. The telnet server is started with the following line:

```
telnetfd = fopen("telnet:", (char_ptr)sockfd);
```

The fopen() calls the telnet application and also sends the socket to be used for the connection.

The server then waits until a new client is connected. When a session starts a socket for the server is created.

```
sock= accept(listensock, NULL, NULL);
sockfd = fopen("socket:", (char_ptr)sock);
```

Once the server gets socket and a new connection is established the STDIN and STDOUT handlers are redirected to this socket where the telnet server is receiving and sending data to the client connected. In a few words, any write to the STDIN and read from the STDOUT is through the telnet socket. The code responsible of it is the next:

```
_io_set_handle(IO_STDIN, telnetfd);
_io_set_handle(IO_STDOUT, telnetfd);
```

After, the function user_fn is called and opens the serial. The application enters into a loop where everything that is received in the serial port is written in the STDOUT which is already redirected to the telnet socket.

```
         c = fgetc( serial_fd );
         if (c==IO_ERROR)
            break;
         putchar((char)c);
```

In this same loop, everything that is received in the STDIN, which is already redirected to the telnet socket; is written in the serial port.

```
         c = getchar();
         if (c==IO_ERROR)
            break;
         fputc((char)c,serial_fd);
```

This way whatever the telnet server receives is passed to the serial port. And whatever the serial port receives is passed to the telnet server.