

# AN10846

## Getting started with NicheLite for LPC3250

Rev. 01 — 1 July 2009

Application note

### Document information

Info	Content
<b>Keywords</b>	LPC3250, NicheLite, Ethernet, TCP/IP Stack, Web Server, TFTP
<b>Abstract</b>	A guide to getting started with NicheLite for LPC3250.

**Revision history**

Rev	Date	Description
01	20090701	Initial version.

**Contact information**

For additional information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: [salesaddresses@nxp.com](mailto:salesaddresses@nxp.com)

## 1. Introduction

The NXP LPC3250 microcontroller is an ARM926E-J-S core with a Vector Floating Point (VFP) co-processor and a large set of standard peripherals, including an Ethernet controller. The “NicheLite for LPC” source code, which is a variant of the full stack available from InterNiche, is available for free download from NXP’s website:

<http://www.standardics.nxp.com/support/software/nichelite/>

This application note will assist the user in configuring and testing the NicheLite Stack using the phyCORE-LPC3250 evaluation board from PHYTEC. Keil tool chain is used (full version is required) in this application note, but the same concepts apply for other tools.

## 2. Connecting and configuring the PHY3250 board

Configure all jumpers in the default position, as per the PHY3250 board’s User Manual.

Follow the below steps in order to prepare the connections (see [Fig 1](#) for details):

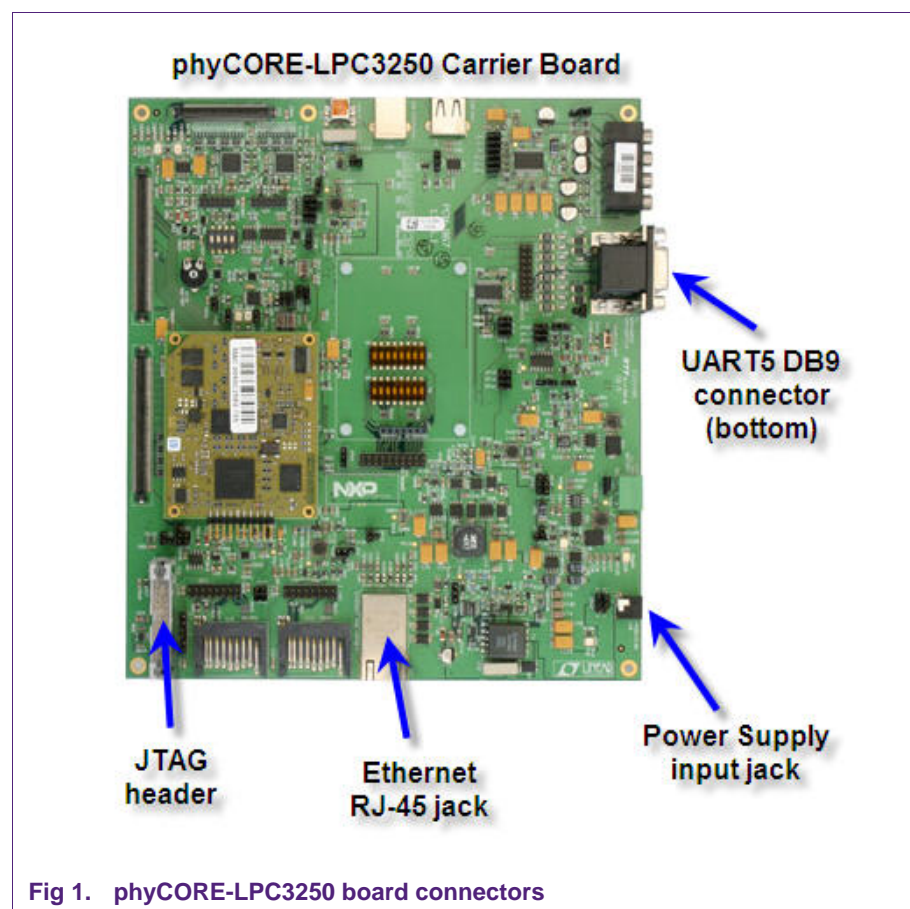


Fig 1. phyCORE-LPC3250 board connectors

1. Connect the Keil U-Link to the JTAG header connector.
2. Connect a serial cable between the PHY3250 board (UART5 DB9 connector) and the PC serial port.

3. Use an Ethernet cable to connect the PHY3250's Ethernet port with an Ethernet router, switch, hub, or directly to a PC (using a crossover cable). If dynamic IP assignment is desired, use a router/switch with DHCP server, or install DHCP server software in the PC.
4. Connect the external Power Supply adaptor to the PHY3250's power input jack.

### 3. Installing the software

Unzip the software into the PC's disk drive. [Fig 2](#) shows the directory structure.



**Fig 2. NicheLite Source Code Organization**

The LPC\_30doc folder contains the documentation.

The LPC\_30src folder contains the source code which is organized in the following directories:

- *allports*: code common to all ports, and not specific to any TCP/IP layer.
- *h*: most header files are included in this folder.
- *listener*: files necessary to implement a simple TCP “listener” (a kind of simple web server).
- *lpc325x*: this folder includes all code specific to the LPC3250.
- *Lst*: all listing files generated by the tool are included in this folder.
- *mip*: this folder contains the implementation of IP, ARP, ICMP, and UDP protocols.
- *misclib*: contains the files which implement a command-line interface (CLI) used for testing and monitoring the stack.
- *mtcp*: this files implement the TCP layer.
- *net*: this folder contains network support files (DHCP, DNS) and code for manage queues.
- *Obj*: all object files generated by Keil. The executable code is also generated into this folder.
- *tftp*: a TFTP client and server code.
- *vfs*: files needed for a File System implementation.

The Keil project files are located in the LPC\_30src directory.

## 4. Configuring the TCP/IP stack

### 4.1 Selecting the TCP/IP stack modules

In the *ipport.h* header file, the modules can be disabled/enabled by commenting/uncommenting the corresponding lines:

```
#define INCLUDE_ARP      1 /* use Ethernet ARP */
#define FULL_ICMP        1 /* use all ICMP || ping only */
#define OMIT_IPV4        1 /* not IPV4, use with MINI_IP */
#define MINI_IP          1 /* Use Nichelite mini-IP layer */
#define MINI_TCP         1 /* Use Nichelite mini-TCP layer */
#define MINI_PING        1 /* Build Light Weight Ping App for Niche Lite */
#define BSDISH_RECV      1 /* Include a BSD recv()-like routine with mini_tcp */
#define BSDISH_SEND      1 /* Include a BSD send()-like routine with mini_tcp */
#define NB_CONNECT       1 /* support Non-Blocking connects (TCP, PPP, et al) */
#define MUTE_WARNINGS    1 /* gen extra code to suppress compiler warnings */
#define IN_MENUS         1 /* support for InterNiche menu system */
#define NET_STATS        1 /* include statistics printf's */
#define QUEUE_CHECKING   1 /* include code to check critical queues */
#define INICHE_TASKS     1 /* InterNiche multitasking system */
#define MEM_BLOCKS       1 /* list memory heap stats */
#define TFTP_CLIENT      1 /* include TFTP client code */
#define TFTP_SERVER      1 /* include TFTP server code */
#define DNS_CLIENT       1 /* include DNS client code */
#define INICHE_TIMERS    1 /* Provide Interval timers */
#define DHCP_CLIENT      1 /* include DHCP client code */
//#define INCLUDE_NVPARMS 1 /* non-volatile (NV) parameters logic */
#define NPDEBUG          1 /* turn on debugging dprintf()s */
#define VFS_FILES        1 /* include Virtual File System */
#define USE_MEMDEV       1 /* Pseudo VFS files mem and null */
#define PRINTF_STDARG    1 /* build ...printf() using stdarg.h */
#define TK_STDIN_DEVICE  1 /* Include stdin (uart) console code */
#define BLOCKING_APPS    1 /* applications block rather than poll */
#define INCLUDE_TCP      1 /* this link will include NetPort TCP w/MIB */
```

Disabling unused modules helps minimize resource requirements.

## 4.2 Setting the MAC address

The MAC address is hard-coded in the **emac.c** file. The **eth\_info** structure is initialized with the default value.

```
static
struct eth_info eth_info[ETH_DEVICES] = {
    { MAC_BASE_ADDR, 0x00, 0x0f, 0xcc, 0x23, 0x00, 0x01, 0x00, 0x00, EMAC_INT } };
    // MAC address: 00:0F:CC:23:00:01
```

Fig 3. The **eth\_info** structure in the **emac.c** file contains the default MAC address

## 4.3 Configuring the IP address

In a TCP/IP Network, every station has to be configured with a unique and valid IP address. This configuration can be hard coded into the program (known as a static IP address), or it could be assigned dynamically using a specific protocol, such as DHCP (Dynamic Host Configuration Protocol), which requires a DHCP server attending the request from the DHCP client component running on the devices (known as a Dynamic IP address).

### 4.3.1 Static IP address configuration

In the following code snippet (located in the **pre\_task\_setup()** function in the **in\_stubs.c** file), the desired Static IP address is assigned to 192.168.1.120, when the DHCP option is disabled (the line **#define DHCP\_CLIENT 1** should be commented out).

```
#ifdef USE_EMAC
    /* Ethernet */
    #ifdef DHCP_CLIENT
        netstatic[0].n_flags |= NF_DHCP;
        netstatic[i].n_ipaddr = 0x00000000; /* 0.0.0.0 */
    #else
        netstatic[i].n_ipaddr = 0x7801A8C0; /* 192.168.1.120 */
    #endif
    netstatic[i].smmask = 0x00FFFFFF; /* 255.255.255.0 */
    netstatic[i].n_defgw = 0x0101A8C0; /* 192.168.1.1 */
    i++;
#endif
```

Fig 4. Code snippet that configures the IP address

Please be aware that this code is executed when the non-volatile (NV) parameters are not taken into account, so this option is disabled (line **#define INCLUDE\_NVPARMS 1** should be commented out).

### 4.3.2 Dynamic IP address configuration

If a Dynamic IP address configuration is needed, then the DHCP option needs to be enabled (uncomment the **#define DHCP\_CLIENT 1** line, if necessary). Under these conditions, the IP address is assigned to 0.0.0.0 (see [Fig 4](#)), but using DHCP protocol, a new IP address will be assigned from a DHCP server (a leased IP address). Of course, a DHCP server should be available from the LAN.

### 4.3.3 Dynamic IP address configuration with a default IP address

In some cases where Dynamic IP address is configured, it would be desirable to have a default IP address assigned just in case a DHCP server is temporary unavailable, so the device can continue working on the network.

For such particular cases, configure the previous code as [Fig 5](#) shows.

```
#ifdef USE_EMAC
    /* Ethernet */
#ifdef DHCP_CLIENT
    netstatic[0].n_flags |= NF_DHCP;
    netstatic[i].n_ipaddr = 0x7801A8C0; /* 192.168.1.120 */
#else
    netstatic[i].n_ipaddr = 0x7801A8C0; /* 192.168.1.120 */
#endif
    netstatic[i].snmask = 0x00FFFFFF; /* 255.255.255.0 */
    netstatic[i].n_defgw = 0x0101A8C0; /* 192.168.1.1 */
    i++;
#endif
```

Fig 5. Dynamic IP address configuration with a default IP address

## 5. Testing the TCP/IP stack

Once the stack configuration steps are completed, some tests can be performed.

### 5.1 Initial test

In order to use the PC for the test, the IP address must be configured accordingly. Thus, the PC's IP address should be in the same sub-network as the board. For example; if the board is configured with a static IP address such 192.168.1.120, and subnet mask 255.255.255.0, then the PC could have an IP address like 192.168.1.100 (or any other IP address not used in the same sub-network). In case the stack is configured using DHCP, then the same DHCP server could be used to assign a dynamic IP address to the PC.

Open HyperTerminal using the PC's COM port where the serial cable is connected to, and configure it as 115200,N,8,1.

Start Keil uVision3 and open the NicheLite software project. Build the project (if necessary) and start a Debug session. Once the code is loaded, pressing the Run button starts code execution.

[Fig 6](#) shows a screenshot of the NicheLite system's console after initialization. In this case, a default IP address is configured and DHCP is enabled. After the Ethernet module initialization, the address 192.168.1.120 is assigned to the interface. However, after DHCP successfully acquires a dynamic IP address, the new leased address 192.168.1.102 is reassigned to this interface.

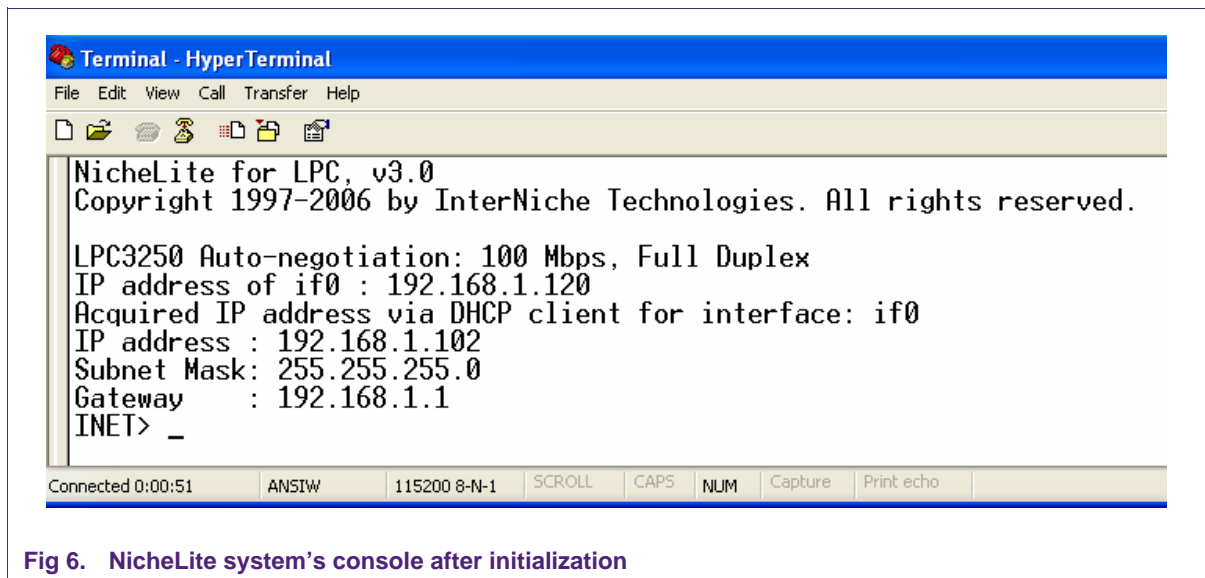


Fig 6. NicheLite system's console after initialization

After dynamic IP assignment, the CLI (command-line interface) prompt appears. General commands and diagnostic commands can be used from this prompt. For more details, refer to [Section 5.5](#).

## 5.2 Testing network connectivity

While the stack runs, you can test the network connectivity using the ping utility. Use Windows command-line interface (cmd.exe program) for this.

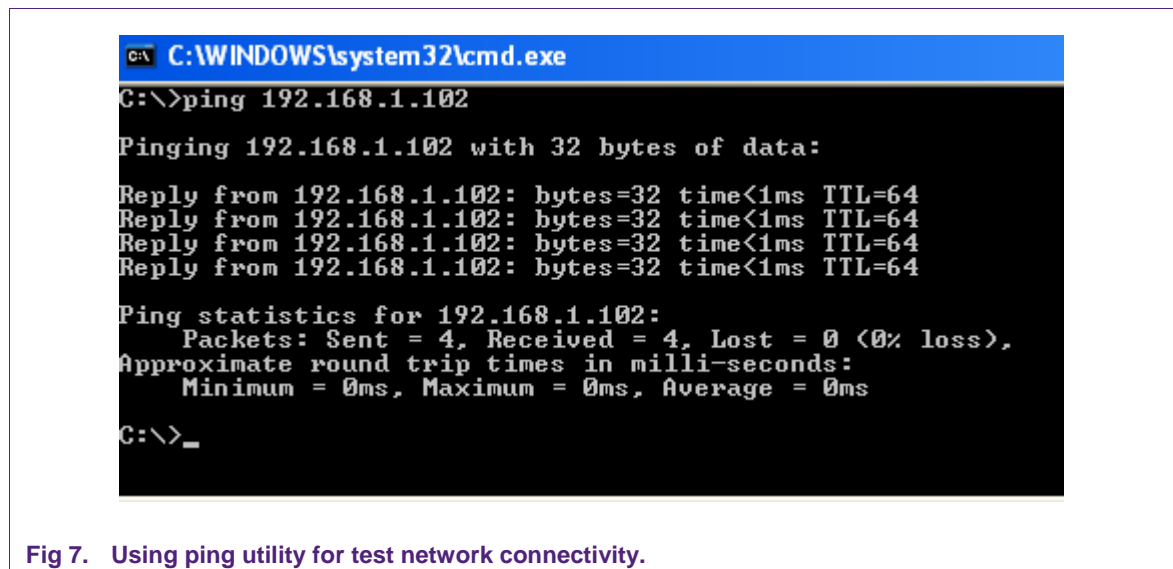


Fig 7. Using ping utility for test network connectivity.

If there are no replies from the board, check the network cables, or any network device such as a router, switch or hub. In general, if DHCP is used and a dynamic IP address is obtained, the network is okay, and the problem may be something like a network cable being disconnected.

If the hardware is checked, and still not working, check if the stack is running, verified by using the CL interface ([Section 5.5](#)). This can be used last, in order to issue a ping command to the PC.

### 5.3 Testing HTTP

NicheLite includes an HTTP listener, which acts as a very simple embedded web server. It offers a basic functionality consisting on detect an incoming HTTP request, and no matter what file is requested, it will always respond with the same web page, which can be seen in [Fig 8](#).

In order to test it, open a web browser and enter the board address (192.168.1.102 in this case), and the HTTP listener will respond with the web page as shown in [Fig 8](#).

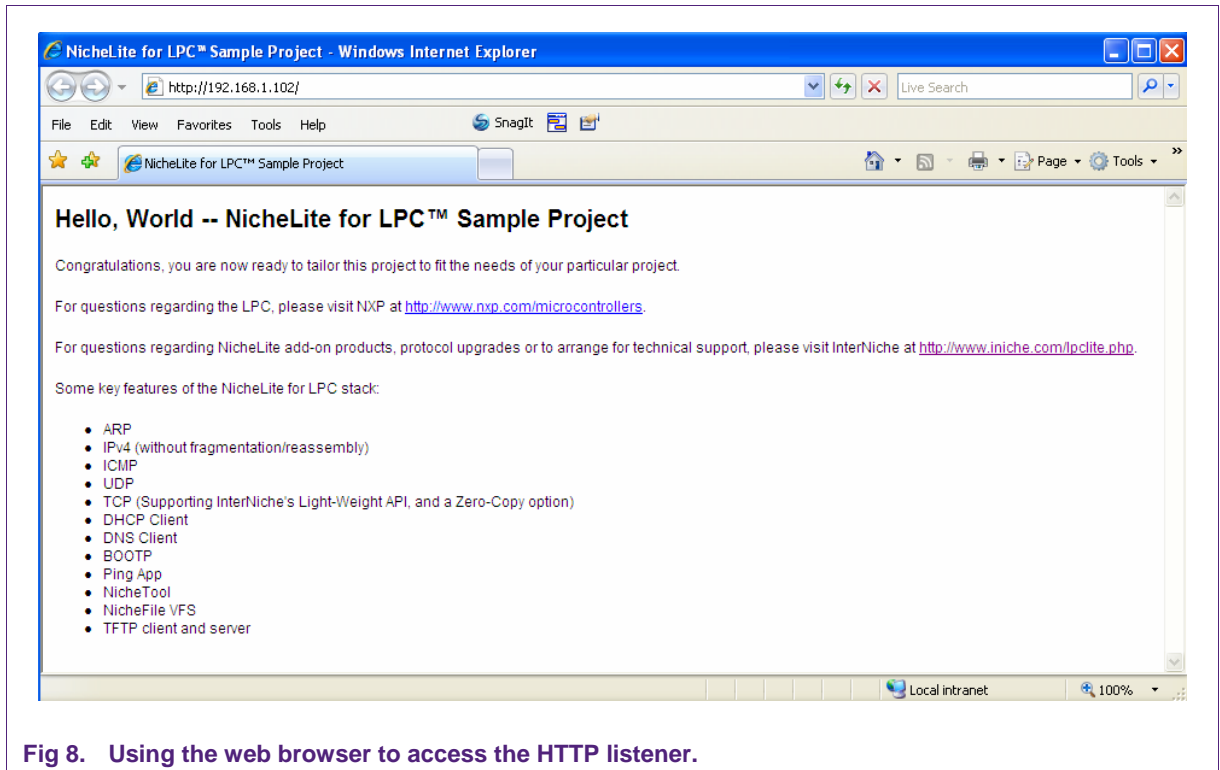


Fig 8. Using the web browser to access the HTTP listener.

### 5.4 Testing TFTP server

NicheLite includes both TFTP server and TFTP client components. This section will show how to test the server functionality. For help regarding the client component, please refer to NicheLite documentation.

The first step in the test is to start the TFTP server, using the **tfsrv** command from the CLI interface. See [Fig 9](#). (The same command is used to stop the server)

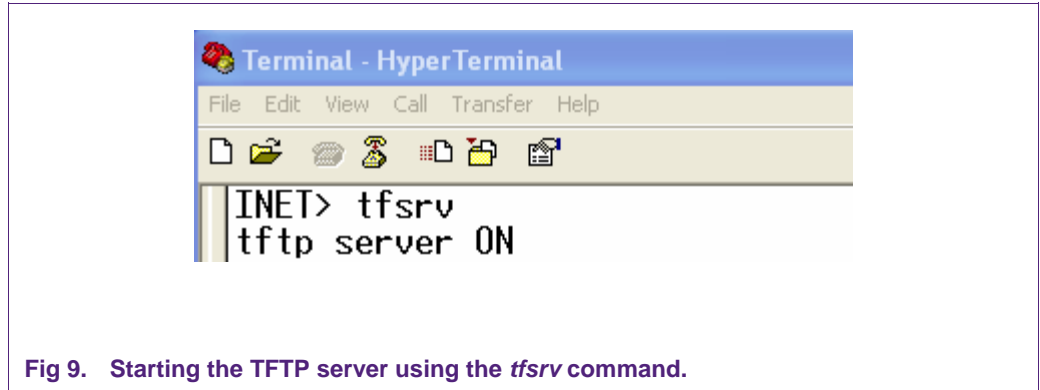


Fig 9. Starting the TFTP server using the *tfsrv* command.

Next, issue a TFTP command from Windows command-line interface, using the following syntax:

```
tftp 192.168.1.102 get mem1M
```

where *mem1M* is an existing file in NicheLite files system. [Fig 10](#) shows the results.

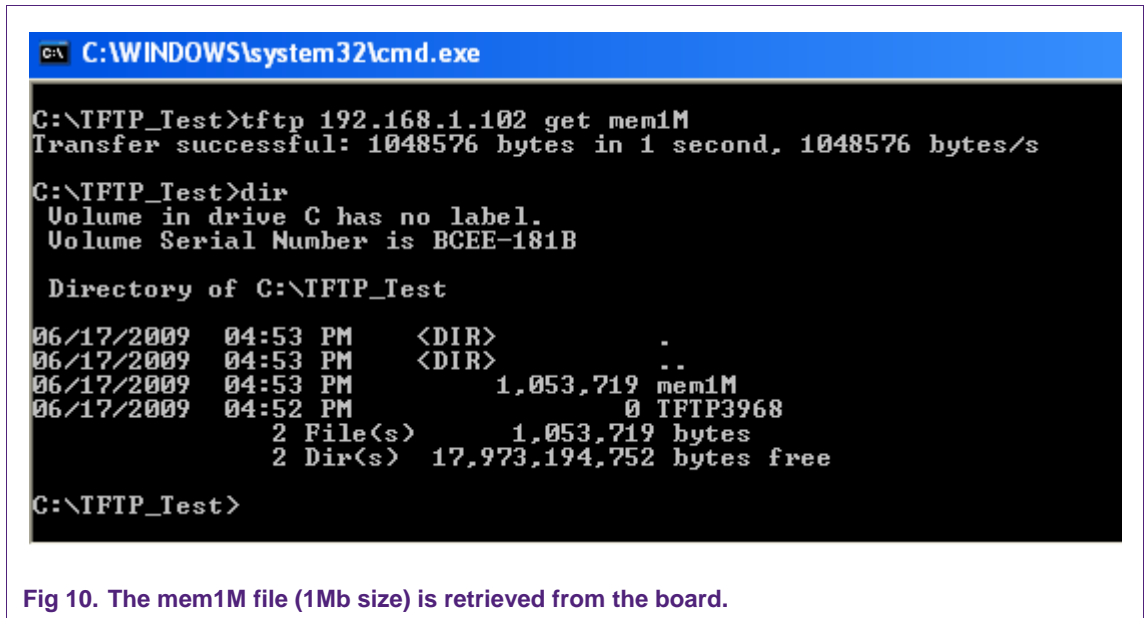


Fig 10. The mem1M file (1Mb size) is retrieved from the board.

The status of the TFTP transfer is shown in the system console (see [Fig 11](#)).

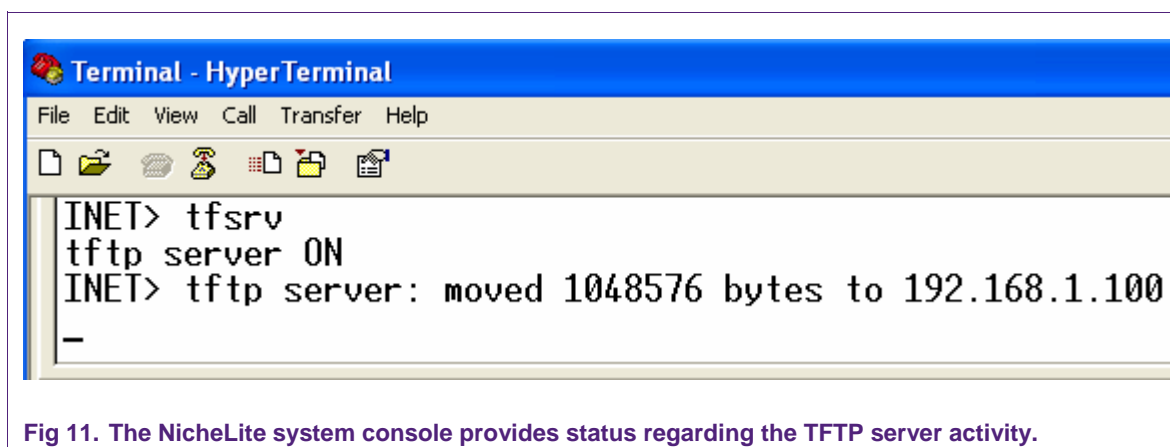


Fig 11. The NicheLite system console provides status regarding the TFTP server activity.

In order to upload a local file from the PC to the device, use the following syntax:

***tftp 192.168.1.102 put filename***

Then use the following command in order to retrieve it from the board, to check if it was successfully saved in the board's TFTP server.

***tftp 192.168.1.102 get filename***

Before using the above command, delete (or rename) the original local file, to be sure the file is retrieved from the server.

The ***vfswfilelist*** command from the system console may also be used in order to get a list of the file system, and verify that the new uploaded file is there.

The Windows command-line interface and the NicheLite system console both provide status of the commands execution.

## 5.5 Using the command-line interface

When the stack is running, the INET> prompt is shown in the system console. Use this prompt in order to interact with the stack. This will allow the ability to Monitor, Debug, or even Configure the stack at runtime.

### 5.5.1 General commands

Type the word ***help*** or ***?*** (question mark) to display a list of available commands.

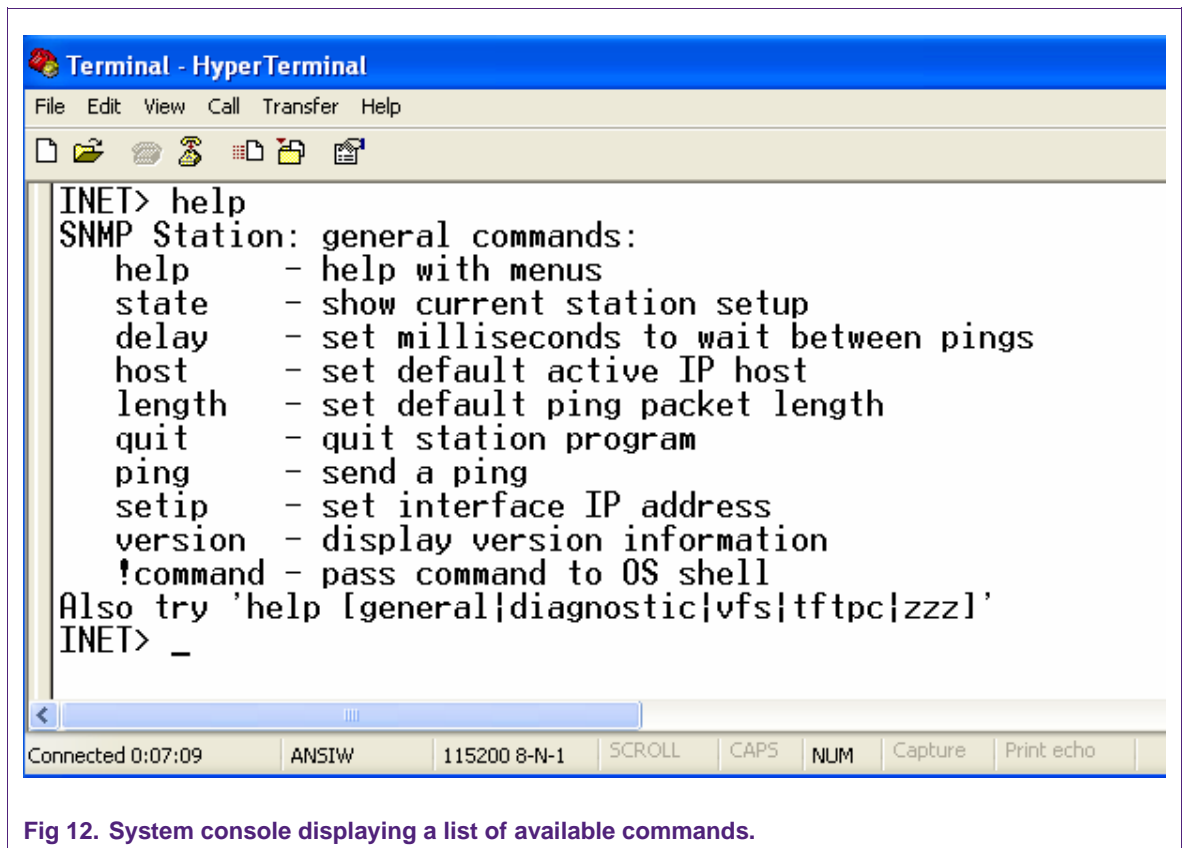


Fig 12. System console displaying a list of available commands.

As an example, you can try the **state** command in order to get the current device setup.

```
INET> state
iface 0-if0 IP addr:192.168.1.102 subnet:255.255.255.0 gateway:192.168.1.1
current tick count 1959334
Task wakeups:netmain: 385211318
nettick: 978109
keyboard: 979085
webport: 385211320
```

The **host** command allows you to specify the remote host IP address.

```
INET> host 192.168.1.100
```

Then, use the **ping** command to check if the host is reachable or not.

```
INET> ping
```

ping sent, check icmp for reply.

Use the *icmp* command (described in [Section 5.5.3](#)) to check the reply.

### 5.5.2 Diagnostic commands

*help diagnostic* can be used in order to display available commands; some are for diagnostic purposes, while others are for statistic information (described in the next section).

The *buffers* command displays allocated packet buffer information.

```
INET> buffers
PACKET len buffer que data offset 0
0801B958,1536,08020004,big:00 00 FF FF FF FF FF FF 00 1C 23 0D .....#.
0801B998,1536,0802060C,non:00 00 01 00 5E 00 00 FB 00 1C 23 0D ....^.....#.
0801B9D8,1536,08020C14,big:00 00 FF FF FF FF FF FF 00 1D 7E E0 .....~.
0801BA18,1536,0802121C,non:00 00 FF FF FF FF FF FF 00 1D 7E E0 .....~.
0801BA58,1536,08021824,big:00 00 00 1D 7E E0 A2 D2 00 0F CC 23 ....~.....#
0801BA98,1536,08021E2C,big:00 00 00 0F CC 23 00 01 00 1D 7E E0 .....#.....~.
```

It is possible to dump a block of memory, using the *dbytes* command.

```
INET> dbytes 0x0 100
18 F0 9F E5 18 F0 9F E5 18 F0 9F E5 18 F0 9F E5 .....
18 F0 9F E5 18 F0 9F E5 18 F0 9F E5 18 F0 9F E5 .....
88 00 00 08 40 00 00 08 88 5A 01 08 48 00 00 08 ....@....Z..H...
4C 00 00 08 44 00 00 08 B4 5A 01 08 38 5D 01 08 L...D...Z..8]..
FE FF FF EA FE FF FF EA FE FF FF EA FE FF FF EA .....
0E 00 A0 E1 13 F0 21 E3 7C D0 9F E5 12 F0 21 E3 .....!|.....!
78 D0 9F E5 11 F0 21 E3 74 D0 9F E5 17 F0 21 E3 x.....!t.....!
70 D0 9F E5 1B F0 21 E3 6C D0 9F E5 1F F0 21 E3 p.....!l.....!
68 D0 9F E5 00 F0 A0 E1 64 00 9F E5 01 10 A0 E3 h.....d.....
00 10 80 E5 00 00 A0 E3 10 0F 01 EE 17 0F 07 EE .....
17 0F 08 EE E9 FF FF EB 40 56 00 EB 13 00 00 EA .....@V.....
00 10 0F E1 00 10 80 E5 80 10 81 E3 01 F0 2F E1 ...../..
0E F0 A0 E1 00 10 90 E5 01 F0 2F E1 0E F0 A0 E1 ...../.....
20 00 9F E5 14 10 9F E5 0E F0 A0 E1 00 ED 03 08 .....
00 F1 03 08 00 F2 03 08 00 F3 03 08 00 F4 03 08 .....
00 EC 03 08 14 40 00 40 6C AC 01 08 00 00 00 00 .....@.@|.....
```

The **linkstats** command displays information for the hardware associated with the provided interface.

```
INET> linkstats 1
Interrupts: rx:109, tx:23 total:132
coll1:0 collx:0 overrun:0
Sendq max:1, current 0. Int enable B9, pending 40
Rx Descriptors and Status:
  Next In: 1, Next Out: 1
  08023480: 08022436 800005FD  08023498: C420003F 00FF014D
  08023488: 08020C16 800005FD  080234A0: C420003F 007E014D
  08023490: 08022A3E 800005FD  080234A8: C460005D 01FE019F
Tx Descriptors and Status:
  Next In: 1, Next Out: 1
  080234BC: 080231DE C000002D  080234CC: 00000000
  080234C4: 08021826 C0000156  080234D0: 00000000
```

### 5.5.3 Statistic commands

The following commands can be used to display statistics related to their protocol layer:

```
INET> arp
arp Requests In: 6, out: 3
arp Replys In: 3, out: 6
X) MAC Address iface pend IP ctime ltime
7) 001D7E-E0A2D2 1 N 192.168.1.1 867052 867052
INET> ip
IP MIB statistics:
Gateway: YES default TTL: 64
rcv: total: 96 header err: 0 address err: 54
rcv: unknown Proctcls: 0 delivered: 42
send: total: 14 discarded: 0 No routes: 0

INET> icmp
ICMP layer stats:
icmpInMsgs 8 icmpInErrors 0, echoReqs 0, echoReps 8, unhandledTypes: 0
```

```
icmpOutMsgs 3  icmpOutErrors 0,
```

```
INET> udp
```

```
UDP MIB dump:
```

```
In: Good: 6  No Port: 0  Bad: 31
```

```
Out: 11
```

```
INET> tcp
```

```
tcpRtoAlgorithm 0,  tcpRtoMin 0
```

```
tcpRtoMax 0,  tcpMaxConn 0
```

```
tcpActiveOpens 0,  tcpPassiveOpens 0
```

```
tcpAttemptFails 0,  tcpEstabResets 0
```

```
tcpCurrEstab 0, tcpInSegs 0
```

```
tcpOutSegs 0,  tcpRetransSegs 0
```

```
tcpInErrs 0,  tcpOutRsts 0
```

## 6. Legal information

### 6.1 Definitions

**Draft** — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

### 6.2 Disclaimers

**General** — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

**Right to make changes** — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

**Suitability for use** — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of a NXP Semiconductors product can reasonably be expected

to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is for the customer's own risk.

**Applications** — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

**Export control** — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from national authorities.

### 6.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

## 7. Contents

---

<b>1.</b>	<b>Introduction .....</b>	<b>3</b>
<b>2.</b>	<b>Connecting and configuring the PHY3250 board .....</b>	<b>3</b>
<b>3.</b>	<b>Installing the software .....</b>	<b>4</b>
<b>4.</b>	<b>Configuring the TCP/IP stack .....</b>	<b>5</b>
4.1	Selecting the TCP/IP stack modules .....	5
4.2	Setting the MAC address .....	6
4.3	Configuring the IP address .....	6
4.3.1	Static IP address configuration .....	6
4.3.2	Dynamic IP address configuration .....	6
4.3.3	Dynamic IP address configuration with a default IP address .....	7
<b>5.</b>	<b>Testing the TCP/IP stack .....</b>	<b>7</b>
5.1	Initial test .....	7
5.2	Testing network connectivity .....	8
5.3	Testing HTTP .....	9
5.4	Testing TFTP server .....	9
5.5	Using the command-line interface .....	11
5.5.1	General commands .....	11
5.5.2	Diagnostic Commands .....	13
5.5.3	Statistic commands .....	14
<b>6.</b>	<b>Legal information .....</b>	<b>16</b>
6.1	Definitions .....	16
6.2	Disclaimers .....	16
6.3	Trademarks .....	16
<b>7.</b>	<b>Contents .....</b>	<b>17</b>

---

Please be aware that important notices concerning this document and the product(s) described herein, have been included in the section 'Legal information'.

---

© NXP B.V. 2009. All rights reserved.

For more information, please visit: <http://www.nxp.com>  
 For sales office addresses, email to: [salesaddresses@nxp.com](mailto:salesaddresses@nxp.com)

