



XAPP1043 (v1.0) October 9, 2008

Measuring Treck TCP/IP Performance Using the XPS LocalLink TEMAC in an Embedded Processor System

Author: Doug Gibbs

Abstract

This application note illustrates how to measure the network performance of the XPS Local-Link Tri Mode Ethernet MAC (TEMAC) in an embedded processor system running the Treck TCP/IP stack. The measurements use a PPC405 processor based system with the ML405 Evaluation Platform or a MicroBlaze™ based system with the ML505 Evaluation Platform. The test software and methods are provided so that similar tests can be performed using different boards.

This application note outlines how to acquire the hardware designs and set up the two boards. The set up of the Treck TCP/IP software is detailed along with how to integrate it with a standalone test application. Test software that can be run on a Windows or Linux PC is also discussed.

Included Systems

Included with this application note are two reference systems:

For the PowerPC 405 processor-based system, go to <https://secure.xilinx.com/webreg/clickthrough.do?cid=113294>.

The project name used in the xapp1043_ppc_405.zip file is ml405_ppc_xps_ll_temac.

For the MicroBlaze processor-based system, go to <https://secure.xilinx.com/webreg/clickthrough.do?cid=113285>.

The project name used in the xapp1043_mb_505.zip file is ml505_mb_xps_ll_temac.

System Specifics

The ML405 PPC405 system is configured as:

- PPC405 clock – 300 MHz
- MPMC / DDR clock – 100 MHz
- PLBv46 clock – 100 MHz
- XPS_LL_TEMAC Clock 100MHz

The ML505 MicroBlaze system is configured as:

- MicroBlaze clock – 125 MHz
- MPMC / DDR2 clock – 125 MHz
- PLBv46 clock – 125 MHz
- XPS_LL_TEMAC Clock 125 Mhz

More information on these systems is available in XAPP1041, XPS Local Link Tri-Mode Ethernet MAC Embedded Systems for MicroBlaze and PowerPC405.

Introduction

Xilinx offers the XPS_LL_TEMAC core in several FPGA families. In devices where hard TEMAC blocks are available, the XPS_LL_TEMAC core can be configured to leverage the hard TEMAC in order to save FPGA resources. In devices that do not offer hard TEMAC blocks, or where the existing TEMACs have already been utilized, the XPS_LL_TEMAC core can be configured to implement the entire function using generic FPGA resources. This application note provides a performance test method for testing using the Treck TCP/IP stack.

A section is provided on setting up the Treck software and including the TCP/IP stack with the test applications. The applications do not use an operating system. For the purpose of these tests, the applications run in the standalone mode. If further information is needed, see the “References” section at the end of this document.

The tests use a PC running either Windows or Linux. The Xilinx ML405 or ML505 board sends and receives Ethernet packets to and from the PC.

The “Results” section provides data from running these tests on the reference system.

Hardware and Software Requirements

The hardware and software requirements are:

- Xilinx ML405 Development Board for the PPC405 reference system
 - Xilinx ML505 Development Board for the MicroBlaze reference system
 - Xilinx Platform Studio 10.1i
 - ISE® 10.1i design tools
 - PC running Linux or Window and a Gigabit-capable Ethernet interface
 - iPerf installed on the PC. Available from <http://sourceforge.net/projects/iperf>
 - Cat5e or Cat6e Ethernet Cable and RS232 cable
-

XPS_LL_TEMAC Configuration

The XPS_LL_TEMAC core can use the Link Layer connection in either FIFO or Scatter Gather Direct Memory Access (SGDMA) modes. The systems in this application use the SGDMA mode.

The SgDMA is part of the Multi Port Memory Controller (MPMC) core. The XPS_LL_TEMAC uses a local link connection to move data directly to and from memory in conjunction with the SGDMA hardware. The scatter gather DMA allows data to be placed in memory without requiring a contiguous buffer. This allows the software to use a chain of buffers for an Ethernet packet. Using a chain of buffers eliminates costly software copy operations in the Ethernet drivers and network stack.

Treck TCP/IP Stack

Treck, Inc. and Xilinx have developed a partnership to deliver products that have been integrated and tested together. The Treck and Xilinx partnership solution offers high performance internet protocols for networking products. The Treck and Xilinx partner ship solution is able to achieve near line rates for Gigabit Ethernet using the performance tests described in this application note.

For the purposes of this application note, it is assumed the reader understands TCP/IP and Ethernet based networks. A thorough introduction is provided in the Treck TCP/IP User Manual, [truser40e.pdf](#), provided with the Treck protocol stack and available for download from the Treck website at <http://www.treck.com>.

This application note uses the Treck protocol stack to provide the following protocols: TCP, IP, ARP, and ICMP. UDP and other protocols are available, but not used in any of the examples.

The Treck stack is built as a library. This library is then linked with the software applications running on the Xilinx embedded processor. In the provided systems only a time-limited pre-compiled linkable library and associated include files are provided. The complete source is available from [Treck, Inc.](#)

Using Treck TCP/IP Examples

A number of examples are bundled into the software application. The software application source code is included in the `Treck/src` directory in the projects main directory.

The applications use the serial port UART to setup and run individual examples. The examples included are:

1. Idle application: allows the software to respond to ping requests.
2. TX Client Application
3. RX Server application

Some other utility functions are provided. Each of the utility menu items is described below:

- **Reset interrupt coalescing parameters:** Allows the user to set the number of packets that must be received before an interrupt is generated. The delay before an interrupt is generated can also be set. These settings can have a large impact on performance.
- **Set PHY address:** can be used to set a different PHY address. This is needed only if the software fails to correctly detect the PHY address.
- **Switch to 10, 100, or 1000 Mbps:** The link speed can be manually set using these menu choices.

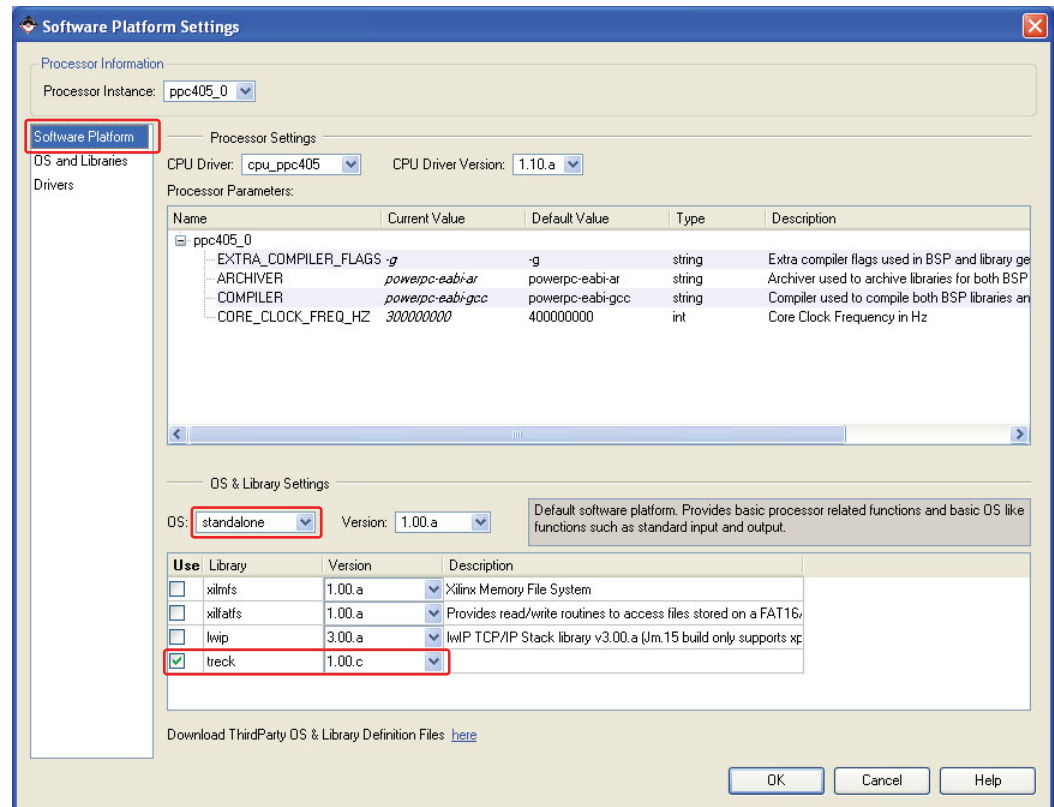
Test Setup

Build the Board Support Package including the Treck TCP/IP stack

The Treck TCP/IP software is built as a library and linked with the standalone application and compile time. The Xilinx Platform Studio (XPS) searches through specific directories to find libraries to build as part of the LibGen process. For XPS to find the Treck software and include it in the Software Platform settings dialog, it must be placed in the `sw_services` directory. The directory structure must be correct. See the UG111 Embedded Systems Tools Reference Manual for details on the directories used.

First, the Treck software must be in the `sw_services` sub directory under the main project directory. Prior to opening the project in XPS, verify that the Treck library is present in the `sw_services` sub directory. Start EDK and navigate to the **Software**→**Software Platform Settings** dialog window shown in [Figure 1](#).

1. Open the project in XPS and select **Software** **Software Platform Settings** to open the dialog window shown in [Figure 1](#).
2. In the Software Platform Settings dialog, under the Software Platform settings, verify that the selected OS is **standalone** and that the **treck** library is selected in the OS & Library settings. These settings will direct the tools to build the Treck library when the Board Support Package (BSP) for the target system is generated.
3. Click **OK**.
4. Build the libraries by selecting **Software** **Generate Libraries and BSPs**. Building the libraries may take a few minutes.
5. Copy the file from
`...\ml405_ppc_xps_ll_temac\sw_services\treck_v1_00_c\src\libtreck.a`
 to the libraries directory, `...\ml405_ppc_xps_ll_temac\ppc405_0\lib`



X1043_01_053008

Figure 1: Software Platform Settings Dialog

Build the Applications

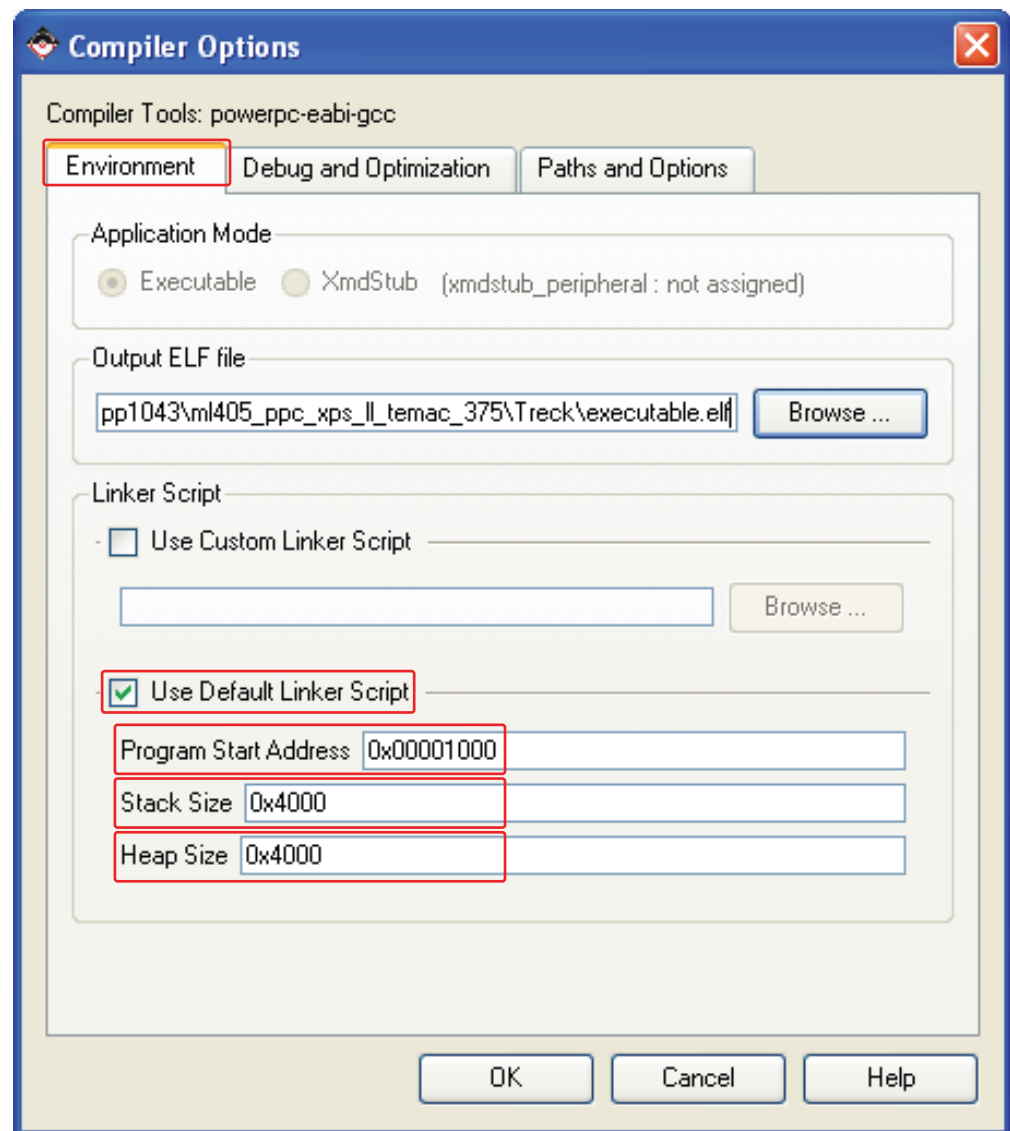
The application is included with the reference systems. The applications can be set up in a different project if needed.

To set up the Treck software applications in a different project,

1. Add a new application in EDK by double clicking the **Add Software Application Project...** in the Applications tab.
2. Assign an appropriate name to the project in the dialog box that pops up, then click **OK**.

3. Double click the **Sources** list item in the project that was just created. Navigate to where the Treck application source code is located. For the example system, the source code is in the `Treck/src` directory.
4. Select all `C` files and click **OK**.
5. Repeat this for the include files by clicking on the Headers, and selecting the `H` files.
6. Double click on the name of the project to display the Compiler Options dialog shown in [Figure 2](#). The Default Linker script will work, but extra stack and heap space are needed. In the Environment tab, set the Stack Size to `0x4000` and the Heap Size to `0x4000`.
7. Because the memory maps of the two systems are different, the program start addresses are different as shown below.
 - a. For the ML405 system set the start address to `0x1000`.
 - a. For the ML505 system set the start address to `0x9a000000`.

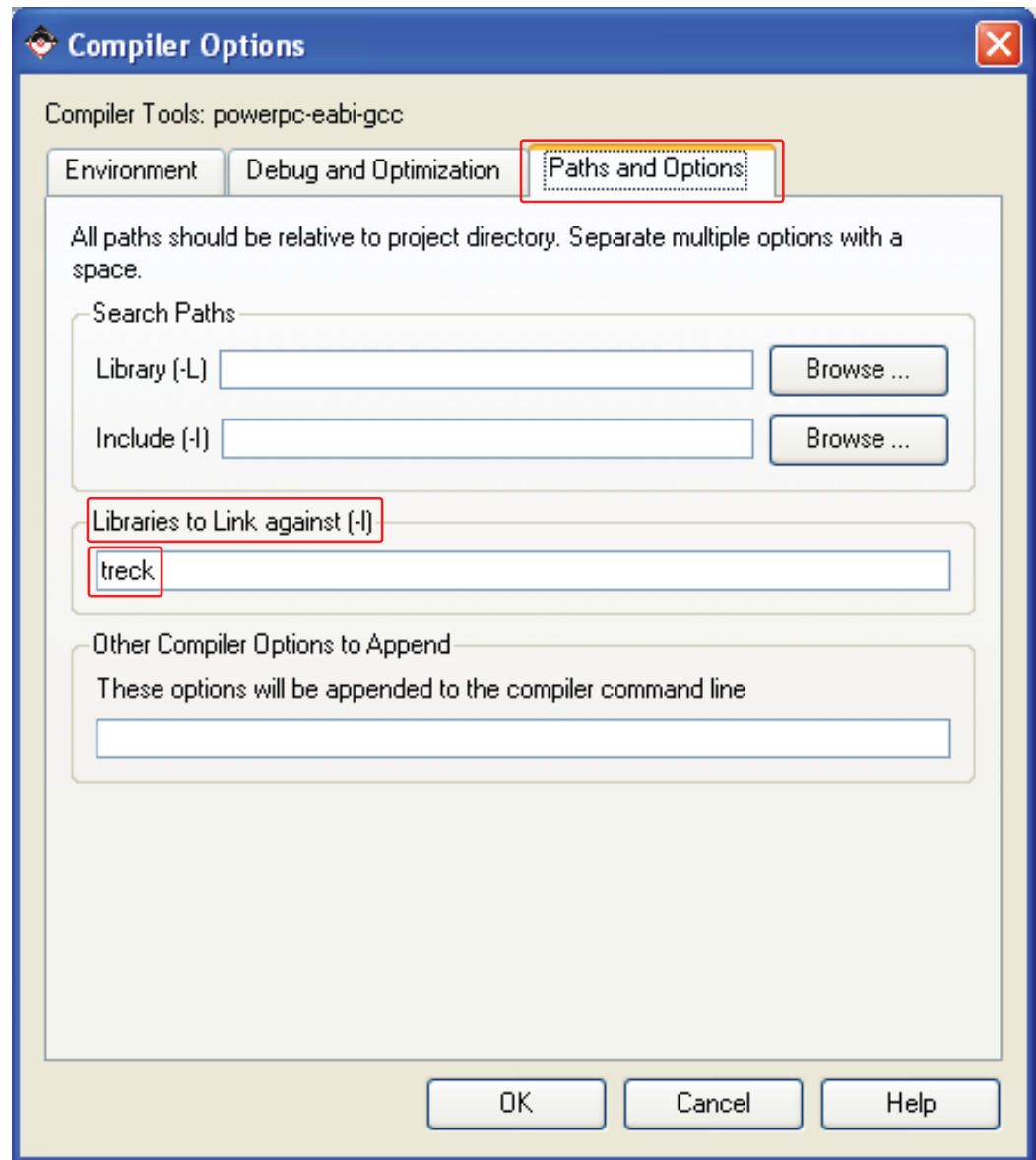
The `xps.h` include file sets up system constants generated when LibGen is run to match constants used in the code, so that the software will build for both systems. The file may need to be modified for the specific system.



X1043_02_053008

Figure 2: Compiler Options Dialog

8. In the Compiler Options window, in the Libraries to Link against section of the Paths and Options tab, specify the **treck** library as shown in Figure 3. This instructs the tool chain to include the `treck` library when compiling the software application.



X1043_03_053008

Figure 3: Compiler Options for Setting Up Libraries

9. Build the project by right clicking the name of the project and selecting `Build Project`. The options specified in the Compiler Options window will be used to generate an executable.elf file which will run on the Xilinx embedded processor.

Executing the Reference System

The bitstream(s) for the system are available in the `ready_for_download/` directory under the project root directory.

For reference systems details, see [XAPP1041 Reference System: XPS Local Link Tri-Mode Ethernet MAC Embedded Systems for MicroBlaze and PowerPC 405](#).

Executing the Reference System Using the Pre-Built Bitstream and the Compiled Software Applications

To execute the system using files in the `ready_for_download/` directory in the project root directory, follow these steps:

1. Launch a bash shell by selecting **Project Launch** → **EDK Shell**.
2. Change directories to the `ready_for_download` directory.
3. Use iMPACT to download the bitstream by using the following command:
`impact -batch xapp1043.cmd.`
4. Invoke XMD and connect to the processor by using the following command:
`xmd -opt xapp1043.opt.`
5. Download the executable by using the following command: `dow executable.elf.`
6. Change directories to `../treck.`
7. Start a HyperTerminal session with the following settings: **115200**, **8**, **n**, and **1**.
8. Run the executable by using the following command: `run`

Executing the Reference System From EDK

To execute the system using EDK, follow these steps:

1. Open `system.xmp` in XPS.
2. Select **Hardware** → **Generate Bitstream** to generate a bitstream for the system.
3. **Device Configuration** → **Download Bitstream** to download the bitstream to the board.
4. Launch **XMD with Debug** → **Launch XMD**.
5. Download the executables by using the following command: `dow executable.elf.`
6. Start a HyperTerminal session with the following settings: **115200**, **8**, **n**, and **1**.
7. Run the executable by using the following command: `run.`

Optimizing the PC network connection

Windows XP

The Windows PC requires a Gigabit-capable Ethernet card. The drivers and card must be able to send and receive jumbo frames.

To enable large frames in the TCP/IP stack in Windows XP, Windows 2000, and Windows Server 2003, use the following steps:

1. Edit the registry via `regedit.exe` and navigate to
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters.`
2. Increase the TCP/IP window size to 132k and enable rfc1323 scaling and timestamps:
 - a. Add a registry DWORD named `TcpWindowSize`, then enter `131400`.
 - b. Add a registry DWORD named `Tcp1323Opts`, then set the value to `3`.
3. Increase the TCP buffer sizes:
 - a. Add a registry DWORD named `ForwardBufferMemory`, then set the value to `80000`.
 - b. Add a registry DWORD named `NumForwardPackets`, then set the value to `60000`.

Linux

The information is an excerpt from a DataTag article found at:
<http://datatag.web.cern.ch/datatag/howto/tcp.html>.

KERNEL CONFIGURATION

ftxqueue length high

There are settings available to regulate the size of the queue between the kernel network subsystems and the driver for network interface card. There are two queues to consider, the txqueuelen, which is related to the transmit queue size, and the netdev_backlog, which determines the recv queue size.

The queue size can be set manually by the user by using by using the following ifconfig command on the required device: `/sbin/ifconfig ethX txqueuelen 2000`

On a network with a rtt of 120ms and at Gig rates, a txqueuelen of at least 10000 is recommended. Where ethX is the Ethernet interface connected to the board (eth0, eth1, and so forth).

kernel receiver backlog

For the receiver side, there is a similar queue for incoming packets. This value by can be set by using the command: `/sbin/sysctl -w sys.net.core.netdev_max_backlog=2000`.

SACKs and Nagle

SACKs (Selective Acknowledgments) are an optimization to TCP. In Gigabit networks with no traffic competition to improve performance, SACKs should be turned off by using the command: `/sbin/sysctl -w net.ipv4.tcp_sack=0`.

However, Nagle algorithm, the default value, should be turned on. The user can determine if the program has Nagle switched off, by checking if it sets the TCP_NODELAY socket option.

SOCKET BUFFERS

TCP uses what it calls the "congestion window" or CWND [4] to determine how many packets can be sent at one time. The maximum congestion window is related to the amount of buffer space that the kernel allocates for each socket. For each socket, there is a default value for the buffer size, which can be changed by the program using a system library call just before opening the socket.

To change the buffer socket size with iperf, use the -W option.

When building an application, use the appropriate "set socket option" system call.

If the program requests more socket buffer memory than the kernel is configured to provide, the program will not receive it. The maximum value of socket buffer memory for the system can be adjusted by using the command: `/sbin/sysctl -w net.core.rmem_max= VALUE`, where VALUE should be enough for the socket buffer size.

In addition, the user should also set the "write" value in the sender and the receiver by using the following commands:

```
/sbin/sysctl -w net.core.wmem_max= VALUE and  
/sbin/sysctl -w net.ipv4.tcp_mem= MIN DEFAULT MAX
```

Test Flow

The console output shown in [Figure 4](#) is displayed when the software starts. Accept the defaults for all tests.

The window also shows the Settings Menu. The default settings can be changed using this menu.

Basic function

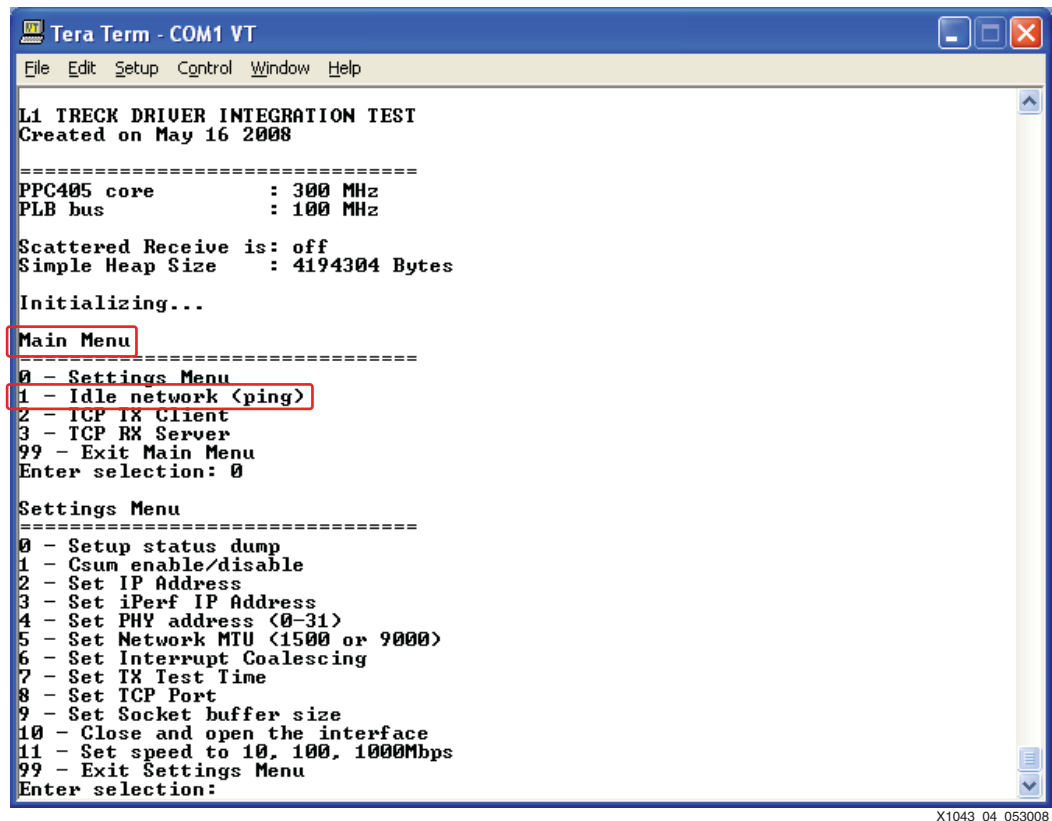
The system will not send or receive packets unless one of the applications is running.

1. To start the Treck idle application, choose menu item 1, **Idle network (ping)**, under Main Menu as shown in [Figure 4](#).
2. Press **Enter**. This starts the Treck TCP/IP stack.

The default IP address is 192.168.2.20. When the Treck idle application is running, the board will respond to ping requests. If the board does not respond to a ping, check the network connections and confirm that the network status LEDs on the board match the network settings.

The test program used for testing is iPerf, which is an open source network performance benchmarking application. The iPerf program opens a network socket, then either sends or receives TCP packets as fast as possible. The program attempts to send the maximum size payloads in the TCP data. See the iPerf help and documentation for a full explanation of the program options.

Note: These tests attempt to find the maximum network performance. The tests should be run on an isolated network. Sending large amounts of network traffic on a corporate network is not recommended.



```

Tera Term - COM1 VT
File Edit Setup Control Window Help

L1 TRECK DRIVER INTEGRATION TEST
Created on May 16 2008

=====
PPC405 core      : 300 MHz
PLB bus         : 100 MHz

Scattered Receive is: off
Simple Heap Size : 4194304 Bytes

Initializing...

Main Menu
=====
0 - Settings Menu
1 - Idle network (ping)
2 - ICP TX Client
3 - ICP RX Server
99 - Exit Main Menu
Enter selection: 0

Settings Menu
=====
0 - Setup status dump
1 - Csum enable/disable
2 - Set IP Address
3 - Set iPerf IP Address
4 - Set PHY address (0-31)
5 - Set Network MTU (1500 or 9000)
6 - Set Interrupt Coalescing
7 - Set TX Test Time
8 - Set TCP Port
9 - Set Socket buffer size
10 - Close and open the interface
11 - Set speed to 10, 100, 1000Mbps
99 - Exit Settings Menu
Enter selection:

```

X1043_04_053008

Figure 4: Treck Driver Integration Test Values

Transmit Tests

The Trek Tx Client application performs a basic transmit test. The test will connect using TCP to a server running on a PC. The test then sends data as fast as possible. The program running the server on the PC outputs the total TCP throughput once per second.

To run the test:

1. Run iPerf from a command line using the following command: `iperf -s -i 5`.
2. Select menu item #2 in the terminal program.

Enter selection: 2

Running TCP TX Client
Sending 261000 bytes at a time

3. The application will then send a TCP connect request to the IP address of the PC running iPerf. The PC application should start to output the total throughput every five seconds. The throughput will vary slightly. By default the client sends for 100 seconds.

The settings can be optimized using iPerf: `iperf -l 8942 -w 1048576 -M 8942 -s`. The results of the test will be displayed when the client disconnects.

Receive tests

The Trek Rx Server application performs a basic receive test. The test waits for a TCP connection from a client running on a PC. The client then sends data as fast as possible. The program running the client on the PC outputs the total TCP throughput once per second.

1. Select menu item #3 in the terminal program. The application will ask some setup questions:

Enter selection: 3

Running TCP RX Server
Setting recv sock size to 1048576 bytes
Waiting for new connection

2. Start the client on the PC. Run iPerf from a command line using the following command: `iperf -t 100 -c 192.168.2.20 -i 5`.
3. The application will then send a TCP connect request to the IP address of the host. The PC application should start to output the total throughput second. The throughput will vary.

The settings can be optimized using iPerf: `iperf -l 71536 -w 262144 -M 8942 -c 192.168.20 -t 100`. The results of the test will be displayed when the client disconnects.

Setting Defaults

Most of the program settings can be changed using the Settings Menu. The application software can be modified to make changes permanent so they do not have to be manually changed each time that the program is run.

The `defaults.h` file can be modified to set all default parameters.

Other program defaults can be set in `defaults.h` and `xtr_lltemac.h`. The conditional compiling of various options should only be done after careful study of the code.

Results

The results shown in [Table 1](#) are based on testing using iPerf version 2.4 with Red Hat Linux. The system used an Intel Pro PCIe Ethernet card. The 375 Mhz system is an optimized PPC405 system that is not included with this application note.

Table 1: Testing Using the PPC405 and MicroBlaze Systems on the MLxxx Boards

Client/Server	MTU 1500	MTU 9000
300 MHz PPC405 System on the ML405 Board		
TX Client	213 Mbps	922 Mbps
RX Server	115 Mbps	785 Mbps
375 MHz PPC405 System on the ML405 Board		
TX Client	233 Mbps	990 Mbps
RX Server	172 MBps	842 Mbps
125 MHz MicroBlaze System on the ML505 Board		
TX Client	101 Mbps	531 Mbps
RX Server	77.1 MBps	434 Mbps

Conclusion

Using the methodology provided, performance measurements are run for the XPS LL_TEMAC using the PPC 405 and MicroBlaze embedded processors and the Treck TCP/IP software stack. The network throughput for a TCP connection can be measured for different network MTU settings and interrupt coalescing parameters.

Using this approach, performance of the XPS_LL_TEMAC can be estimated for a system using the Xilinx embedded processors and the Treck TCP/IP software stack.

References

1. [XAPP1041](#) *Reference System: XPS LL Tri-Mode Ethernet MAC Embedded Systems for MicroBlaze and PowerPC 405*
2. [UG111](#) *Embedded System Tool Guide*

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/9/08	1.0	Initial Xilinx release.

Notice of Disclaimer

Xilinx is disclosing this Application Note to you “AS-IS” with no warranty of any kind. This Application Note is one possible implementation of this feature, application, or standard, and is subject to change without further notice from Xilinx. You are responsible for obtaining any rights you may require in connection with your use or implementation of this Application Note. XILINX MAKES NO REPRESENTATIONS OR WARRANTIES, WHETHER EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL XILINX BE LIABLE FOR ANY LOSS OF DATA, LOST PROFITS, OR FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR INDIRECT DAMAGES ARISING FROM YOUR USE OF THIS APPLICATION NOTE.