**iperf Demo**

Iperf is a networking tool that helps to measure networking bandwidth and performance. The iperf demo application can act as both a client and server for testing. iperf has the ability to test both UDP and TCP. In the case of UDP, you can specify the size of the UDP datagrams. For TCP, iperf measures the throughput of the payload.

In order to run iperf, you'll need a PC that has an iperf application on it as well. There is an open source version that is maintained, as well as many other variants across the internet. iperf is meant to be run at the command line. However, if a GUI is desired, a variant called jperf can be used. On PIC side, there is also a built-in command console that can accept your different iperf settings from system console, like UART.

In the case of the demo application, iperf measures performance data in a unidirectional format. Therefore, the side that the server is running on is considered the receiver, and provides the most accurate performance values.

Command Synopsis

|  |  |  |
| --- | --- | --- |
| iperf | [ -s|-c <IP addr> ] [ -u ] [ -i <sec> ] [ -b <bandwidth> ] [ -t <time> ] | |
| -s | | Runs the iperf server. No IP address needs to be pecified. | |
| -c <IP addr> | | Runs the iperf client. The IP address is the IP address of the server. | |
| -u | | Server side only. Sends UDP datagrams. | |
| -i <sec> | | Specified the time interval, in seconds, that the display will be updated. | |
| -b <bandwidth> | | Specifies the amount of data to try and send. This option is only valid with UDP datagrams. | |
| -t <time> | | Amount of time, in seconds, to run the test. | |

***Note:***

The socket size of TCP/UDP(especially TCP) will affect the benchmark result a lot. And activated TCPIP modules will also consume CPU & Ethernet load. Sure the traffic load in your test network environment will also affect the benchmark test.

So, in order to get a reasonable maximum benchmark/evaluation data for PIC target, you’d better disable other modules, like HTTP, client example, server example, etc… and find a not-congested network.

**Running the Demo**

First, make sure iperf\_app.c& iperf\_console.c are added into your project, and make sure that the symbol “APP\_USE\_IPERF” is defined either in global project settings or in visible header file (like main\_demo.h). Then rebuild and program your project into target board.

After powering on the development board and associating network connected, you'll need to start the server side iperf application first. if you start iperf as a server on the PIC development board in the console, then this implies that you are trying to measure the PIC Ethernet receiver performance. If you start the iperf server on a PC, then you will be measuring PIC Ethernet transmit performance.

**Take PIC32 ESK for example:**

Below tests will show receiver and transmitter performance respectively on PIC32 ESK with the following settings(Note, below data does NOT show the maximum throughput of PIC32 because following settings are not optimized for that):

Only

TCPIP\_STACK\_USE\_DHCP\_CLIENT,

TCPIP\_STACK\_USE\_TCP, TCPIP\_STACK\_USE\_UDP,

TCPIP\_STACK\_CLIENT\_MODE

are enabled from tcpip\_config.h

TCP\_SOCKET\_SIZE = 512

UDP\_SOCKET\_SIZE = 512

Built by C32 V1.10B without any optimization

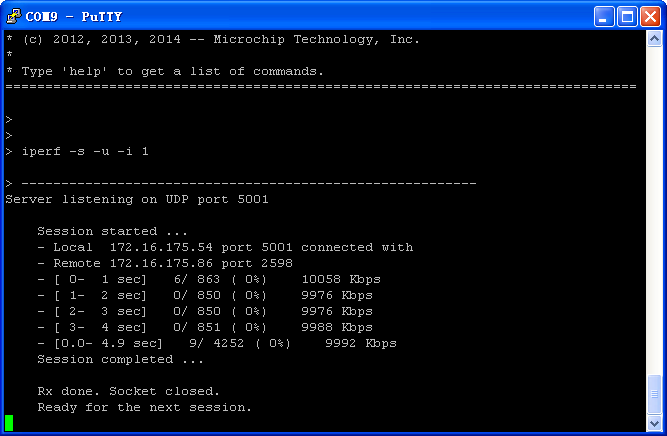
PC IP Address: 172.16.175.86

PIC32 ESK IP Address: 172.16.175.54

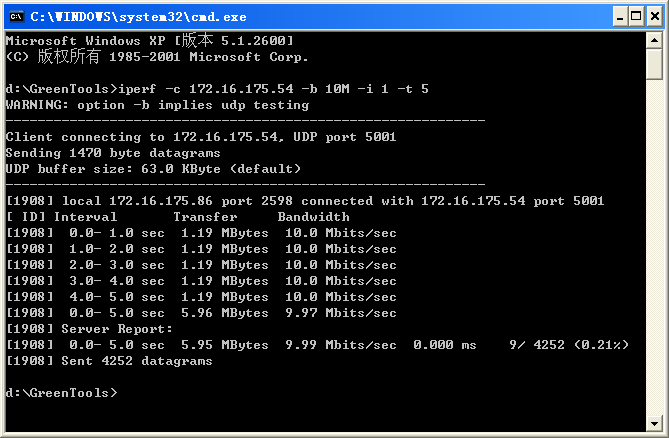
**UDP Test:**

**PIC32 ESK as UDP server(PIC32 receive):**

Command on PIC32: iperf -s -u -i 1

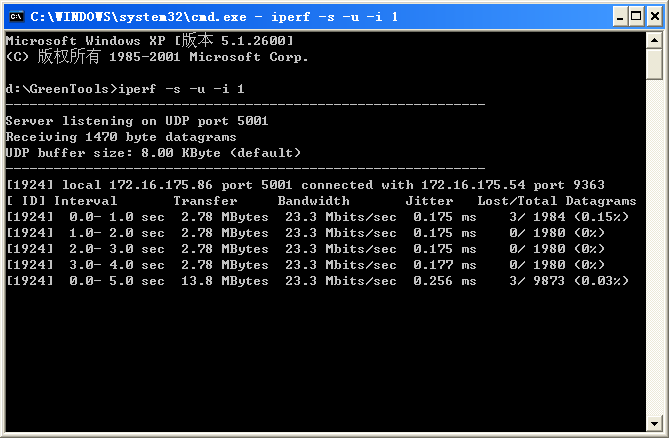


Command on PC: iperf -c 172.16.175.54 -b 10M -i 1 -t 5

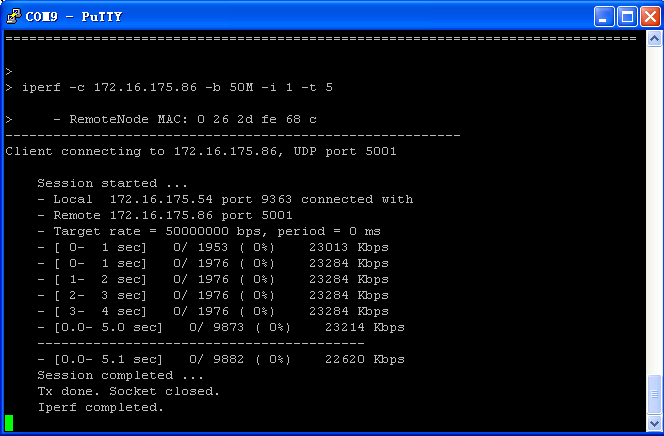


**PC as UDP server(PIC32 transmit)**

Command on PC: iperf -s -u -i 1

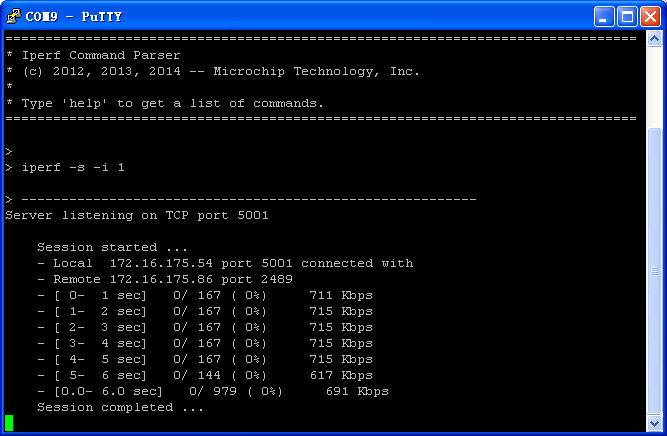


Command on PIC32: iperf -c 172.16.175.86 -b 50M -i 1 -t 5



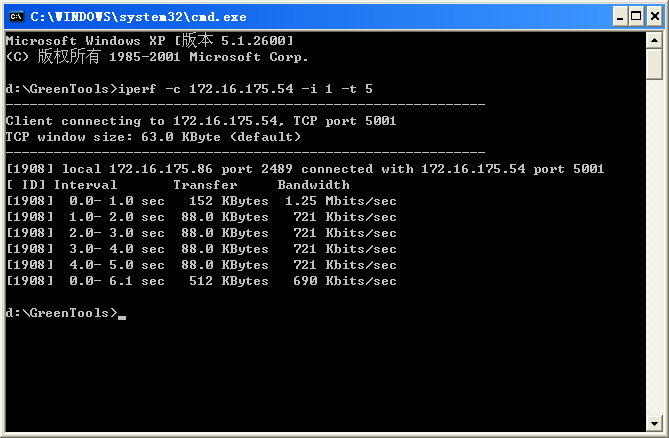
**TCP Test**

**PIC32 ESK as TCP server(PIC32 receiving):**



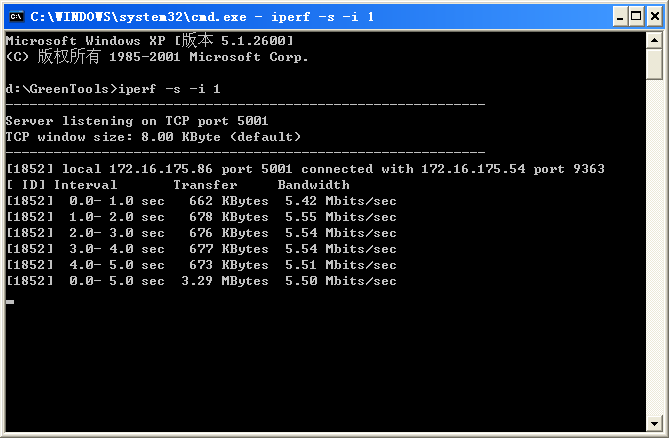
Command on PIC32: iperf -s -i 1

Command on PC: iperf -c 172.16.175.54 -i 1 -t 5



**PC as TCP server(PIC32 transimit):**

Command on PC: iperf -s -i 1



Command on PIC: iperf -c 172.16.175.86 -x 10M -i 1 -t 5

