UDP Benchmark

1.0

Generated by Doxygen 1.8.6

Fri Aug 1 2014 09:23:14

Contents

Chapter 1

Main Page

This application is used to test the performance of the open source FPGA UDP/IP core udp_ip_stack present on opencores.org. In particular this application allows to test the packet loss rate and the transmission data rate both from the PC to the FPGA and from the FPGA to the PC.

The speed performance strongly depends on the software PC implementation, in this case on the Qt implementation.

During the FPGA to PC test, the FPGA sends packets at its maximum speed (125MB/s) so that the Qt application looses some packets thus providing a wrong packet loss measure. In fact by using wireshark its possible to see that all the packets are received correctly by the PC. By setting the delay between two subsequent transmission (in clock cycles) in the FPGA it's possible to obtain a loss rate of 0% by reducing the maximum data rate. This is probably due to the elaboration time of each packet introduced by Qt.

Version control

This software is managed through the version control system git. Every change made to the software should be registered with git in order to keep track of what has been modified and by who.

WARNING: The software located in the shared folder *luxor* should not be modified nor a git repository should be present there (because since the *luxor* folder is managed through an online synchronization service, there is an high chance the the git repository get corrupted). In the *luxor* folder should be placed a copy of the software that has reached a newer stable version. To work on the software the online git repository should be cloned in a private work directory, and all the changes should be than pushed back online to the repository.

The remote repository for this software is located on BitBucket: udp-benchmark

The repository can be cloned locally with the following command (the repository password must be provided):

```
git clone https://quantumfuture-2@bitbucket.org/quantumfuture-2/udp-benchmark.git
```

A complete guide on git is Pro git book.

Documentation

To consult the API documentation open one of the following files:

- **index.html** located in the folder *doc/html/* (Recommended)
- **refman.pdf** in the folder *doc/latex/* (A copy of refman.pdf has been placed in the root folder for convenience. It should be updated every time the documentation is regenerated.)

To update the Doxygen documentation run the following command from the root folder:

```
doxygen Doxygen
```

then, to generate the PDF manual (refman.pdf), execute the following command from the folder doc/latex:

```
make
```

One line command to regenerate the documentation under Linux and MAC:

```
doxygen Doxygen && make -C doc/latex/ && cp doc/latex/refman.pdf refman.pdf
```

Author

Simone Gaiarin (simgunz@gmail.com)

License

References

- UDP/IP core official site
- UDP/IP core improved version containing all the Virtex 6 MAC layer components

Chapter 2

Bug List

Class UDPBenchmark

Under Windows the software is not able to use the QSocketNotifier object (or some workaround may be implemented as specified in the API page of QSocketNotifier), because of this the PC>FPGA is performed by sending the UDP packets one after the other and the Sleeper calss is used to suspend the process for some milliseconds in order to not overload the network layer, which implies a packet loss. The method startPCToFpgaTest has two different implementations in Windows Unix (using compiler the #if compiler directive).

Under Windows the readPendingDatagram() method is currently never activated so that it's impossible to read any packets coming from the FPGA. Because of this all the test measures won't be displayed. In any case it's possible to use Wireshark to check if any packet has been lost during the test thus doing a manaul test. This bug can be solved with some workarounds.

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 6

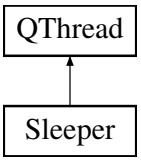
Class Documentation

6.1 Sleeper Class Reference

The Sleeper class provides an easy way to suspend a parent process for the specified number of microseconds.

`#include <sleeper.h>`

Inheritance diagram for Sleeper:



Public Member Functions

- Sleeper (QObject *parent=0)

  Default constructor.

Static Public Member Functions

- static void sleep (int usecs)

  Suspend the QThread process for the specified number of microseconds.

6.1.1 Detailed Description

The Sleeper class provides an easy way to suspend a parent process for the specified number of microseconds.

It acts like the usleep() function of c but it wraps this function inside a QThread so that it can be used in a parent Qt application.

Definition at line 37 of file sleeper.h.

6.1.2 Member Function Documentation

6.1.2.1 void Sleeper::sleep (int *usecs*) [static]

Suspend the QThread process for the specified number of microseconds.

Parameters

usecs	The number of microseconds that the process should sleep.

Definition at line 32 of file sleeper.cpp.

The documentation for this class was generated from the following files:

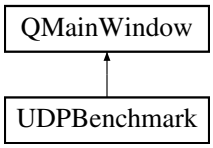
- src/sleeper.h
- src/sleeper.cpp

6.2 UDPBenchmark Class Reference

The UDPBenchmark class provides a graphical interface to perform a speed and loss rate test on the Vitrtex 6 UDP/IP core.

`#include <udpbenchmark.h>`

Inheritance diagram for UDPBenchmark:



Public Types

- enum TestType { **Idle**, **PcToFpga**, **FpgaToPc** }

  Enumerates the possible types of the test in progress.

Public Member Functions

- UDPBenchmark (QWidget *parent=0)

  Default constructor.
- ~UDPBenchmark ()

  Deconstructor.

Static Public Attributes

- static const int MTU = 1472

  Constant defining the maximum data bytes an UDP packet can contains.
- static const quint16 m_LocalPort = 0x6af0

  Constant defining the port on which the PC listen for incoming UDP packets.
- static const quint16 m_fpgaPort = 0x84be

  Constant defining the port on which the FPGA listen for incoming UDP packets.

Private Slots

- void setFpgaAddress (QString address)

  Sets the FPGA IP address to which send the UDP packets from the UI form.
- void setPktNum (int num)

Sets the FPGA IP address to which send the UDP packets from the UI form.

- void readPendingDatagrams ()

  Reads the incoming UDP packets and elaborates them according to the test in progress.

- void startPcToFpgaTest ()

  Initialize the PC>FPGA test.

- void keepSending ()

  Sends an UDP packet to the FPGA everytime it's activated until reaching the max number of packets to be sent.

- void endPcToFpgaTest ()

  Terminates the PC>FPGA test.

- void startFpgaToPcTest ()

  Initialize the FPGA>PC test.

- void endFpgaToPcTest ()

  Terminates the FPGA>PC test.

Private Attributes

- Ui::UDPBenchmark * ui

  Pointer to the main window object used to access the ui elements.

- QUdpSocket * m_udpSocket

  UDP socket used to perform the communication.

- QSocketNotifier * m_notifier

  Socket notifier used in linux to trasmit the packets at a safe rate.

- TestType m_currentTestType

  The type of the test in progress.

- QByteArray m_defaultData

  The dummy bytes that are sent by the PC in each UDP packet.

- QHostAddress m_fpgaAddress

  The UDP packet destination address.

- int m_pktNum

  Number of packets to be sent in the test (used in both tests).

- int m_pktCountTx

  Number of packets already transmitted to FPGA (PC>FPGA).

- int m_pktCountRx

  Number of packets already received from FPGA (FPGA>PC).

- QTime * m_benchTimer

  Timer used to compute the data rate.

6.2.1 Detailed Description

The UDPBenchmark class provides a graphical interface to perform a speed and loss rate test on the Vitrtex 6 UDP/IP core.

The program can perform the following two tests:

- **PC>FPGA** This test sends a certain amount of packets to the FPGA and measures the throughput in MB/s by measuring the time required to finish the trasnmission. Moreover it shows the percentual loss rate by comparing the number of sent packets with the number of received packets by the FPGA, which send back to the PC a packet containing this information when requested.

- - **FPGA>PC** This test requires to the FPGA to send a certain amount of packets, measure the throughput in MB/s and evaluates the loss rate by comparing the received packets with the number of expected packets.

Bug Under Windows the software is not able to use the QSocketNotifier object (or some workaround may be implemented as specified in the API page of QSocketNotifier), because of this the PC>FPGA is performed by sending the UDP packets one after the other and the Sleeper calss is used to suspend the process for some milliseconds in order to not overload the network layer, which implies a packet loss. The method startPCToFpgaTest has two different implementations in Windows Unix (using compiler the #if compiler directive).

Bug Under Windows the readPendingDatagram() method is currently never activated so that it's impossible to read any packets coming from the FPGA. Because of this all the test measures won't be displayed. In any case it's possible to use Wireshark to check if any packet has been lost during the test thus doing a manaul test. This bug can be solved with some workarounds.

Definition at line 61 of file udpbenchmark.h.

6.2.2 Member Enumeration Documentation

6.2.2.1 enum UDPBenchmark::TestType

Enumerates the possible types of the test in progress.

The program can be in an Idle state or one of the two tests PC>FPGA or FPGA>PC can be in progress.

Definition at line 73 of file udpbenchmark.h.

6.2.3 Constructor & Destructor Documentation

6.2.3.1 UDPBenchmark::UDPBenchmark (QWidget * *parent* = 0) [explicit]

Default constructor.

Initializes the UDP socket, creates a dummy UDP package containing a MTU bytes.

Parameters

The	QWidget this object is parented to.

Definition at line 35 of file udpbenchmark.cpp.

6.2.4 Member Function Documentation

6.2.4.1 void UDPBenchmark::endFpgaToPcTest ( ) [private], [slot]

Terminates the FPGA>PC test.

Computes and displays in the UI the data rate and the data loss.

Definition at line 250 of file udpbenchmark.cpp.

6.2.4.2 void UDPBenchmark::endPcToFpgaTest ( ) [private], [slot]

Terminates the PC>FPGA test.

Computes and displays in the UI the data rate. By sending the byte 'BB' to the FPGA the software requires the number of received packets by the FPGA. The response packet is catched by readPendingDatagrams() and the loss rate label is set.

Definition at line 169 of file udpbenchmark.cpp.

6.2.4.3 void UDPBenchmark::keepSending ( ) [private],[slot]

Sends an UDP packet to the FPGA everytime it's activated until reaching the max number of packets to be sent.

This method is activated by the 'activated' signal of the QSocketNotifier each time the network layer is ready to receive new packets. Using this approach there won't be any packet loss due to network overload.

This method is compiled only under Linux.

Definition at line 159 of file udpbenchmark.cpp.

6.2.4.4 void UDPBenchmark::readPendingDatagrams ( ) [private],[slot]

Reads the incoming UDP packets and elaborates them according to the test in progress.

This method is automatically activated each time an UDP packets has arrived to the UDP socket on the specified port.

- **PC>FPGA test** When this test is in progress the FPGA send a packet containing the number of packets that it has received. This number is used to compute the packet loss, which is then displayed to the user. To request this packet to the FPGA, a packet containing the byte 'BB' must be sent to the FPGA.
- **FPGA>PC test** When this test is in progress a counter is incremented each time a packet arrives. A packet with first byte equal to 'DD' marks the end of the test.

Warning

During the FPGA>PC test, if the last packet is lost the test won't produce any results, so it must be re-run again.

Definition at line 102 of file udpbenchmark.cpp.

6.2.4.5 void UDPBenchmark::setFpgaAddress (QString *address*) [private],[slot]

Sets the FPGA IP address to which send the UDP packets from the UI form.

This method is automatically invoked each time the user edit the FPGA IP address form in the user interface.

Parameters

address	A string containing the address in the form XXX.XXX.XXX.XXX.

Definition at line 92 of file udpbenchmark.cpp.

6.2.4.6 void UDPBenchmark::setPktNum (int *num*) [private],[slot]

Sets the FPGA IP address to which send the UDP packets from the UI form.

This method is automatically invoked each time the user edit the FPGA IP address form in the user interface.

Parameters

The	number of packets to be sent in the test.

Definition at line 97 of file udpbenchmark.cpp.

6.2.4.7 void UDPBenchmark::startFpgaToPcTest ( ) [private],[slot]

Initialize the FPGA>PC test.

Resets the loss rate and data rate labels in the UI, resets the local packet counter and the FPGA packet counter (by sending the byte 'AA'), starts the timer to measures the data rate. Finally it send a packet to the FPGA to begin the test, to set the number of packets to be sent and to set the number of clock cycles the FPGA should wait between two subsequent packet transmission.

Packet format: CCXXXXXXXXYYYYYYYY

- CC byte that marks the start of the test
- XXXXXXXX hexadecimal representation padded with zeros of the number of packets the FPGA should send
- YYYYYYYY hexadecimal representation padded with zeros of the number of clock cycles the FPGA should wait between subsequent packet transmission

Definition at line 218 of file udpbenchmark.cpp.

6.2.4.8 void UDPBenchmark::startPcToFpgaTest ( ) [private], [slot]

Initialize the PC>FPGA test.

Resets the loss rate and data rate labels in the UI, resets the local packet counter and the FPGA packet counter (by sending the byte 'AA'), starts the timer to measures the data rate.

- **Linux** Enables the QSocketNotifier in order to subsequently let the keepSending() method send all the test packets.
- **Windows** Sends all the UDP packets to the FPGA waiting a certain amount of time between each transmission by using the Sleeper:sleep() method. Finally displays the test results on the UI.

Definition at line 135 of file udpbenchmark.cpp.

The documentation for this class was generated from the following files:

- src/udpbenchmark.h
- src/udpbenchmark.cpp

Chapter 7

File Documentation

7.1 src/main.cpp File Reference

Contains the main of the UDPBenchmark project.

```
#include "udpbenchmark.h"
#include <QApplication>
```

Functions

- int **main** (int argc, char *argv[])

7.1.1 Detailed Description

Contains the main of the UDPBenchmark project.

Definition in file main.cpp.

7.2 src/sleeper.cpp File Reference

Contains the implementation of the Sleeper class.

```
#include "sleeper.h"
```

7.2.1 Detailed Description

Contains the implementation of the Sleeper class.

Definition in file sleeper.cpp.

7.3 src/sleeper.h File Reference

Contains the interface of the Sleeper class.

```
#include <QThread>
```

Classes

- class Sleeper

  The Sleeper class provides an easy way to suspend a parent process for the specified number of microseconds.

7.3.1 Detailed Description

Contains the interface of the Sleeper class.

Definition in file sleeper.h.

7.4 src/udpbenchmark.cpp File Reference

Contains the implementation of the UDPBenchmark class.

```
#include "udpbenchmark.h"
#include "ui_udpbenchmark.h"
#include "sleeper.h"
#include <QByteArray>
#include <QSocketNotifier>
#include <QTime>
#include <QUdpSocket>
```

7.4.1 Detailed Description

Contains the implementation of the UDPBenchmark class.

Definition in file udpbenchmark.cpp.

7.5 src/udpbenchmark.h File Reference

Contains the interface of the UDPBenchmark class.

```
#include <QHostAddress>
#include <QMainWindow>
```

Classes

- class UDPBenchmark

  The UDPBenchmark class provides a graphical interface to perform a speed and loss rate test on the Vitrtex 6 UDP/IP core.

7.5.1 Detailed Description

Contains the interface of the UDPBenchmark class.

Definition in file udpbenchmark.h.

Index