DE4 NetFPGA Reference Router User Guide
### Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Comment</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/11/2011</td>
<td>Initial draft</td>
<td>Harikrishnan</td>
</tr>
<tr>
<td>08/15/2012</td>
<td>Revision 1 – DMA APIs included</td>
<td>Harikrishnan</td>
</tr>
<tr>
<td>08/23/2012</td>
<td>Revision 2 – Directory Structure &amp; Linux sections</td>
<td>Siva</td>
</tr>
</tbody>
</table>
1. **Introduction**

DE4 NetFPGA is an open source port of the NetFPGA [2] platform on Altera DE4 board. DE4 NetFPGA provides network researchers with a powerful open platform to build complex network applications. Its features:

- PCI Express Interface for faster host PC to FPGA data transfers.
- Up to 8G external DDR memory

This tutorial describes the steps to obtain DE4 NetFPGA hardware, download the DE4 NetFPGA reference router design from the project website, compile the design using Altera Quartus II software, and verify the reference router by sending test packets.

2. **System Requirements**

To successfully install and test DE4 NetFPGA, you must satisfy the following system requirements.

**Operating System Requirements**

- Microsoft Windows XP or later / Linux (for JTAG mode)
- Microsoft Windows XP (for PCI Express mode)

**Software Requirements**

- Altera Quartus II 11.1 or later and SOPC Builder
- Microsoft Visual Studio 2010 (PCIe mode in Windows only)
- Jungo Windriver 32bit (PCIe mode in Windows only)

**Hardware Requirements**

- PC with PCIe x8 or x16 slot (PCIe mode in Windows only)
- Altera DE-4 board
- Two Gigabit Ethernet Cables for testing reference router.

**IP License Requirements**

- Altera Triple Speed Ethernet MAC (TSE MAC) IP Core

*Note: Altera provides a free time limited Open Core Plus core of Triple Speed Ethernet MAC (TSE MAC) with a time limit of 1 hour. This core is available as part of the Altera Quartus II software.*

**Optional (for testing packet transmission with DE4-NetFPGA)**

- Pre-installed NetFPGA Cube with packet generator software, available from NetFPGA website [2].
Table 1 provides additional information on obtaining the hardware and software necessary to compile DE4 NetFPGA reference router.

### Table 1: Software and Hardware Requirements

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Altera Quartus II 11.0 and SOPC Builder</td>
<td><a href="https://www.altera.com/download/software/quartus-ii-se">https://www.altera.com/download/software/quartus-ii-se</a></td>
</tr>
<tr>
<td>NetFPGA Cube with packet generator software</td>
<td><a href="http://www.accenttechnologyinc.com/netfpga.php">http://www.accenttechnologyinc.com/netfpga.php</a> <a href="http://netfpga.org/foswiki/bin/view/NetFPGA/OneGig/Projects/PacketGenerator">http://netfpga.org/foswiki/bin/view/NetFPGA/OneGig/Projects/PacketGenerator</a></td>
</tr>
</tbody>
</table>

3. Obtaining Altera DE4 Board

Altera DE4 is available from Terasic Technologies that integrates the state of the art Altera Stratix IV FPGA (EP4SGX230/EP4SGX530) with four Gigabit Ethernet interfaces, several high speed I/Os, and high density static and dynamic memory interfaces. The DE4 board can be purchased from Terasic’s website [8].

4. Installing NetFPGA DE4 Board in the Host PC

The DE4 NetFPGA can be connected to the host PC in two configurations as follows:

- **JTAG mode**
- **PCI Express mode (Windows only)**

The PCI Express mode is not supported in Linux yet. Select the 100MHz clock by setting SW7 switch to 00 (both ON) position as shown in Figure 1.

![Figure 1: Select 100 MHz clock](https://via.placeholder.com/150)
JTAG mode:

To operate the Altera DE4 in JTAG mode, set the SW0 switch in ON position. This is shown in Figure 2.

Figure 2: SW0 switch ON position

The Altera DE4 board is powered up using an external stand alone power supply in the JTAG mode. This configuration is shown in Figure 3. The FPGA bit-stream and the router configuration are programmed through the USB/JTAG interface.

Figure 3: JTAG mode
PCI Express mode (Windows only):

To operate Altera DE4 in PCI Express mode, perform the following two steps:

1. Enable PCI mode by setting the SW0 switch in **OFF** position as shown in Figure 4.

2. Configure PCI Express for Gen1 operation. To enable Gen1, set the PCI Express Control DIP switch SW9.1 in **ON** position as shown in Figure 5.

![Figure 4: SW0 switch OFF position](image1)

![Figure 5: PCI Express Control DIP Switch](image2)
The PCI Express mode does not require a standalone power supply. Instead, the Altera DE4 board can be powered up through the x8 or x16 PCI Express edge connector attached to a PC as shown in Figure 6.

5. Obtaining DE4 NetFPGA Reference Router

The NetFPGA reference router is an open line rate (1Gbps) IPv4 router [3] available for education and research purposes from Stanford University. The DE4 NetFPGA reference router provides a port of the NetFPGA reference router on the Altera DE4 platform. Download the gateware and software for DE4 NetFPGA reference router as two separate packages:

- Download **DE4 NetFPGA directories package (NF2_DE4_BASE.zip)** provided by University of Massachusetts, Amherst from [17]. A link to the repository is also available from the project website [1].
- Obtain the **DE4 Reference Router package (DE4_Reference_Router_v2.0.zip)**, a modified version of Stanford University’s NetFPGA reference router design by sending an e-mail to Professor Russell Tessier (email: tessier@ecs.umass.edu).

6. Unzipping Files

- Unzip NF2_DE4_BASE.zip and DE4_Reference_Router_v2.0.zip to the same directory. Unzipping NF2_DE4_BASE.zip will create the following directory structure:
Unzipping \texttt{DE4\_Reference\_Router\_v2.0.zip} will merge reference router source files with the “lib” and “project” directories in the base package. These directories are shown in Figure 8 and Figure 9.

The \texttt{nf2\_de4\_base} folder contains the following subdirectories –

- \texttt{bin} - Scripts to build the project and download the bit-stream with Linux build.
- \texttt{bitfiles} - Bit-streams generated in Linux environments are copied after compilation.
- \texttt{Lib} - Contains source files and megafunctions that are common for all projects.
- \texttt{Projects} – Project specific source files and compilation environment.

The \texttt{lib} folder contains the following subdirectories –

- \texttt{Verilog/core} – Source files and megafunctions that are common across all projects
-The projects folder contains the following subdirectories –

- **src** – Project specific verilog files must be placed here. Source files placed in this folder override the files with the same name in **lib/verilog/core**. In Linux, these files will be automatically sourced from Makefiles. In Windows, if users need to create/replace a module in ‘lib/verilog/core’, they should place the files in ‘src’ and manually add them to the project (the corresponding ‘lib/verilog/core’ files have to be removed manually).

- **sw** - Contains two subdirectories - **jtag_config** and **pcie_config**. The ‘jtag_config’ folder contains TCL scripts for configuring the Gigabit Ethernet and reference router tables in JTAG mode of operation while ‘pcie_config’ contains the Visual Studio software application for the same in PCIE mode.

- **Synth** – Contains two subdirectories ‘linux’ and ‘windows’. In ‘windows’ directory, the Altera project design files for compiling the project in Windows are included. Similarly in ‘linux’, ‘build_dir’ has the makefile to compile the project; ‘project_files_backup’ has the Altera project files that are copied into ‘build_dir’ when **make clean** command is used.

---

8. Compiling and configuring the DE4 NetFPGA Reference Router in Windows

8.1. Compilation

The compilation steps for Linux are described in Section 9. The detailed compilation steps for Windows are described below:

1. Open the DE4_Reference_Router project from Quartus by double clicking **NF2_DE4_BASE/projects/DE4_Reference_Router/synth/windows/DE4_Reference_Router.qpf**

2. Open SOPC Builder (Quartus → Tools → SOPC Builder)
3. Generate the SOPC System. Click Generate on the SOPC builder

4. Once SOPC system generation is successful, compile the design.
   (Quartus → Processing → Start Compilation)
5. Install the Altera DE4 board and turn it on if using JTAG mode. Make sure that the board is connected to PC via the USB/JTAG cable.

6. Install USB/JTAG drivers for the PC [13].

7. Download the bit file generated after the compilation to the target DE4 board by double clicking the "Program Device" in the Task Window.
8. Click Start in the Programmer window to begin download.

Note: If you are using PCI Express mode, restart the host PC after downloading the bit-file.

8.2. Configuration

Once the reference router bit-stream has been programmed onto the FPGA, the following steps must be completed before we can test the DE4 NetFPGA reference router:

1. Bring up the Gigabit Ethernet Interfaces (hereafter referred to as gige_config).
2. Configure router tables (hereafter referred to as set_router_tables).

The first step (gige_config) brings up the four Gigabit Ethernet interfaces (MAC [0-3] and PHY [0-3] in Figure 10) attached to the DE4 board. The second step (set_router_tables) assigns user defined MAC, IP address, LPM and ARP values to the reference router’s routing table. The configuration files to achieve the steps above are shown in Table 2. {project root} refers to NF2_DE4_BASE/projects/DE4_Reference_Router.

<table>
<thead>
<tr>
<th>Step</th>
<th>JTAG Mode</th>
<th>PCIe Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>gige_config</td>
<td>{project root}\sw\jtag_config\config_gige.tcl</td>
<td>{project root}\sw\pcie_config\pcie_ethernet_config.h</td>
</tr>
<tr>
<td>set_router_tables</td>
<td>{project root}\sw\jtag_config\mac.tcl</td>
<td>{project root}\sw\pcie_config\set_router_tables.h</td>
</tr>
</tbody>
</table>
Next, we describe the procedure to execute these steps in detail for both JTAG mode and PCIe modes of operation.

8.2.1. JTAG Mode

Before you start, make sure that the board is installed in JTAG mode according to the steps described in section 4 and the FPGA has been programmed with the reference router bit-stream.

To configure the DE4 reference router in JTAG mode perform the following steps:

1. Open SOPC Builder from Quartus II (In Quartus, select Tools → SOPC Builder)
2. In SOPC Builder, open system console (Tools → System Console).
3. Change the current directory (synth/windows/) to `jtag_config`
   
   ```
   cd ../../sw/jtag_config
   ```
4. To configure the Gigabit Ethernet interfaces, run the following command.
   ```
   > source config_gige.tcl
   ```
This step runs the TCL scripts for configuring MAC and PHY for the four Gigabit Ethernet interfaces. You would see a message displayed on the console window similar to the following if a Gigabit Ethernet cable is connected.

```
Info: Opened JTAG Master Service

Info: Configure TSE PCS

TSE PCS read rev = 0x00000000
TSE PCS write scratch = 0x00000000
TSE PCS read scratch = 0x00000000
TSE PCS read if_mode = 0x00000000
TSE PCS read_control register = 0x00000000
Waiting Link Up....
Link is established!
Partner Ability:
Copper link interface is up.
Copper operating in Full Duplex mode.
Copper operating Speed 1000Mbps

Info: Closed JTAG Master Service
```

5. The script `set_router_tables.tcl` contains the user defined values for configuring the reference router’s routing table.

In this example, we configure the MAC tables as shown in the Table 3.

### Table 3: MAC Configuration Table

<table>
<thead>
<tr>
<th>Interface</th>
<th>MAC</th>
<th>MAC Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETHERNET 0</td>
<td>MAC 0</td>
<td>00:4e:46:32:43:00</td>
</tr>
<tr>
<td>ETHERNET 1</td>
<td>MAC 1</td>
<td>00:4e:46:32:43:01</td>
</tr>
<tr>
<td>ETHERNET 2</td>
<td>MAC 2</td>
<td>00:4e:46:32:43:02</td>
</tr>
<tr>
<td>ETHERNET 3</td>
<td>MAC 3</td>
<td>00:4e:46:32:43:03</td>
</tr>
</tbody>
</table>

In this example, we configure the IP tables as shown in the Table 4.

### Table 4: Interface IP Configuration Table

<table>
<thead>
<tr>
<th>Interface</th>
<th>IP</th>
<th>IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETHERNET 0</td>
<td>IP 0</td>
<td>20.1.1.1</td>
</tr>
<tr>
<td>ETHERNET 1</td>
<td>IP 1</td>
<td>20.2.1.1</td>
</tr>
<tr>
<td>ETHERNET 2</td>
<td>IP 2</td>
<td>10.2.3.1</td>
</tr>
<tr>
<td>ETHERNET 3</td>
<td>IP 3</td>
<td>10.1.4.1</td>
</tr>
</tbody>
</table>
In this example, we configure the IP tables as shown in the Table 5.

**Table 5: IP LPM Configuration Table**

<table>
<thead>
<tr>
<th>Entry</th>
<th>Destination IP</th>
<th>Mask</th>
<th>Next Hop IP</th>
<th>Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10.1.4.0</td>
<td>255.255.255.0</td>
<td>10.1.4.1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>10.2.3.0</td>
<td>255.255.255.0</td>
<td>10.2.3.1</td>
<td>2</td>
</tr>
</tbody>
</table>

In this example we configure the ARP tables as shown in the Table 6.

**Table 6: ARP Configuration Table**

<table>
<thead>
<tr>
<th>Entry</th>
<th>Next Hop IP</th>
<th>MAC Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10.1.4.1</td>
<td>00:4e:46:32:43:03</td>
</tr>
<tr>
<td>2</td>
<td>10.2.3.1</td>
<td>00:4e:46:32:43:01</td>
</tr>
</tbody>
</table>

6. To configure the routing table run the following command.
   > source set_router_tables.tcl

8.2.2. PCIe Mode

Before you start, make sure that the board is installed in PCIe mode according to the steps described in section 4 and the FPGA has been programmed. Restart the host PC after downloading the reference router bit stream.

The software architecture to configure the FPGA via PCI Express bus is shown in Figure 11.

![Software Architecture](image-url)

**Figure 11: Software Architecture**

We use Jungo’s Win Driver [7] 32-bit to establish communication between the host PC running Windows XP and the Altera DE-4 board. Win Driver exposes two APIs (OnRCSlaveWrite and
OnRCSlaveRead) for applications to read and write using the PCI Express bus. The Visual Studio 2010 based application uses these APIs to read and write Gigabit Ethernet configurations and program routing table entries to the DE-4 board.

In addition, two APIs are provided for performing DMA read (DMA_read) and DMA write (DMA_write) with the FPGA. These APIs can be used for performing DMA transfers between the host PC and the DE4 board. More information on DMA is given in section 11. The Visual Studio application is available in {project root}\sw\pcie\config directory. {project root} refers to NF2_DE4_BASE/projects/DE4_Reference_Router.

The file structure of the pcie_config folder is shown below:

```
pcie_sw_installation
x86
altpcie_demo.cpp
pcie_ethernet_config.h
pcie_sw.inf
pcie_sw.vdp
pcie_sw_diag.cpp
pcie_sw_files.txt
pcie_sw_lib.c
pcie_sw_lib.h
router_tables.h
```

Next, we describe the steps to configure the Gigabit Ethernet interfaces, Interface MAC tables, Interface IP tables, IP LPM tables and ARP tables in PCIe mode below:

1. Double click on pcie_sw_diag.sln: The file is present at the location {project root}\sw\pcie\config\x86\msdev_2010

2. The Visual Studio 2010 window pops up with the following file structure.
3. The Gigabit Ethernet configuration and the router table configuration are mentioned in the following files (`pcie_ethernet_config.h`, `set_router_tables.h`) at location (`{project root}/sw/pcie_config`).

4. Rebuild the design: In Visual Studio, click Build → Rebuild Solution.

5. Click the green Run button (shown in Figure 12) to start the configuration utility. The configuration utility configures Gigabit Ethernet interfaces, Interface MAC tables, Interface IP tables, IP LPM tables and ARP tables.

```
5. Click the green Run button (shown in Figure 12) to start the configuration utility. The configuration utility configures Gigabit Ethernet interfaces, Interface MAC tables, Interface IP tables, IP LPM tables and ARP tables.
```

6. Click the green Run button (shown in Figure 12) to start the configuration utility. The configuration utility configures Gigabit Ethernet interfaces, Interface MAC tables, Interface IP tables, IP LPM tables and ARP tables.

### Figure 12: Start Debugging

#### 9. Compiling and configuring the DE4 NetFPGA Reference Router in Linux

##### 9.1. Compilation

To compile a project, carry out the following steps:

1. Make sure that Quartus is installed and its environmental variables are set up in `$HOME/.bashrc` file
   a. `export QUARTUS_HOME={path to Quartus installation}/altera/{version}`
   b. `PATH=$QUARTUS_HOME/quartus/bin:$PATH`
   c. `PATH=$QUARTUS_HOME/quartus/sopc_builder/bin:$PATH`
   d. `source $HOME/.bashrc`

2. Now setup the environmental variables needed for the DE4_Reference_Router project. If you want to create a new project “My_project” in ‘NF2_DE4_BASE/projects’ then use
My_project instead of DE4_Reference_Router in the following steps:

a. Export the path of NF2_DE4_BASE as NF2_DE4_ROOT; Example:
   ```
   export NF2_DE4_ROOT =/home/user/Desktop/NF2_DE4_BASE
   ```

b. Export the path to the DE4_Reference_Router project folder; e.g.:
   ```
   export DESIGN_DIR={NF2_DE4_ROOT}/projects/DE4_Reference_Router
   ```

c. Export the name of the project; e.g.:
   ```
   export PROJECT=DE4_Reference_Router
   ```

d. Export the name of the top module in project; e.g.:
   ```
   export TOP=DE4_Reference_Router
   ```

e. Export the name of the SOPC system in the project; e.g:
   ```
   export SOPC_SYS_NAME=DE4_SOPC
   ```

3. Run “make” in {NF2_DE4_ROOT}/projects/{PROJECT}/synth/linux/build_dir to compile
   the project. Note: SOPC generation command exits with a code 4 even on success. So make
   reports an error. This bug will be fixed in the next release. Running “make” again will start
   the compilation.

4. After the compilation is done, the bit-stream is found in
   {NF2_DE4_ROOT}/projects/{PROJECT}/synth/build_dir and a copy is found in
   {NF2_DE4_ROOT}/bitfiles.

5. Use “jtag_config” command to check if USB-Blaster driver is installed and JTAG hardware
   is detected. To debug common JTAG setup issues associated with Quartus running in linux,
   see [13].

6. To download the bit-stream onto the DE4 board, use the following command in the
   {NF2_DE4_ROOT}/bin directory:
   ```
   sh download.sh ../bitfiles/<bitstream.sof>
   ```

7. “make clean” can be run in {NF2_DE4_ROOT}/projects/{PROJECT}/synth/linux/build_dir
   if you wish to clean the built project.

Note: If using a time-limited version of the Open Core Plus core of Triple Speed Ethernet MAC
(TSE MAC), make sure that you don’t quit the programmer (in command line) or close the pop-up
window (in GUI) after programming is successful.

9.2. Configuration

Once the reference router bit-stream has been programmed onto the FPGA, complete the following
steps before testing the DE4 NetFPGA reference router:

1. Bring up the Gigabit Ethernet Interfaces (hereafter referred to as gige_config).
2. Configure router tables (hereafter referred to as set_router_tables).

   The first step (gige_config) brings up the four Gigabit Ethernet interfaces (MAC [0-3] and
   PHY [0-3] in Figure 13) attached to the DE4 board. The second step (set_router_tables)
   assigns user defined MAC, IP address, LPM and ARP values to the reference router’s routing
   table. The configuration files to achieve the steps above are shown in Table 7.

<table>
<thead>
<tr>
<th>Step</th>
<th>Configuration File</th>
</tr>
</thead>
<tbody>
<tr>
<td>gige_config</td>
<td>{NF2_DE4_ROOT}/projects/Reference_Router/sw/jtag_config/config_gige.tcl</td>
</tr>
<tr>
<td>set_router_tables</td>
<td>{NF2_DE4_ROOT}/projects/Reference_Router/sw/jtag_config/set_router_tables.tcl</td>
</tr>
</tbody>
</table>
config_gige.tcl step runs the TCL scripts for configuring MAC and PHY for the four Gigabit Ethernet interfaces.

The script set_router_tables.tcl contains the user defined values for configuring the reference router’s routing table. In the design provided (DE4_Reference_Router), we configure the MAC, IP, LPM and ARP tables as shown in Table 8, Table 9, Table 10 and Table 11 respectively.

Table 8: MAC Configuration Table

<table>
<thead>
<tr>
<th>Interface</th>
<th>MAC</th>
<th>MAC Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETHERNET 0</td>
<td>MAC 0</td>
<td>00:4c:46:32:43:00</td>
</tr>
<tr>
<td>ETHERNET 1</td>
<td>MAC 1</td>
<td>00:4c:46:32:43:01</td>
</tr>
<tr>
<td>ETHERNET 2</td>
<td>MAC 2</td>
<td>00:4c:46:32:43:02</td>
</tr>
<tr>
<td>ETHERNET 3</td>
<td>MAC 3</td>
<td>00:4c:46:32:43:03</td>
</tr>
</tbody>
</table>

Table 9: Interface IP Configuration Table

<table>
<thead>
<tr>
<th>Interface</th>
<th>IP</th>
<th>IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETHERNET 0</td>
<td>IP 0</td>
<td>20.1.1.1</td>
</tr>
<tr>
<td>ETHERNET 1</td>
<td>IP 1</td>
<td>20.2.1.1</td>
</tr>
<tr>
<td>ETHERNET 2</td>
<td>IP 2</td>
<td>10.2.3.1</td>
</tr>
<tr>
<td>ETHERNET 3</td>
<td>IP 3</td>
<td>10.1.4.1</td>
</tr>
</tbody>
</table>
Table 10: IP LPM Configuration Table

<table>
<thead>
<tr>
<th>Entry</th>
<th>Destination IP</th>
<th>Mask</th>
<th>Next Hop IP</th>
<th>Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10.1.4.0</td>
<td>255.255.255.0</td>
<td>10.1.4.1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>10.2.3.0</td>
<td>255.255.255.0</td>
<td>10.2.3.1</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 11: ARP Configuration Table

<table>
<thead>
<tr>
<th>Entry</th>
<th>Next Hop IP</th>
<th>MAC Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10.1.4.1</td>
<td>00:4e:46:32:43:03</td>
</tr>
<tr>
<td>2</td>
<td>10.2.3.1</td>
<td>00:4e:46:32:43:01</td>
</tr>
</tbody>
</table>

After setting up the tables as per requirement in the script ‘set_router_tables.tcl’ and setting up the test according to the test setups in section 10, run the following command:

```
{NF2_DE4_ROOT}/projects/Reference_Router/sw/jtag_config/config
```

10. Testing

The DE4 reference router can be tested once the reference router bit file has been downloaded into the FPGA and router tables have been programmed successfully into the FPGA. The reference router can be tested by sending sample packets to it and collecting the forwarded packets back. We have also ported the NetFPGA packet generator reference design to the DE4 platform. The DE4 based packet generator code (downloadable for the website) or the packet generator available with the NetFPGA platform (NetFPGA packet generator [11]) can be used to generate test packets and capture forwarded packets from the reference router. Unlike software-based packet generation utilities, packet generator allows packets to be transmitted and captured at line rate with a high degree of accuracy. The user can load a custom PCAP file into the packet generator and subsequently transmit the packets in the PCAP file at user defined rates through NetFPGA ports. For more information refer to the DE4 Packet Generator Design user guide [16].

10.1. Test Setup – DE4 Packet Generator

1. We configure a DE4 board with the packet generator reference design. You can download the reference design from the website.
2. Setup the packet generator as per the DE4 packet generator user guide [16].
3. The test topology is shown in Figure 14.
4. If you are operating DE4 Reference router in JTAG mode, connect two gigabit Ethernet cables between packet generator and the DE4 Reference router as shown in Figure 15.
5. If you are operating DE4 Reference router in PCIe mode, connect two gigabit Ethernet cables between packet generator and the DE4 Reference router as shown in Figure 16.
10.2. Test Setup – NetFPGA Packet Generator

2. To configure the NetFPGA-1G as a packet generator, navigate to $NF2_ROOT\projects\packet_generator\sw$ where NF2_ROOT is the root directory of the NetFPGA source tree [11].
3. Configure the NetFPGA 1G card attached to the NetFPGA Cube packet generator

   >setup_packet_generator.pl

4. The test topology is shown in Figure 14.
5. If you are operating DE4 Reference Router in JTAG mode, connect two gigabit Ethernet cables between packet generator and the DE4 Reference Router as shown in Figure 17.
6. If you are operating DE4 Reference Router in PCIe mode, connect two gigabit Ethernet cables between packet generator and the DE4 Reference Router as shown in Figure 18.

![Figure 14: Reference Router Test Setup](image-url)
Figure 15: Test setup with JTAG mode DE4 Reference Router

Figure 16: Test setup with PCI Express mode DE4 Reference Router
Figure 17: Test setup with NetFPGA cube and JTAG mode DE4 Reference Router

Figure 18: Test setup with NetFPGA cube and PCI Express mode DE4 Reference Router
10.3. Running the Test

1. If DE4 packet generator is used for testing, run the packet generator software from the Visual Studio. Observe the output from the output.txt file.
2. If NetFPGA packet generator is used for testing follow the steps below:
   - Start the packet generator using the command,

   ./packet_generator.pl –q <queue number> <pcap file> -r <queue number> <rate> -i <queue number> <number of iterations> -d <queue number> <delay between packets>

   - This command sends packets through the specific Ethernet interface queue (queue_number) at a user defined rate (rate) with a user defined delay (delay between packets) for a specific number of iterations (number of iterations).

Example:

To test packet transmission with 3200000 packets and zero delay between packets at line rate (1Gbps) through queue 2, use

./packet_generator.pl –q2 test.pcap -i2 3200000 –r2 1000000 –d2 0

When the topology as shown in Figure 14 is used, the packet transmitted at line rate (1Gbps) captured at port 3 of packet generator is illustrated in Figure 19.

Figure 19: Sample packet capture at line rate (1Gbps)
11. DMA

Direct Memory Access (DMA) is used for communicating between host PC and DE4 NetFPGA. The DMA interface is available in the project DE4_Reference_Router_with_DMA. To obtain this project, send a mail to Prof. Tessier (tessier@ecs.umass.edu). After downloading the project (DE4_Reference_Router_with_DMA_v2.1.zip) extract it in the same directory where NF2_DE4_BASE.zip was extracted. Follow this user guide for compiling the design. The project will be located in NF2_DE4_BASE/projects/DE4_Reference_Router_with_DMA (hereafter referred to as “project root”). DMA feature is described in detail below:

The PCIe mode DE4_Reference_Router_with_DMA has Direct Memory Access (DMA) feature to transfer data between the host PC and the CPU queues. Two APIs are provided for performing DMA read (DMA_read) and DMA write (DMA_write) with the FPGA. In the distribution available from the website [1], DMA is used to test the functioning of the reference router’s CPU queues. The following steps need to be performed to configure DMA functionality.

1. Double click on pcie_sw_diag.sln: The file is present at the location {project root}/sw/pcie_config/x86/msdev_2010

2. The Visual Studio 2010 window pops up with the following file structure.

3. Install WinPcap driver [14] and WinPcap developer’s pack [15]. The developer’s pack is installed in the location “C:\winpcap\WpdPack_4_0_2\WpdPack”.
4. Link Visual Studio with the WinPcap libraries as mentioned below.
5. In Visual Studio, under the Configuration Properties → C/C++ → General tab, add the WinPcap include path to Additional Include Directories.
6. Under the Configuration Properties → Linker → General tab, add the WinPcap library path to Additional Library Directories.
7. Under the Configuration Properties → Linker → Input tab, add the two main WinPcap libraries (wpcap.lib & Packet.lib) to Additional Dependencies.

8. Under the Configuration Properties → Debugging tab, add the file (test.pcap). This packet capture file is used for DMA read operation.

9. The Gigabit Ethernet configuration and the router table configuration are mentioned in the following files (pcie_ethernet_config.h, set_router_tables.h) at location ({project root}/sw/pcie_config).

10. Rebuild the design: In Visual Studio, click Build → Rebuild Solution.
11. Click the green Run button (shown in Figure 20) to start the configuration utility. The configuration utility configures Gigabit Ethernet interfaces, Interface MAC tables, Interface IP tables, IP LPM tables and ARP tables.

![Figure 20: Start Debugging](image)

DMA features can be accessed from your programs through the following APIs.

**11.1. DMA read**

The API is used as follows:

\[ \text{DMA\_read} \text{(Device handle, MAC port)} \]

Here \text{MAC port} is the port through which the data should be sent out of the reference router. The packet data is read from the packet capture file included (\text{test.pcap}). An example DMA read operation is shown below.

\[ \text{DMA\_read}(\text{hdev}, 0x03) \], this sends packet data from the \text{test.pcap} file to MAC port 3 of the reference router.

**11.2. DMA write**

The API is used as follows:

\[ \text{DMA\_write} \text{(Device handle, Packet length, CPU port)} \]

Here \text{Packet length} is the length of each packet you are expecting from the reference router’s CPU queue. Each time the \text{DMA\_write} function is called one complete packet is send from the CPU queue to the PC. \text{CPU port} is the CPU queue from where packet data needs to be transferred. The data transferred from the CPU queue gets stored in a buffer inside the \text{DMA\_write} function (\text{DMA\_write\_buf}). The \text{DMA\_write} function needs to be called multiple times to transfer all the packets from the CPU queues. An example DMA write operation is shown below.
**DMA_write**((hdev,106,0x0)), this stores a packet of size 106 bytes, transferred from the CPU queue 0 into the buffer.

### 12. References


[9]. Altera Quartus II 11.0 and SOPC Builder, [https://www.altera.com/download/software/quartus-ii-sc](https://www.altera.com/download/software/quartus-ii-sc)


[11]. NetFPGA packet generator [http://netfpga.org/foswiki/bin/view/NetFPGA/OneGig/Projects/PacketGenerator](http://netfpga.org/foswiki/bin/view/NetFPGA/OneGig/Projects/PacketGenerator)


[17]. DE4 Base repository [https://github.com/UmassRCG/nf2_de4_base](https://github.com/UmassRCG/nf2_de4_base)