

## **EE533: Network Processor Design & Programming**

### **Lab #4: NetFPGA BitFile Generation and Using NetFPGA**

Instructor: Prof. Young Cho, PhD

Team Number: **#3**

*Project Partners:*

*Member #1: Sarthak Jain*

*Member #2: Archit Sethi*

*Member #3: Justin Santos*

*Designed, Created, and Submitted by Team #3*

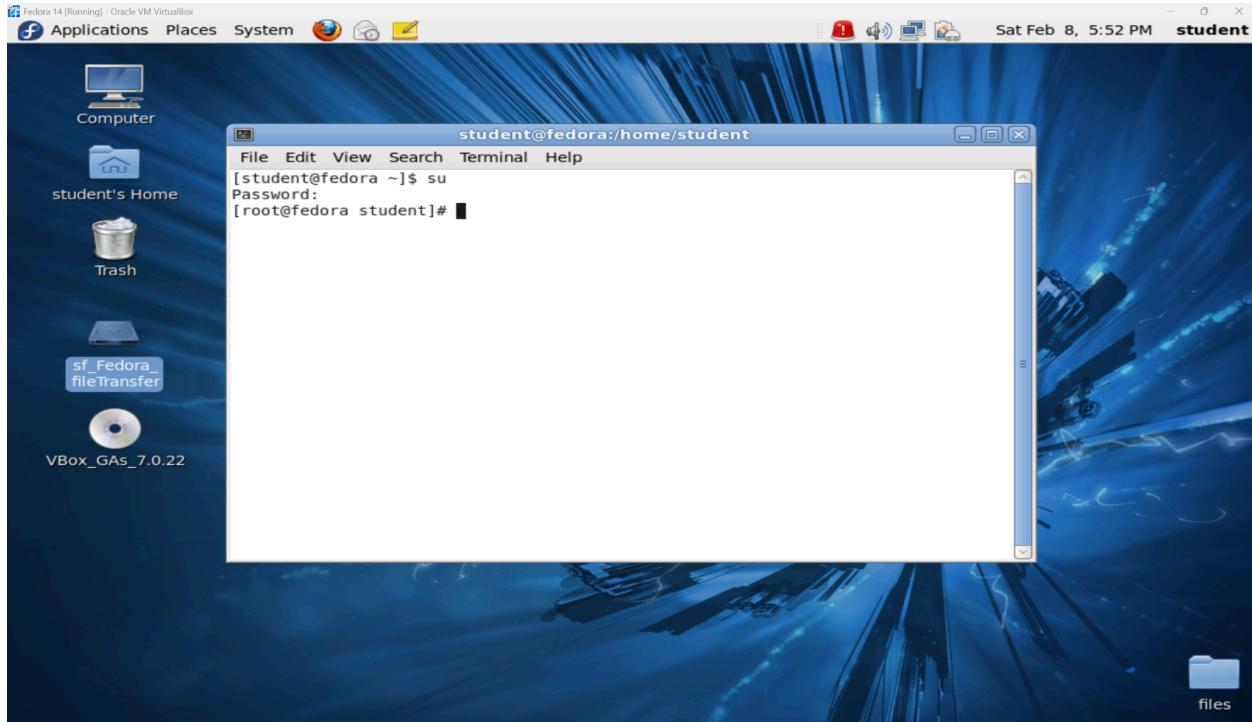
*University of Southern California*

*Los Angeles, CA 90007*

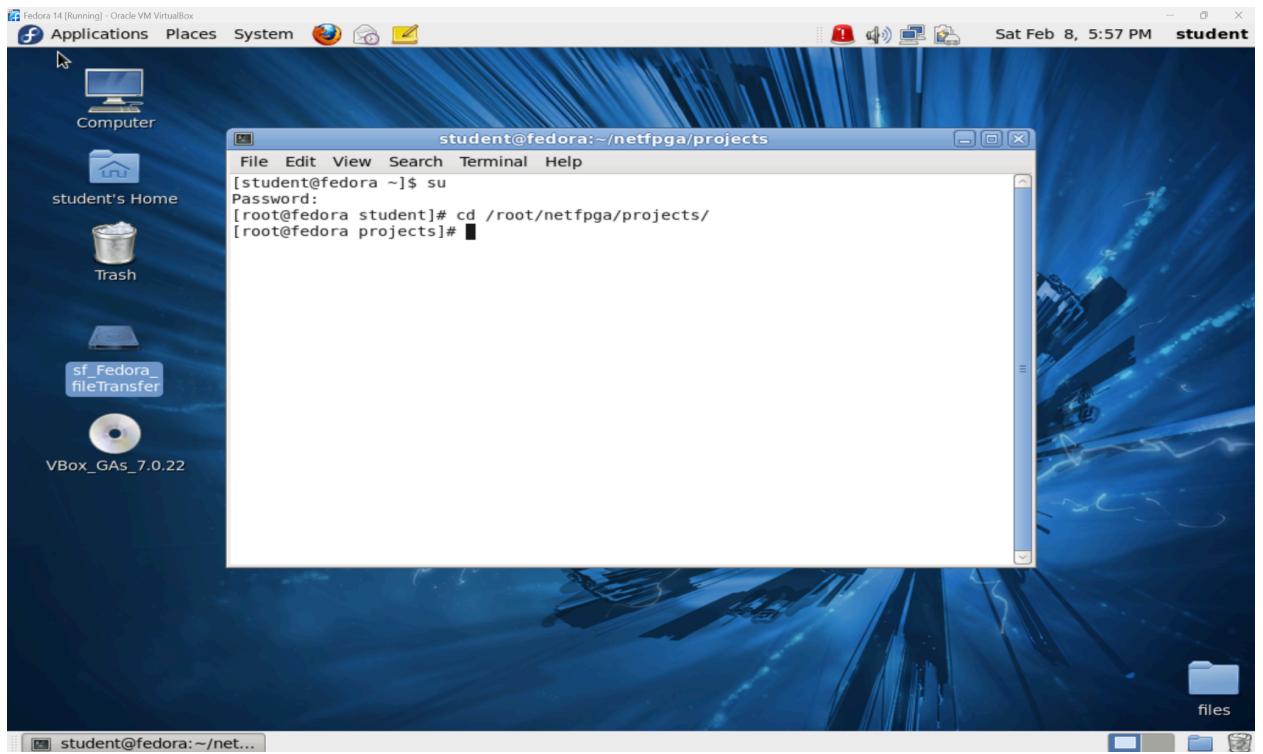
Team #4 GitHub Repository Link: [Team#3 Link](#)

## Part-1: Download and Set-Up NetFPGA Tool Virtual Machine

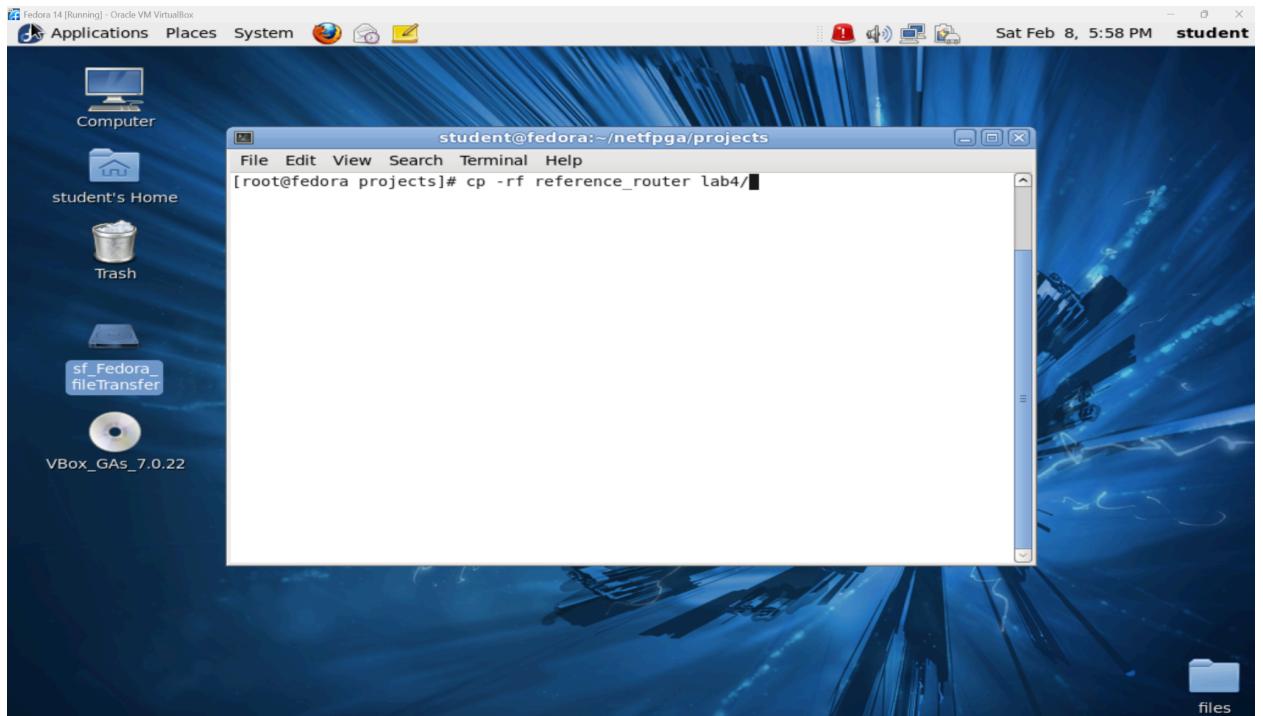
1. Virtual Machine Fedora 14 was successfully downloaded and installed on the VirtualBox software.
2. In the terminal, enter 'su' to switch into root user to compile the design into a bitfile.



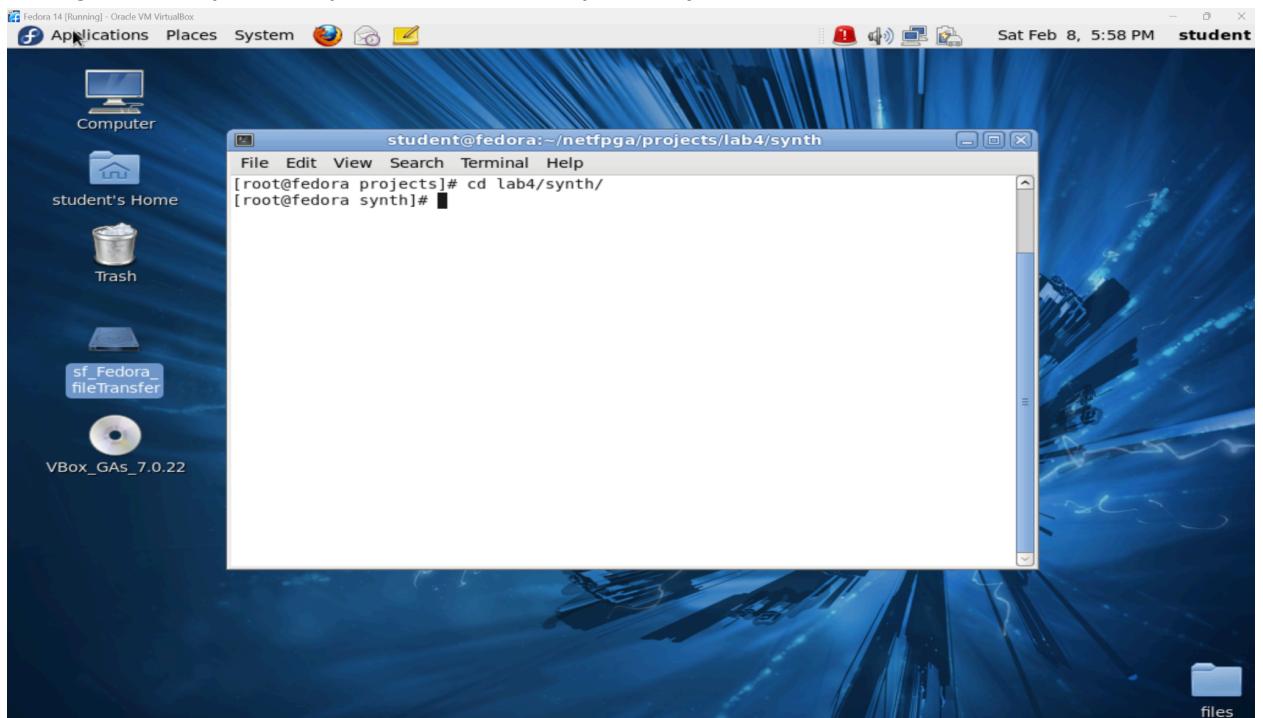
3. Change directory into the NetFPGA projects directory with 'cd' command.



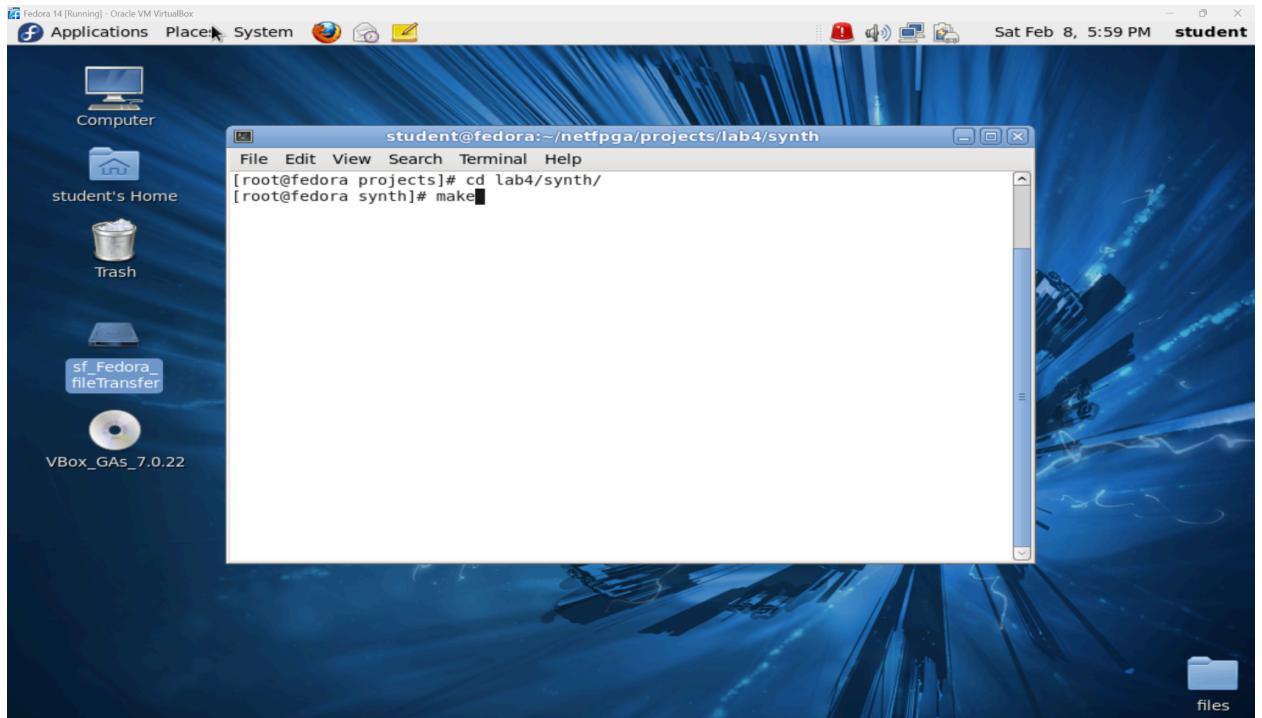
4. Copy the reference\_router project into your project folder with ‘cp’ command



5. Change directory into a synthesis folder into your project with ‘cd’ command



6. Compile the design by typing 'make'



7. This should start the compilation and generation of a bit-file for your reference\_router project

The command 'make' was running and we found that there were many errors from the imported schematics from the previous lab-3. The debugging was a tedious process but was handled with efficiency.

## Part-2: Compile and generate a design bitfile for NetFPGA

1. Converting schematics to verilog code as it was demonstrated in the following video.
2. After extraction of 'ids\_hw' from lab-3\_mini\_ids\_src.zip
3. Copied all converted verilog and ip core files into ids\_hw/src.
4. This is how the make command ran and we got a few errors which have been depicted further below.

The screenshot shows a terminal window titled "student@fedora:~/netfpga/projects/lab4/synth". The window contains the output of a make command. The output includes several warning messages related to XML processing for various modules like 'core/sram\_arbiter/sram\_weighted\_rr' and 'core/user\_data\_path/reference\_user\_data\_path'. It also shows the processing of Verilog files for cores like 'cpci' and 'nf2\_top'. The process involves generating XML files for device ID registers and generic registers. The output ends with a summary of the environment variables (Root dir, Project name, Project dir, Work dir), a summary of the project details (Project: 'CPCI' (cpci), Description: NetFPGA PCI interface, Version: 4.1.0, Device ID: 0), and a final message indicating all steps were completed successfully. The terminal prompt at the bottom is "[root@fedora synth]#".

```
student@fedora:~/netfpga/projects/lab4/synth
File Edit View Search Terminal Help
Processing /root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/xml/sram_rr_output_queues.xml...
WARNING: No module specific XML found for module 'core/sram_arbiter/sram_weighted_rr'
WARNING: No module specific XML found for module 'core/user_data_path/reference_user_data_path'
Processing /root/netfpga/lib/verilog/core/io/mdio/xml/mdio.xml...
WARNING: No module specific XML found for module 'core/cpci_bus'
Processing /root/netfpga/lib/verilog/core/dma/xml/dma.xml...
WARNING: No module specific XML found for module 'core/user_data_path/udp_reg_master'
WARNING: No module specific XML found for module 'core/io_queues/add_rm_hdr'
Processing /root/netfpga/lib/verilog/core/strip_headers/keep_length/xml/strip_headers.xml...
Processing /root/netfpga/lib/verilog/core/utils/xml/device_id_reg.xml...
WARNING: No module specific XML found for module 'core/utils/generic_regs'
Processing /root/netfpga/projects/lab4/include/ids.xml...
Processed registers.
+++testFinished:build.registers.lab4
Made cores.
+++testStarted:build.registers.cpci
Switch will be removed from the Perl core distribution in the next major release. Please install it from CPAN. It is being used at /root/netfpga/bin/nf_register_gen.pl, line 16.

NetFPGA environment:
  Root dir:      /root/netfpga
  Project name:  cpci
  Project dir:   /root/netfpga/projects/cpci
  Work dir:     /tmp/student

Processing /root/netfpga/lib/verilog/core/common/xml/global.xml...
Processing /root/netfpga/lib/verilog/core/common/xml/nfDefines.xml...
Processing /root/netfpga/projects/cpci/include/project.xml...

Project: 'CPCI' (cpci)
Description: NetFPGA PCI interface
Version: 4.1.0
Device ID: 0

Processing /root/netfpga/projects/cpci/include/cpci_REGS.xml...
Processed registers.
+++testFinished:build.registers.cpci
Made all...
[root@fedora synth]#
```

From the figure below, you can clearly see that we were able to successfully generate the bitfile. The mentioned bitfile that was generated was '***nf2\_top\_par.bit***'

This is the generated bitfile from the 'make' command as the output.

```
File Edit View Search Terminal Help
4 -rwxrwxrwx 1 root root      814 Feb 6 18:16 netlist.lst
84 -rwxrwxrwx 1 root root    85608 Feb 6 18:16 nf2_top.bld
4 -rwxrwxrwx 1 root root       5 Feb 6 18:10 nf2_top.lso
4 -rwxrwxrwx 1 root root    3438 Feb 6 18:19 nf2_top.map
164 -rwxrwxrwx 1 root root   166686 Feb 6 18:19 nf2_top_map.xrpt
1484 -rwxrwxrwx 1 root root  1518448 Feb 6 18:19 nf2_top.mrp
7280 -rwxrwxrwx 1 root root  7452463 Feb 6 18:12 nf2_top.ngc
12 -rwxrwxrwx 1 root root   11693 Feb 6 18:12 nf2_top.ngc_xst.xrpt
15572 -rwxrwxrwx 1 root root 15944723 Feb 6 18:16 nf2_top.ngd
12 -rwxrwxrwx 1 root root   11928 Feb 6 18:16 nf2_top_ngdbuild xrpt
36748 -rwxrwxrwx 1 root root 37627080 Feb 6 18:16 nf2_top.ngm
44 -rwxrwxrwx 1 root root   44578 Feb 6 18:22 nf2_top_par.bgn
2324 -rwxrwxrwx 1 root root  2377763 Feb 6 18:22 nf2_top_par.bit
40 -rwxrwxrwx 1 root root   38161 Feb 6 18:22 nf2_top_par.drc
1112 -rwxrwxrwx 1 root root  1135059 Feb 6 18:21 nf2_top_par.grf
9012 -rwxrwxrwx 1 root root  9226568 Feb 6 18:21 nf2_top_par.ncd
60 -rwxrwxrwx 1 root root   59158 Feb 6 18:21 nf2_top_par.pad
60 -rwxrwxrwx 1 root root   59190 Feb 6 18:21 nf2_top_par_pad.csv
232 -rwxrwxrwx 1 root root   234589 Feb 6 18:21 nf2_top_par_pad.txt
16 -rwxrwxrwx 1 root root   13658 Feb 6 18:21 nf2_top_par.par
24 -rwxrwxrwx 1 root root   22555 Feb 6 18:21 nf2_top_par.ptwx
68 -rwxrwxrwx 1 root root   65871 Feb 6 18:22 nf2_top_par.twr
152 -rwxrwxrwx 1 root root   152950 Feb 6 18:22 nf2_top_par.twx
4 -rwxrwxrwx 1 root root    159 Feb 6 18:21 nf2_top_par.unroutes
4 -rwxrwxrwx 1 root root    43 Feb 6 18:21 nf2_top_par.xpi
4 -rwxrwxrwx 1 root root   2290 Feb 6 18:21 nf2_top_par.xrpt
2932 -rwxrwxrwx 1 root root  3000805 Feb 6 18:19 nf2_top.pcf
12 -rwxrwxrwx 1 root root   10212 Feb 6 18:07 nf2_top.prj
4 -rwxrwxrwx 1 root root   1144 Feb 6 18:19 nf2_top.psr
4 -rwxrwxrwx 1 root root   893 Feb 6 02:01 nf2_top.scr
888 -rwxrwxrwx 1 root root  907228 Feb 6 18:12 nf2_top.srp
4 -rwxrwxrwx 1 root root   404 Feb 6 18:22 nf2_top_summary.xml
124 -rwxrwxrwx 1 root root  124949 Feb 6 17:26 nf2_top.ucf
52 -rwxrwxrwx 1 root root  51332 Feb 6 18:22 nf2_top_usage.xml
64 -rwxrwxrwx 1 root root  62885 Feb 6 02:01 pci2net_16x60.ngc
156 -rwxrwxrwx 1 root root  159494 Feb 6 02:01 rxfifo_8kx9_to_36.ngc
188 -rwxrwxrwx 1 root root  189250 Feb 6 02:01 rxfifo_8kx9_to_72.ngc
56 -rwxrwxrwx 1 root root  56070 Feb 6 02:01 rxlengthfifo_128x13.ngc
9608 -rwxrwxrwx 1 root root  9837596 Feb 6 18:16 smartguide.ncd
```

When we used the 'ls' command we were able to clearly see our generated bitfile in the lab4/synth/folder. See the snippet below for details.

```

File Edit View Search Terminal Help
[root@fedora synth]# ls
async_fifo_256x72_to_36.ngc      nf2_top.map          nf2_top_par.twr      srl_cam_32x32.ngc
async_fifo_512x36_progfull_500.ngc nf2_top_map.xrpt   nf2_top_par.twx      srl_cam_64x32.ngc
async_fifo_512x36_to_72_progfull_500.ngc nf2_top_mrp    nf2_top_par.unroutes srl_cam_unencoded_32x32.ngc
bram_cam_32x32.ngc                nf2_top.ngc        nf2_top_par.xpi      syncfifo_512x32.ngc
bram_cam_64x32.ngc                nf2_top.ngc_xst.xrpt nf2_top_par.xrpt      syncfifo_512x36_fallthrough.ngc
bram_cam_unencoded_32x32.ngc       nf2_top.ngd       nf2_top.pcf       syncfifo_512x36.ngc
cam_16x32.ngc                     nf2_top_ngdbuild.xrpt nf2_top.prj       syncfifo_512x72.ngc
cdq_rx_fifo_512x36.ngc            nf2_top.ngm       nf2_top.psr       synth.txt
cdq_rx_fifo_512x36_to_72.ngc     nf2_top_par.bgn   nf2_top.scr       transcript
cdq_tx_fifo_256x72_to_36.ngc     nf2_top_par.bit   nf2_top.srp       tri_mode_eth_mac.ngc
cdq_tx_fifo_512x36.ngc           nf2_top_par.drc   nf2_top_summary.xml txfifo_1024x36_to_9.ngc
dualportmemory.ngc                nf2_top_par.grf   nf2_top.ucf       txfifo_512x72_to_9.ngc
hdr_fifo.ngc                      nf2_top_par.ncd   nf2_top_usage.xml  xlnx_auto_0.ise
Makefile                          nf2_top_par.pad   pci2net_16x60.ngc xlnx_auto_0.xdb
net2pci_16x32.ngc                nf2_top_par_pad.csv rx fifo_8kx9_to_36.ngc
netlist.lst                        nf2_top_par_pad.txt rx fifo_8kx9_to_72.ngc
nf2_top.bld                        nf2_top_par.par  rxlengthfifo_128x13.ngc
nf2_top.lso                         nf2_top_par.ptwx smartguide.ncd
[root@fedora synth]#

```

We also took screenshots of a few errors we encountered while running the 'make' command.

```

344 *          Design Hierarchy Analysis
345 =====
346 ERROR:HDLCompilers:209 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Count operand of multiple concatenation operator is negative.
347 ERROR:HDLCompilers:185 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Illegal right hand side of continuous assign.
348 ERROR:HDLCompilers:209 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Count operand of multiple concatenation operator is negative.
349 ERROR:HDLCompilers:185 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Illegal right hand side of continuous assign.
350 ERROR:HDLCompilers:209 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Count operand of multiple concatenation operator is negative.
351 ERROR:HDLCompilers:185 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Illegal right hand side of continuous assign.
352 ERROR:HDLCompilers:209 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Count operand of multiple concatenation operator is negative.
353 ERROR:HDLCompilers:185 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Illegal right hand side of continuous assign.
354 ERROR:HDLCompilers:209 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Count operand of multiple concatenation operator is negative.
355 ERROR:HDLCompilers:185 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Illegal right hand side of continuous assign.
356 ERROR:HDLCompilers:209 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Count operand of multiple concatenation operator is negative.
357 ERROR:HDLCompilers:185 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Illegal right hand side of continuous assign.
358 ERROR:HDLCompilers:209 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Count operand of multiple concatenation operator is negative.
359 ERROR:HDLCompilers:185 - "/root/netfpga/lib/verilog/core/output_queues/sram_rr_output_queues/src/oq_regs_generic_reg.grp.
v" line 402 Illegal right hand side of continuous assign.
360 =====
347 ERROR:HDLCompilers:88 - "../src/fallthrough_small_fifo.v2.v" line 41 Parameter 'PROG_FULL_THRESHOLD' does not exist in module 'small_fifo'.
348 ERROR:HDLCompilers:88 - "../src/fallthrough_small_fifo_v2.v" line 41 Parameter 'PROG_FULL_THRESHOLD' does not exist in module 'small_fifo'.
349 ERROR:HDLCompilers:88 - "../src/fallthrough_small_fifo_v2.v" line 41 Parameter 'PROG_FULL_THRESHOLD' does not exist in module 'small_fifo'.
350 --> .
351 .

```

```

10982 .
10983 Checking expanded design ....
10984 ERROR:NgdBuild:604 - logical block.
10985   'nf2_core/user_data_path/ids/drop_fifo/XLXI_34' with type 'dualportmemory'.
10986   could not be resolved. A pin name misspelling can cause this, a missing edif.
10987   or ngc file, or the misspelling of a type name. Symbol 'dualportmemory' is.
10988   not supported in target 'virtex2p'..
10989 WARNING:NgdBuild:452 - logical net 'nf2_core/core_256kb_0_reg_addr(100)' has no.
10990   driver.
10991 WARNING:NgdBuild:452 - logical net 'nf2 core/core 256kb 0 req addr(101)' has no.

```

We also are uploading the bitfile generation summary & timing report as a proof of work it took us to debug each of the errors while running the make command.

```

2180
2181
2182 Timing summary:
2183 -----
2184
2185 Timing errors: 0 Score: 0
2186
2187 Constraints cover 835729 paths, 4 nets, and 103291 connections
2188
2189 Design statistics:
2190   Minimum period: 11.622ns (Maximum frequency: 86.044MHz)
2191   Maximum path delay from/to any node: 1.850ns
2192   Maximum net delay: 0.603ns
2193   Minimum input required time before clock: 2.865ns
2194   Minimum output required time after clock: 4.689ns
2195
2196
2197 Analysis completed Thu Feb 6 18:22:11 2025
2198 -----
2199
2200 Generating Report...
2201
2202 Number of warnings: 0
2203 Number of info messages: 3
2204 Total time: 27 secs
2205 +++testFinished:build.trce.nf2_top_par.twr
2206 =====
2207 === Generate bit stream file from ncd
2208 =====
2209 +++testStarted:build.bitgen.nf2_top_par.bit
2210 bitgen -intstyle ise -g Binary:No -w nf2_top_par.ncd
2211 WARNING:PhysDesignRules:1060 - Dangling pins on
2212   block:<nf2_core/cpu_queues[0].cpu_dma_queue_i/cpu_dma_rx_queue/cpu_fifos64.rx
2213   _fifo/BU2/U0/grf_rf/mem/gbm_gbmg_gbmg_gbmg_ngecc_bmg_blk_mem_generator/valid.cstr
2214   /ramloop[0].ram.r/v2.ram/dp18x36.ram.A>:<RAMB16_RAMB16A>. The block is
2215   configured to use input parity pins. There are dangling output parity pins.
2216 WARNING:PhysDesignRules:1060 - Dangling pins on
2217   block:<nf2_core/cpu_queues[1].cpu_dma_queue_i/cpu_dma_rx_queue/cpu_fifos64.rx
2218   _fifo/BU2/U0/grf_rf/mem/gbm_gbmg_gbmg_gbmg_ngecc_bmg_blk_mem_generator/valid.cstr
2219   /ramloop[0].ram.r/v2.ram/dp18x36.ram.A>:<RAMB16_RAMB16A>. The block is
2220   configured to use input parity pins. There are dangling output parity pins.
2221 WARNING:PhysDesignRules:1060 - Dangling pins on

```

```
2793 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2794 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2795 ✓ WARNING:PhysDesignRules:812 - Dangling pin <DOB22> on
2796 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2797 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2798 ✓ WARNING:PhysDesignRules:812 - Dangling pin <DOB23> on
2799 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2800 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2801 ✓ WARNING:PhysDesignRules:812 - Dangling pin <DOB24> on
2802 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2803 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2804 ✓ WARNING:PhysDesignRules:812 - Dangling pin <DOB25> on
2805 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2806 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2807 ✓ WARNING:PhysDesignRules:812 - Dangling pin <DOB26> on
2808 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2809 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2810 ✓ WARNING:PhysDesignRules:812 - Dangling pin <DOB27> on
2811 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2812 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2813 ✓ WARNING:PhysDesignRules:812 - Dangling pin <DOB28> on
2814 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2815 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2816 ✓ WARNING:PhysDesignRules:812 - Dangling pin <DOB29> on
2817 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2818 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2819 ✓ WARNING:PhysDesignRules:812 - Dangling pin <DOB30> on
2820 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2821 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2822 ✓ WARNING:PhysDesignRules:812 - Dangling pin <DOB31> on
2823 ...block:<nf2_core/user_data_path/output_queues/oq_regs/oq_reg_instances/oq_addr
2824 ..._hi_reg/ram/Mram_ram.B>:<RAMB16_RAMB16B>.
2825 +++testFinished:build.bitgen.nf2_top_par.bit
2826 Made all...
2827 rm nf2_top.ncd
```

### Part-3: Set Up VPN to USC

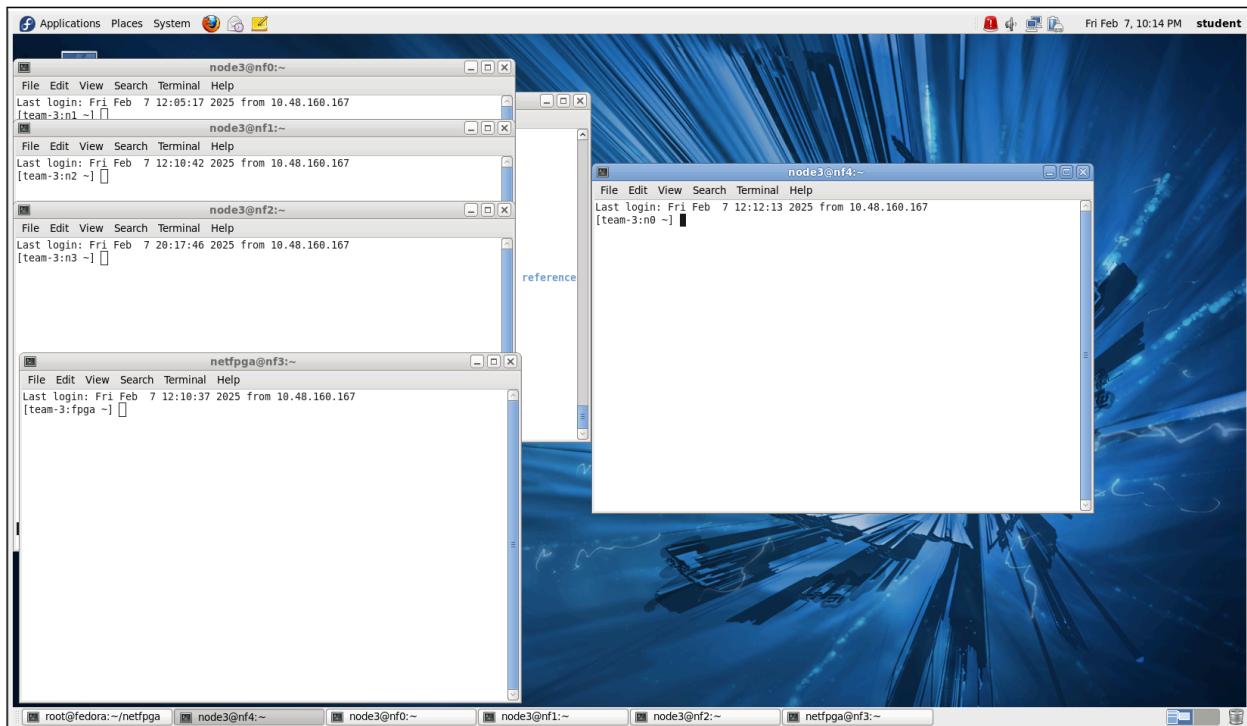
We successfully set up the VPN at [vpn.usc.edu](https://vpn.usc.edu)



## Part-4: NetFPGA Environment

1. Openssh & nano were installed using:  
`sudo apt-get install openssh-client nano`
2. Ssh client was edited using nano editor:  
`sudo nano /etc/ssh/ssh_config`
3. Following text was added:  
`HostkeyAlgorithms ssh-dss,ssh-rsa  
KexAlgorithms +diffie-hellman-group1-sha1  
StrictHostKeyChecking=accept-new`
4. After creating the executable file and connecting to vpn.  
`openterm <TEAM #> <TEAM_PASSWORD>`  
**[NOTE: We are Team#4 but the Team number assigned to us is Team#3]**

As depicted in the following snippet, we were able to execute openterm and launch the NetFPGA node and 4 nodes that are connected to NetFPGA.



In order to better understand the openterm:  
We have added a short explanation for this openterm.

```
#!/bin/bash
if [ "$#" -le 1 ]; then
    echo "$0 <TEAM #> <PASSWORD>"
```

```

    exit 0
fi
if [ "$#" -le 1 ]; then ... fi → Ensures the script receives at least two
Arguments:

```

- **\$1 (TEAM #)** → A team number.
- **\$2 (PASSWD)** → A password for SSH authentication.

\*\*

```

ND=$((1 % 5))      # Compute remainder when TEAM # is divided by 5
LL=$((1 + 1))       # Increment TEAM #
CM=$LL"q;d"         # Unused variable (concatenates TEAM + "q;d")
PW=$2                # Stores the SSH password
CL=$(( ($1 / 5) * 5 )) # Rounds TEAM # down to the nearest multiple of 5

```

```

ND=$(( $1 % 5 ))      → Computes a node index (0-4) based on the team number.
LL=$(( $1 + 1 ))        → Adds 1 to the team number (possibly used for reference).
CM=$LL"q;d"            → Seems unused; it creates a string with team number + "q;d".
PW=$2                  → Stores the SSH password.
CL=$(( ($1 / 5) * 5 )) → Computes the nearest multiple of 5 for team number.

```

\*\*

```

M0=$((ND + CL))
M1=$(( ($ND + 1) % 5) + CL)
M2=$(( ($ND + 2) % 5) + CL)
M3=$(( ($ND + 3) % 5) + CL)
M4=$(( ($ND + 4) % 5) + CL))

```

\* This calculates five machine numbers (**M0** – **M4**) that the script will SSH into.

\* Each **M#** represents a different machine ID in a cyclic manner.

\* It ensures the machines are assigned within groups of five (**CL** helps keep machines within the same batch).

\*\*

```

gnome-terminal -- bash -c "sshpass -p $PW ssh -o StrictHostKeyChecking=no
netfpga@nf$M0.usc.edu" &

```

```

gnome-terminal -- bash -c "sshpass -p $PW ssh -o StrictHostKeyChecking=no
node$ND@nf$M1.usc.edu" &

```

```
gnome-terminal -- bash -c "sshpass -p $PW ssh -o StrictHostKeyChecking=no node$ND@nf$M2.usc.edu" &
```

```
gnome-terminal -- bash -c "sshpass -p $PW ssh -o StrictHostKeyChecking=no node$ND@nf$M3.usc.edu" &
```

```
gnome-terminal -- bash -c "sshpass -p $PW ssh -o StrictHostKeyChecking=no node$ND@nf$M4.usc.edu" &
```

Opens five separate GNOME terminal windows, each executing an SSH session.

**sshpass -p \$PW ssh ...** → Uses the given password (**\$PW**) for SSH authentication.

**-o StrictHostKeyChecking=no** → Disables SSH's host key verification prompt.

Targets machines:

- First SSH login is for **netfpga@nf\$M0.usc.edu**.
- The next four logins are for **node\$ND@nfM1.usc.edu**, **node\$ND@nfM2.usc.edu**, etc.

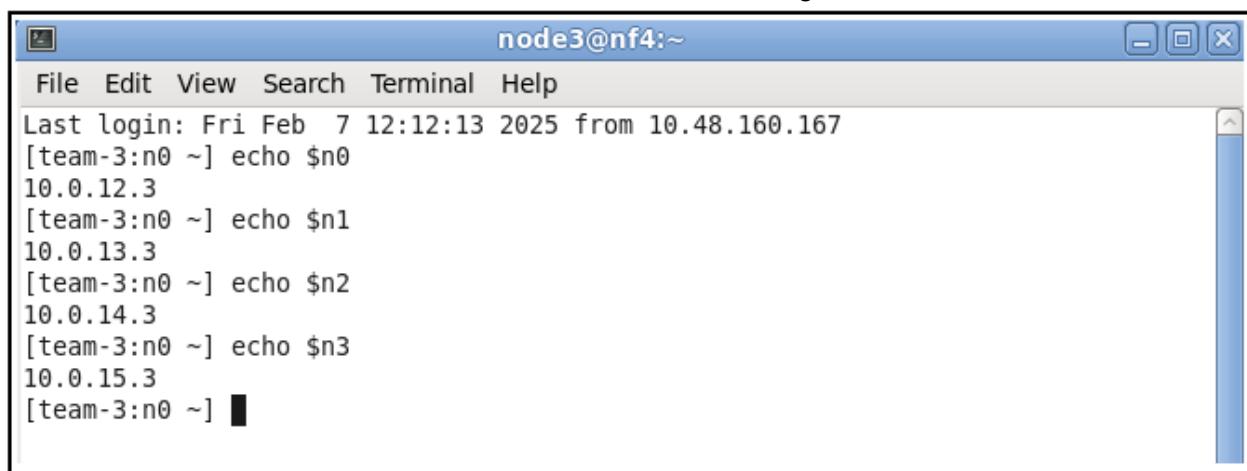
Each command ends with **&**, which runs it in the background so that all SSH sessions open simultaneously.

\*\*

For TEAM # = 3, the script:

- ✓ Computes machine IDs: 3, 4, 0, 1, 2.
- ✓ Uses password authentication with **sshpass**.
- ✓ Opens five GNOME terminal windows, each running an SSH session.
- ✓ Logs into a mix of **netfpga** and **node3** user accounts on different machines.

We were also able to find the IP addresses of each node using “echo”.



```
node3@nf4:~
```

```
File Edit View Search Terminal Help
Last login: Fri Feb  7 12:12:13 2025 from 10.48.160.167
[team-3:n0 ~] echo $n0
10.0.12.3
[team-3:n0 ~] echo $n1
10.0.13.3
[team-3:n0 ~] echo $n2
10.0.14.3
[team-3:n0 ~] echo $n3
10.0.15.3
[team-3:n0 ~] █
```

## Part-5: NetFPGA-based Linux Kernel IP Router

On the FPGA node, we loaded the bitstream of the reference\_nic design.

```
netfpga@nf3:~$ Last login: Fri Feb 7 12:10:37 2025 from 10.48.160.167 [team-3:fpga ~] nf_download /home/netfpga/bitfiles/reference_nic.bit Found net device: nf2c0 Bit file built from: nf2_top_par.ncd;HW_TIMEOUT=FALSE Part: 2vp50ff1152 Date: 2011/11/17 Time: 16:21:17 Error Registers: 0 Good, after resetting programming interface the FIFO is empty Download completed - 2377668 bytes. (expected 2377668). DONE went high - chip has been successfully programmed. CPCI Information ----- Version: 4 (rev 1) Device (Virtex) Information ----- Project directory: reference_nic Project name: Reference NIC Project description: Reference NIC Device ID: 1 Version: 1.1.0 Built against CPCI version: 4 (rev 1) Virtex design compiled against active CPCI version [team-3:fpga ~]
```

We were also able to ping all nodes from each node. For instance, we were able to ping n3 from n0.

```
node3@nf4:~$ File Edit View Search Terminal Help [team-3:n0 ~] ping $n3 PING 10.0.15.3 (10.0.15.3) 56(84) bytes of data. 64 bytes from 10.0.15.3: icmp_seq=1 ttl=63 time=0.853 ms 64 bytes from 10.0.15.3: icmp_seq=2 ttl=63 time=0.911 ms 64 bytes from 10.0.15.3: icmp_seq=3 ttl=63 time=0.969 ms --- 10.0.15.3 ping statistics --- 3 packets transmitted, 3 received, 0% packet loss, time 2002ms rtt min/avg/max/mdev = 0.853/0.911/0.969/0.047 ms [team-3:n0 ~]
```

We then executed iperf commands to test the connectivity between nodes. In the following snippet, n2 is the server and n1 is the client. From the snippet, we note that the bandwidth is about 77.8 Mbits/sec.

```
node3@nf1:~$ File Edit View Search Terminal Help Last login: Fri Feb 7 12:10:42 2025 from 10.48.160.167 [team-3:n2 ~] iperf -s ----- Server listening on TCP port 5001 TCP window size: 128 KByte (default) ----- [ 4] local 10.0.14.3 port 5001 connected with 10.0.13.3 port 47068 [ ID] Interval Transfer Bandwidth [ 4] 0.0- 5.4 sec 49.8 MBytes 77.8 Mbits/sec [team-3:n2 ~] $ node3@nf0:~$ File Edit View Search Terminal Help Last login: Fri Feb 7 12:05:17 2025 from 10.48.160.167 [team-3:n1 ~] iperf -c 10.0.14.3 ----- Client connecting to 10.0.14.3, TCP port 5001 TCP window size: 16.0 KByte (default) ----- [ 3] local 10.0.13.3 port 47068 connected with 10.0.14.3 port 5001 [ ID] Interval Transfer Bandwidth [ 3] 0.0- 5.3 sec 49.8 MBytes 79.0 Mbits/sec [team-3:n1 ~] $
```

## Part-6: NetFPGA Hardware IP Router

1. This is the part where the NetFPGA was tested as a hardware IP router.
2. Load the reference router into the NetFPGA. What does the reference router do? How should this help?

ANS: The reference router extracts all the information from the Linux Kernel to the routing table of the NetFPGA. This is done to accelerate the routing process.

```
netfpga@nf3:~
```

```
File Edit View Search Terminal Help
Date: 2011/11/17
Time: 16:21:17
Error Registers: 0
Good, after resetting programming interface the FIFO is empty
Download completed - 2377668 bytes. (expected 2377668).
DONE went high - chip has been successfully programmed.
CPCI Information
-----
Version: 4 (rev 1)

Device (Virtex) Information
-----
Project directory: reference_nic
Project name: Reference NIC
Project description: Reference NIC

Device ID: 1
Version: 1.1.0
Built against CPCI version: 4 (rev 1)

Virtex design compiled against active CPCI version
[team-3:fpga ~] nf_download /home/netfpga/bitfiles/reference_router.bit
Found net device: nf2c0
Bit file built from: nf2_top_par.ncd;HW_TIMEOUT=FALSE
Part: 2vp50ff1152
Date: 2011/11/17
Time: 17:49:43
Error Registers: 0
Good, after resetting programming interface the FIFO is empty
Download completed - 2377668 bytes. (expected 2377668).
DONE went high - chip has been successfully programmed.
CPCI Information
-----
Version: 4 (rev 1)

Device (Virtex) Information
-----
Project directory: reference_router
Project name: Reference router
Project description: Reference IPv4 router

Device ID: 2
Version: 1.0.0
Built against CPCI version: 4 (rev 1)

Virtex design compiled against active CPCI version
[team-3:fpga ~] rkd &
[1] 13159
[team-3:fpga ~] █
```

We then loaded the reference\_router bitstream onto the fpga and ran “rkd &” to load all of the needed information from the Linux kernel into the routing table of the NetFPGA in order to accelerate the routing process.

- Now re-run the iperf tests. Has the bandwidth changed? If so, why?

*ANS:* When we re-run the commands identify the acceleration. The measured bandwidth of the server was 354 Mbits/sec. In comparison, this is way higher than the execution of ‘iperf’ when reference nic was loaded 77.8 Mbits/sec

```
node3@nf1:~
```

```
[team-3:n2 ~] iperf -s
-----
Server listening on TCP port 5001
TCP window size: 128 KByte (default)
-----
[ 4] local 10.0.14.3 port 5001 connected with 10.0.13.3 port 57612
[ ID] Interval Transfer Bandwidth
[ 4] 0.0- 6.8 sec 286 MBytes 354 Mbits/sec
[team-3:n2 ~]
```

```
node3@nf0:~
```

```
[team-3:n1 ~] iperf -c 10.0.14.3
-----
Client connecting to 10.0.14.3, TCP port 5001
TCP window size: 16.0 KByte (default)
-----
[ 3] local 10.0.13.3 port 57612 connected with 10.0.14.3 port 5001
[ ID] Interval Transfer Bandwidth
[ 3] 0.0- 6.7 sec 286 MBytes 356 Mbits/sec
[team-3:n1 ~]
```

Now that we have the reference\_router loaded, we re-run the ‘iperf’ commands to see the acceleration. In the snippet above, the bandwidth according to the server is 354 Mbits/sec. This is significantly higher than the execution of ‘iperf’ when reference\_nic was loaded (77.8 Mbits/sec).

We then created two shell scripts on the fpga node, “udpclients\_launch.sh” and “udpservers\_launch.sh”. The server script iterates through a loop that SSHes to each node (n0-n3) to launch a UDP server on each of them. The client script iterates through a loop that SSHes to each node and launches a client on each node. Each client that is launched will then connect to a server that is on a different node (i.e. client n3 → server n0, client n2 → server n1, etc.). Additionally, the client script specifies a bandwidth of 1G, a packet size of 512 bytes, and a test duration. Both scripts write the results into a separate log file, one for the results from the client’s side and one from the server’s side.

Using the reference\_nic bitstream, here are the results after running the two scripts.

- Make sure to run the test several times and take the average results. Look at the output of the servers to see how much performance you are able to attain. What is the total bandwidth you are able to observe through the NetFPGA? Why does using the small packets stress the system?

*ANS:* From the snippet below, we can see that 91% of the datagrams are lost. Additionally, the total bandwidth is 86.5 Mbits/sec (adding up the bandwidth of each client node). Small packets stress the system due to increased packet processing. This limits the throughput and results in packet loss as the system cannot keep up with the incoming data rate. Additionally, small packets might lead to more frequent use of network buffers, leading to packet loss if buffer overflow occurs,

```
netfpga@nf3:~/iperf_scripts
```

File Edit View Search Terminal Help

```
-----
```

Server listening on UDP port 5004  
Receiving 1470 byte datagrams  
UDP buffer size: 2.00 MByte (default)

```
-----
```

Server listening on UDP port 5001  
Receiving 1470 byte datagrams  
UDP buffer size: 2.00 MByte (default)

```
-----
```

Server listening on UDP port 5003  
Receiving 1470 byte datagrams  
UDP buffer size: 2.00 MByte (default)

```
-----
```

Server listening on UDP port 5002  
Receiving 1470 byte datagrams  
UDP buffer size: 2.00 MByte (default)

```
-----
```

[ 3] local 10.0.12.3 port 5001 connected with 10.0.15.3 port 43894  
[ 3] local 10.0.13.3 port 5002 connected with 10.0.14.3 port 36022  
[ ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams  
[ 3] 0.0-30.3 sec 78.8 MBytes 21.8 Mbits/sec 2,447 ms 1659936/1821255 (91%)  
[ 3] 0.0-30.3 sec 1 datagrams received out-of-order  
[ ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams  
[ 3] 0.0-30.3 sec 79.1 MBytes 21.9 Mbits/sec 1,870 ms 1652785/1814802 (91%)  
[ 3] local 10.0.14.3 port 5003 connected with 10.0.13.3 port 48668  
[ ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams  
[ 3] 0.0-30.2 sec 78.5 MBytes 21.8 Mbits/sec 4,585 ms 1653382/1814210 (91%)  
[ 3] 0.0-30.2 sec 1 datagrams received out-of-order  
[ 3] local 10.0.15.3 port 5004 connected with 10.0.12.3 port 41238  
[ ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams  
[ 3] 0.0-30.3 sec 75.7 MBytes 21.0 Mbits/sec 6,284 ms 1658469/1813406 (91%)  
-----

```
-----
```

```
netfpga@nf3:~/iperf_scripts
```

File Edit View Search Terminal Help

GNU nano 1.3.12 File: clients nic results.log

```
-----
```

Client IP: 10.0.15.3 connecting to Server IP: 10.0.12.3 on Port 5001 ...  
Client IP: 10.0.14.3 connecting to Server IP: 10.0.13.3 on Port 5002 ...  
Client IP: 10.0.13.3 connecting to Server IP: 10.0.14.3 on Port 5003 ...  
Client IP: 10.0.12.3 connecting to Server IP: 10.0.15.3 on Port 5004 ...

```
-----
```

Client connecting to 10.0.12.3, UDP port 5001  
Sending 512 byte datagrams  
UDP buffer size: 2.00 MByte (default)

```
-----
```

[ 3] local 10.0.15.3 port 43894 connected with 10.0.12.3 port 5001  
-----

Client connecting to 10.0.13.3, UDP port 5002  
Sending 512 byte datagrams  
UDP buffer size: 2.00 MByte (default)

```
-----
```

[ 3] local 10.0.14.3 port 36022 connected with 10.0.13.3 port 5002  
-----

Client connecting to 10.0.14.3, UDP port 5003  
Sending 512 byte datagrams  
UDP buffer size: 2.00 MByte (default)

```
-----
```

Client connecting to 10.0.15.3, UDP port 5004  
Sending 512 byte datagrams  
UDP buffer size: 2.00 MByte (default)

```
-----
```

[ 3] local 10.0.12.3 port 41238 connected with 10.0.15.3 port 5004  
[ ID] Interval Transfer Bandwidth  
[ 3] 0.0-30.0 sec 886 MBytes 248 Mbits/sec  
[ 3] Sent 1814804 datagrams  
[ ID] Interval Transfer Bandwidth  
[ 3] 0.0-30.0 sec 889 MBytes 249 Mbits/sec  
[ 3] Sent 1821258 datagrams  
[ 3] Server Report:  
[ 3] 0.0-30.3 sec 78.8 MBytes 21.8 Mbits/sec 2,446 ms 1659936/1821255 (91%)  
[ 3] 0.0-30.3 sec 1 datagrams received out-of-order  
[ 3] Server Report:  
[ 3] 0.0-30.3 sec 79.1 MBytes 21.9 Mbits/sec 1,869 ms 1652785/1814802 (91%)  
[ ID] Interval Transfer Bandwidth  
[ 3] 0.0-30.0 sec 885 MBytes 248 Mbits/sec

5. Load the reference router.bit file again and start the ‘rkd’ daemon. Run the same iperf. What do you observe? Is it fair to say that the NetFPGA can route IP traffic bi-directionally at line-speed for a total of 4Gbps of cross-wise bandwidth?

ANS: We then loaded the reference\_router.bit file and ran the scripts again. Below are the output log files. We can see that there is less packet loss, but it is still bad with 83% packet loss. Additionally, the total bandwidth is 146.8 Mbps by adding all the bandwidths from the server output log. Hence, it is not fair to say that the NetFPGA can route IP traffic bi-directionally at line-speed for a total of 4Gbps of cross-wise bandwidth.

```
netfpga@nf3:~/iperf_scripts
```

File Edit View Search Terminal Help  
GNU nano 1.3.12 File: servers router results.log

```
-----  
Server listening on UDP port 5001  
Receiving 1470 byte datagrams  
UDP buffer size: 2.00 MByte (default)  
  
-----  
Server listening on UDP port 5004  
Receiving 1470 byte datagrams  
UDP buffer size: 2.00 MByte (default)  
  
-----  
Server listening on UDP port 5003  
Receiving 1470 byte datagrams  
UDP buffer size: 2.00 MByte (default)  
  
-----  
Server listening on UDP port 5002  
Receiving 1470 byte datagrams  
UDP buffer size: 2.00 MByte (default)  
  
[ 3] local 10.0.14.3 port 5003 connected with 10.0.13.3 port 50886  
[ 3] local 10.0.12.3 port 5001 connected with 10.0.15.3 port 47776  
[ 3] local 10.0.13.3 port 5002 connected with 10.0.14.3 port 57493  
[ 3] local 10.0.15.3 port 5004 connected with 10.0.12.3 port 42646  
[ ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams  
[ 3] 0.0-30.2 sec 152 MBytes 42.3 Mbits/sec 0.795 ms 1486491/1798050 (83%)  
[ ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams  
[ 3] 0.0-30.3 sec 122 MBytes 33.7 Mbits/sec 12.272 ms 1556694/1805728 (86%)  
[ 3] 0.0-30.3 sec 3 datagrams received out-of-order  
[ ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams  
[ 3] 0.0-30.1 sec 123 MBytes 34.2 Mbits/sec 0.809 ms 1510553/1762452 (86%)  
[ ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams  
[ 3] 0.0-30.2 sec 132 MBytes 36.6 Mbits/sec 12.132 ms 1524920/1794560 (85%)
```

-----  
Client IP: 10.0.15.3 connecting to Server IP: 10.0.12.3 on Port 5001 ...  
Client IP: 10.0.14.3 connecting to Server IP: 10.0.13.3 on Port 5002 ...  
Client IP: 10.0.13.3 connecting to Server IP: 10.0.14.3 on Port 5003 ...  
Client IP: 10.0.12.3 connecting to Server IP: 10.0.15.3 on Port 5004 ...  
  
-----  
Client connecting to 10.0.14.3, UDP port 5003  
Sending 512 byte datagrams  
UDP buffer size: 2.00 MByte (default)  
[ 3] local 10.0.13.3 port 50886 connected with 10.0.14.3 port 5003  
  
-----  
Client connecting to 10.0.15.3, UDP port 5004  
Sending 512 byte datagrams  
UDP buffer size: 2.00 MByte (default)  
[ 3] local 10.0.12.3 port 42646 connected with 10.0.15.3 port 5004  
[ ID] Interval Transfer Bandwidth  
[ 3] 0.0-30.0 sec 861 MBytes 241 Mbits/sec  
[ 3] Sent 1762454 datagrams  
  
-----  
Client connecting to 10.0.12.3, UDP port 5001  
Sending 512 byte datagrams  
UDP buffer size: 2.00 MByte (default)  
[ 3] local 10.0.15.3 port 47776 connected with 10.0.12.3 port 5001  
[ ID] Interval Transfer Bandwidth  
[ 3] 0.0-30.0 sec 878 MBytes 245 Mbits/sec  
[ 3] Sent 1798052 datagrams  
[ ID] Interval Transfer Bandwidth  
[ 3] 0.0-30.0 sec 882 MBytes 246 Mbits/sec  
[ 3] Sent 1805738 datagrams  
[ 3] Server Report:  
[ 3] 0.0-30.1 sec 123 MBytes 34.2 Mbits/sec 0.808 ms 1510553/1762452 (86%)  
[ 3] Server Report:  
[ 3] 0.0-30.3 sec 122 MBytes 33.7 Mbits/sec 12.271 ms 1556694/1805728 (86%)  
[ 3] 0.0-30.3 sec 3 datagrams received out-of-order  
  
-----  
Client connecting to 10.0.13.3, UDP port 5002  
Sending 512 byte datagrams  
UDP buffer size: 2.00 MByte (default)

## Team GitHub: EE533-TEAM3/EE533\_LAB\_PROJECTS/ Commits & Changes

Commits

History for EE533\_LAB\_PROJECTS / LAB4 on main

All users All time

-> Commits on Feb 8, 2025

iperf udp scripts with results in log files	JSantos6456 committed 31 minutes ago	61a2656			
adding file that needed edits for sim and synth	EE533-TEAM3 authored 1 hour ago	6d0de48			
bitfile (Main) for LAB4	EE533-TEAM3 authored 1 hour ago	abadedb			
Create placeholder_bitfile.txt	EE533-TEAM3 authored 1 hour ago	4d41f8b			
Include files	EE533-TEAM3 authored 1 hour ago	7a9569f			
Create placeholder_include.txt	EE533-TEAM3 authored 1 hour ago	5068477			
Adding src file folders	EE533-TEAM3 authored 1 hour ago	254d9f2			
Create placeholder_src.txt	EE533-TEAM3 authored 1 hour ago	a0446f5			
Add files via upload	EE533-TEAM3 authored 1 hour ago	a9a8671			
Add files via upload	EE533-TEAM3 authored 1 hour ago	b465dba			
uploading ngm file	EE533-TEAM3 authored 1 hour ago	04cc2c0			
Add files via upload	EE533-TEAM3 authored 1 hour ago	4d8b613			
Create placeholder_synth.txt	EE533-TEAM3 authored 1 hour ago	b6846be			
Create placeholder_Code_For_Netfpga.txt	EE533-TEAM3 authored 1 hour ago	7e2f45f			
Create placeholder.txt	EE533-TEAM3 authored 1 hour ago	53638b8			

-> End of commit history for this file