P51 - Lab 2
Introduction to NetFPGA - Part 2

Dr Noa Zilberman

Lent, 2017/18

The goal of this lab is to introduce you to the NetFPGA register infrastructure, as well as to the test environment, providing hands on experience in NetFPGA Development.

1 Development Machines

This week you will continue and use the machines assigned to you last week. All the machines are located in the Practical Classroom (SW02).

You will interact with the machines via ssh:

1. On a computer in the Practical Classroom, log in using your own UIS credentials.
2. `ssh -X root@<hostname>.nf.cl.cam.ac.uk` and enter the password. Hosts ending in .cl.cam.ac.uk are permitted to ssh into these machines. `-X` enables X11 forwarding, allowing you to run graphical applications.
   To ssh to the machines from outside the lab, follow the instructions on `https://www.cl.cam.ac.uk/local/sys/ssh/`.

Hostname	IP Address
nf-test101	128.232.82.84
nf-test108	128.232.82.78
nf-test110	128.232.82.93
nf-test111	128.232.82.81

Important: The IP addresses noted above should not be used for anything except for communication with the machines. The network interfaces assigned for the tests use different IP addresses.

2 Practical Instructions

This section provides step-by-step instructions how to add a new register to your design. To this end, we will be using the Reference Switch design studied in class.

2.1 Accessing the board

1. Login to the development machine:

   ```
   $ ssh root@<hostname>.nf.cl.cam.ac.uk
   ```

2. Pull the latest NetFPGA release:

   ```
   $ cd ~/P51/NetFPGA-SUME-live/
   $ git pull
   ```

3. ```
   $ cd tools
   $ vim settings.sh
   ```

4. Make sure that NF_PROJECT_NAME is set to "reference_switch"

5. Load the environment settings:

   ```
   $ source settings.sh
   ```

6. Compile all cores:

   ```
   $ cd $SUME_FOLDER
   $ make
   ```

2.2 Adding a new register

The following instructions use the spreadsheet. You can alternatively use the csv file, combined with the script *csv_gen.py*. Then skip to step 7.

1. In Libreoffice, set security to medium.

2. Open tools/infrastructure/module_generation.xls or
   $IP_FOLDER/module_name/data/module_generation.xls

3. Change block name to match your module name (for sub-module this is optional)

4. Delete all indirect registers (and others you dont want) (note potential issues in some releases)

5. Change OS to Linux

6. Press Generate Registers

7. From console, run:

   ```
   $ python regs_gen.py
   $ cp *.v $IP_FOLDER/<module_name>/hdl
   ```
   ```

8. Copy data files to the data folder:

   ```
   $ cp <*.tcl,*.h,*.txt> $IP_FOLDER/module_name/data
   ```

9. Copy the lines from the template file to <module_name>.v

10. Add in <module_name>.v support for the register functionality

11. Compiling your IP core:

    ```
    cd $IP_FOLDER/<ip core name>
    make
    ```

12. Update the functional test to read (or write) your register:

    ```
    $ cd $NF_DESIGN_DIR/test/both_learning_sw
    $ vim run.py
    ```

13. Run a simulation of your design and test that it works:

    ```
    cd $SUME_FOLDER/tools/scripts
    ./nf_test.py sim --major learning --minor sw
    ```

3 Building a project

The following steps are typically required when building a project. We recommend that you follow them to test your design on the hardware:

- Compiling an IP core:

  ```
  cd $IP_FOLDER/<ip core name>
  make
  ```

  This step is required only for new IP cores or when changes are made to the tcl file of the core. There is no need to run make if only the HDL files were modified.

- Compiling CAM/TCAM cores:
  Follows the instructions on `https://github.com/NetFPGA/NetFPGA-SUME-public/wiki/NetFPGA-SUME-TCAM-IPs`
  The CAM core is required for building the NetFPGA project. You only need to run this step once.

- Compiling all cores and building libraries:

  ```
  cd $SUME_FOLDER
  make
  ```

This step if typically required only once: after the git repository is cloned or pulled. It is also required if the *make clean* command was called.

- Building a project:

```
cd $NF_DESIGN_DIR
make
```

The result of this step is the programming (bit) file. *This step takes 45 minutes or more. Do not run it during class.*

- Follow the instructions for hardware testing from Lab 1.

4 Common Issues

- Problem: Vivado's GUI does not open.
  Solution: 1) make sure to ssh using -X.
  2) edit ~/.bashrc. comment the line *export DISPLAY=:0*.

- Problem: Receiving extra or unexpected packets in the hardware test.
  Solution: this is likely as the network manager is not disabled and/or the network is not configured properly. Follow the steps in `https://github.com/NetFPGA/NetFPGA-SUME-public/wiki/Reference-Operating-System-Setup-Guide`, under the *Network Configuration Manager* section. In particular, make sure that your NIC's MAC address appears on the list of non managed interfaces (/etc/NetworkManager/NetworkManager.conf) and that **all** your NIC's host interfaces are set to manual (/etc/network/interfaces).

5 Useful links

- NetFPGA Repository: `https://github.com/NetFPGA/NetFPGA-SUME-live/`
- NetFPGA Wiki: `https://github.com/NetFPGA/NetFPGA-SUME-public/wiki`
- NetFPGA registration page: `https://netfpga.org/site/#/SUME_reg_form/`