Selected Areas in Cryptography Summer School – Hardware tutorial

August 13, 2019, Waterloo, Canada

Nele Mentens (KU Leuven)
with input from Pedro Maat Costa Massolino (Radboud University)

PRE-TUTORIAL INSTALLATION GUIDELINES

1. Install GHDL

GHDL is an open-source command-line simulator for the VHDL language: http://ghdl.free.fr

To install GHDL, follow these steps:
- Step 1: Download the installation files from https://github.com/ghdl/ghdl/releases for the operating system of your choice. E.g., if you are using Windows, download "ghdl-0.36-mingw32-mcode.zip". The following steps explain the setup for Windows; similar steps can be followed for other operating systems. In case you cannot find the version for your operating system, you can build it by following the mcode instructions at this link: http://ghdl.readthedocs.io/en/latest/building/Building.html
- Step 2: Unzip the downloaded file.
- Step 3: The folder "<download_location>\GHDL\0.36-mingw32-mcode\bin" contains the executable "ghdl.exe". Add the location of this folder to the "Path" system variable such that you can use the "ghdl" command from anywhere. E.g., in Windows 7, you can do this by right-clicking the "Computer" icon (on your desktop, in the start menu, or in an explorer window) and going to "Properties" → "Advanced System Settings" → "Advanced" tab → "Environment variables". Highlight the "Path" variable in the "System variables" section and click the "Edit" button. Add "<download_location>\GHDL\0.36-mingw32-mcode\bin;" to the path.

2. Install GTKWave

GTKWave is an open-source waveform viewer: http://gtkwave.sourceforge.net

Download the installation files from https://sourceforge.net/projects/gtkwave/files for the operating system of your choice. Follow the same steps as for the setup of GHDL. Make sure to add "<download_location>\gtkwave-3.3.100-bin-win64\bin" (in the case of a Windows install) to the Path such that the "gtkwave" command can be executed from anywhere.

3. Run an example program

To verify that GHDL and GTKWave are working correctly, we will run an example program by following these steps:

- Step 1: Create two VHDL files: "invert.vhd" and "test_invert.vhd". Copy the code below into these files or download the files from https://github.com/nmentens/SAC.

1

invert.vhd	test_invert.vhd

```
library ieee;
use ieee.std_logic_1164.all;

entity invert is
        port(    a: in std_logic;
                 b: out std_logic);
end invert;

architecture arch of invert is
begin
        b <= not a;
end arch;
```

```
library ieee;
use ieee.std_logic_1164.all;

entity test_invert is
end test_invert;

architecture arch of test_invert is
        signal a, b: std_logic;
        component invert is
                port(    a: in std_logic;
                         b: out std_logic);
        end component;
begin
        inst_invert: invert
                port map(a => a, b => b);

        p: process
        begin
                a <= '0';
                wait for 10 ns;
                a <= '1';
                wait for 10 ns;
                wait;
        end process;
end arch;
```

"invert.vhd" describes an inverter with input *a* and output *b*. "test_invert.vhd" describes a testbench that applies test stimuli to the input: *a* is initialized to 0 and toggled to 1 after 10 ns. The simulation ends 10 ns later. We will elaborate more on the structure of these files during the tutorial.

- Step 2: Open a command/terminal window. In Windows, this can be done by opening the "cmd" program and navigating to the directory in which the VHDL files are stored. Another option to open a command prompt in Windows, is to go to your folder in Windows Explorer, highlight the complete folder path in the top pane and then overwrite the path by typing "cmd". This will open a command prompt in that folder. To analyze both VHDL files, use ghdl –a. To elaborate on the testbench, use ghdl –e. To run the testbench and specify the waveform output file, use ghdl –r. This can be done by executing the following four commands in the command/terminal window:

ghdl -a invert.vhd
ghdl -a test_invert.vhd
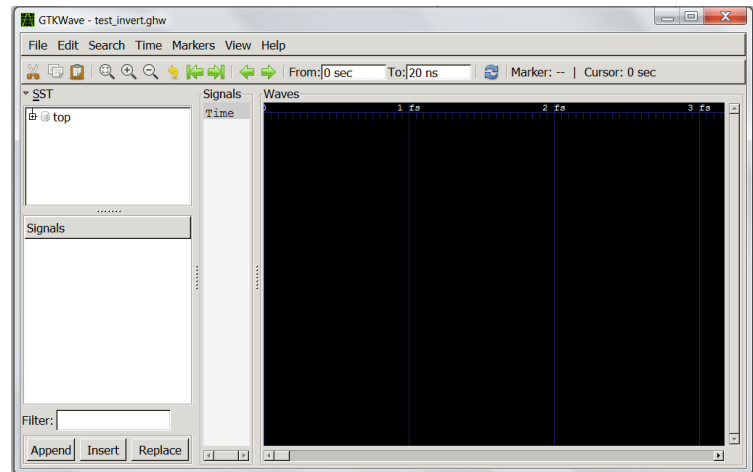ghdl -e test_invert
ghdl -r test_invert --wave=test_invert.ghw

- Step 3: Open a new command/terminal window in the directory where the VHDL files are stored, while keeping the GHDL command/terminal window of Step 2 open. Use the new command/terminal window to display the waveforms using gtkwave:
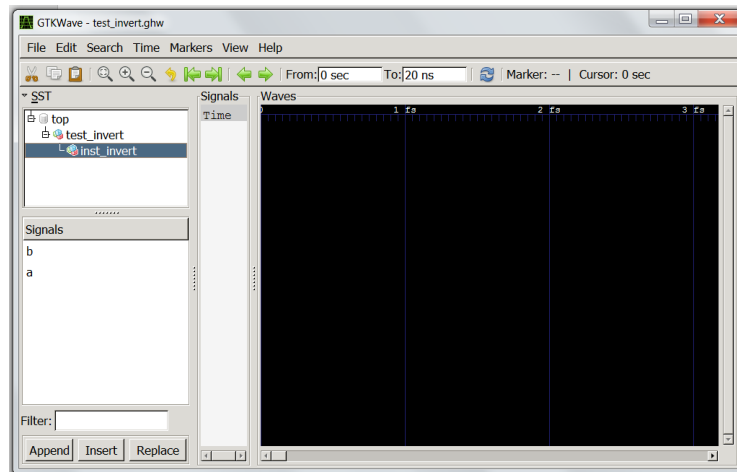
gtkwave test_invert.ghw

The GTKWave graphical user interface should open now, presenting this view:

- Step 4: Expand the top module in the SST (Signal Search Tree) Window by clicking the + sign. Expand the test_invert module by clicking the + sign. Select the inst_invert module such that you get this view:

- Step 5: Select the signal *b* and drag it to the Signals Window. Do the same for signal *a*. Zoom to the full simulation view by using the Zoom Fit button. Finally, this view should be displayed:
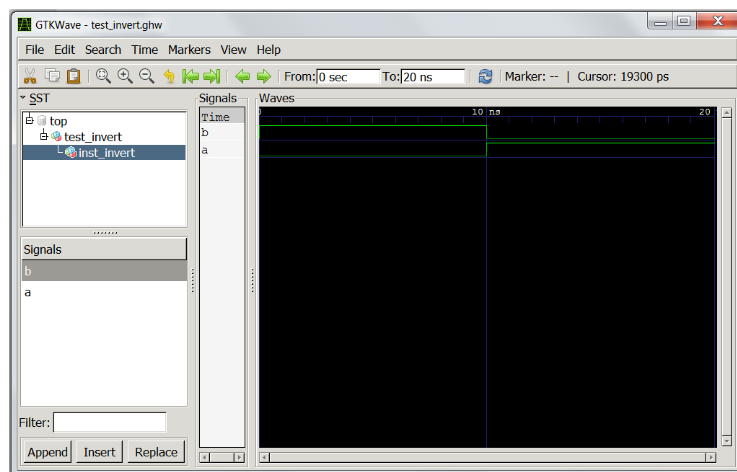
Now the waveforms of *b* and *a* are shown for the total duration of the simulation, namely 20 ns. We see that *a* behaves as driven by the testbench, and *b* is the inverse of *a*.

- Step 6: This step shows how the simulation can easily be rerun after making changes to the VHDL code. Change the code of "invert.vhd" to remove the inversion:

  invert.vhd

  ```
  library ieee;
  use ieee.std_logic_1164.all;

  entity invert is
          port(    a: in std_logic;
                   b: out std_logic);
  end invert;

  architecture arch of invert is
  begin
          b <= a;
  end arch;
  ```

  Save the file and rerun the simulation commands in the GHDL window:

  ```
  ghdl -a invert.vhd
  ghdl -a test_invert.vhd
  ghdl -e test_invert
  ghdl -r test_invert --wave=test_invert.ghw
  ```

  Press the reload button 🔄 in the GTKwave window to update the waveform. Observe the change in the output signal *b*. It is no longer the inverse of *a*, but it is equal to *a*.

  This is very useful when debugging or updating an implementation, since you do not need to open GTKwave and add all signals again.

- Step 7: The simulation can be stopped before the end after a given amount of time. Redo the simulation as follows to make it stop after 10 ns (while the simulation would normally end after 20 ns):

  ```
  ghdl -r test_invert --wave=test_invert.ghw --stop-time=10ns
  ```

  Press the reload button in the GTKwave window and observe that the simulation finishes after 10 ns instead of 20 ns. This feature can be used to debug intermediate signals in a long simulation.