Introduction to VHDL

What is VHDL?

VHSIC (Very High Speed Integrated Circuit) Hardware Description Language is a programming language that was made so people can create circuits. This is compared to using block diagrams in Quartus. VHDL is used to create circuits on Field Programmable Gate Arrays (FPGA's), which is what our boards are, and other integrated circuits.

VHDL programming is not like programming in High Level Language. You will use very few loops, if-else statements, etc. The main reason is that VHDL is used to describe circuits, while a HLL is used to design something that will use a circuit. You are not out to make a circuit do anything, you are simply making a ciruit.

One difference about this way of programming is that the order you generally do something in will not matter. If I did the following in a HLL I would not get the right answer:

```
void main(){
  int x = 7;
  int y = 5;
  int z = 0;
  \\I want z to equal x-2+y-3
  z = x+y;
  x = x-2;
  y = y-3;
  \\But z ends up being 12 not 7
  return
}
```

I can however do the this in VHDL because it will synthesize the inputs before it does anything with the outputs. The only restriction to this is if your circuit requires synchronization.

Why VHDL?

There are a few distinct advantages and a few disadvantages to using VHDL. The advantages are that you won't have to deal with wires, your design will compile much faster, and changes are a lot easier to make than with circuits. A few of the disadvantages are that you won't be able to completely visual your circuit and it is an unknown language with somewhat difficult syntax. This is why we are giving out two documents so that you can choose how to develop your project.

Example VHDL code

The below code is an example of how to create a 1-bit AND gate in VHDL. Look through the code and then read about the breakdown of it afterwards.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY andGate IS
  PORT(
    A,B    :IN STD_LOGIC;
    f      :OUT STD_LOGIC
  );
END andGate;

ARCHITECTURE behavior OF andGate IS
BEGIN
  f <= A AND B;
END behavior;
```

You may be thinking that this is an over complicated way to design something as simple as an AND gate. You would be right. The power of VHDL comes when you begin designing much more complicated circuits. If you choose to do your ALU or Register File in VHDL your program will end up being smaller, faster, quicker to compile, and far more readable than a block diagram.

Let us start the breakdown from the top:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
```

These lines of code act like an include statement in C or an import statement in Java. Your are telling the synthesizer that you want to include the definitions from the ieee library and more specifically you want all of the definitions in the std_logic_1164 section.

Now we define the circuit

```
ENTITY andGate IS
  PORT(
    A,B     :IN STD_LOGIC;
    f       :OUT STD_LOGIC
  );
END andGate
```

This may seem confusing at first, but it really is quite simple. All you do here is define an Entity, which we name andGate, and then specify its inputs and outputs. An entity is the circuit you are designing. Every VHDL file will have an entity declaration in it and the name of the file **has** to match the name of the entity.

Now we elaborate on what the circuit will do:

```
ARCHITECTURE behavior OF andGate IS
BEGIN
  f <= A AND B;
END behavior;
```

So far all we have done is include a library and then decide what the name of the circuit is and what its inputs and outputs are. Now we need to implement the code that will tell our circuit how to work. The ARCHITECTURE keyword here is telling the synthesizer that it will now tell it how to handle the inputs and outputs. Behavior is a name we gave this implementation. andGate is then the name of our entity. After that we use the BEGIN command and write code in between until the END command.

Lets look more closely at:

```
f <= A AND B;
```

This is a basic operation command. It means that whatever port/signal on the left will take on the value of whatever operation is conducted on the right. In this occurrence it means that f, an output, will be either a 0 or a 1 depending on the operation of A AND B, which are both inputs.

Testing the AND gate

Testing of the code we just created will follow mostly how it is done with a block diagram. Follow the steps below to create a project and add this VHDL file to it and then compile it. The file is uploaded on Piazza or you can copy the code from this document

1. Open up Quartus II
2. Create a new project name AndGateTest
3. Use the same specifications as you have for any labs
4. After the project is created add the VHDL file to the code
   (a) Click on Project tab at top
   (b) Select Add/Remove files in Project
   (c) Brows to the VHDL file and add it
5. Now open the file in Quartus
6. Now set the VHDL file as the top level entity by hitting CTRL-SHIFT-J with the file focused in Quartus
7. Finally hit CTRL-L to compile the design

Once the design has finished compiling, you can now create the .do file that will simulate the circuit, add the .do file to your project, and finally simulate it and find out if it works. You can copy from the code listed below:

```
vsim andGate
view wave

add wave A
add wave B
add wave f

force A 0 0, 1 10 -repeat 20
force B 0 0, 1 5 -repeat 10
run 20
```

Once you have the .do file created or download, set it as the simulation file by doing these steps:

1. Click on Assignment tab at top
2. Select Settings
3. Look for Simulation under EDA Tool Settings and click on it
4. Look under where it says NativeLink setting and click the button for Script to compile test bench
5. Search for your .do file and select it
6. Hit Apply and then Ok

Finally all you have to do is click on Tools–Run Simulation Tool–Gate Level Simulation–Slow-Model(If it asks)–Run. View the waveform and make sure the AND gate worked.

Questions

You should hopefully be able to answer the questions below now. If not, review above, or follow one the linked resources below

1. What is an Entity?
2. What is a port?
3. How do I define an input port called SUM of type std_logic?
4. What does the following do?

   ```
   f <= A XOR B;
   ```

References

Review the following websites if you want to learn more about VHDL before the next lab or if you need a reference guide:

- `http://esd.cs.ucr.edu/labs/tutorial/`
- `http://www.seas.upenn.edu/~ese171/vhdl/vhdl_primer.html#_Toc526061341`
- `http://www.csee.umbc.edu/portal/help/VHDL/VHDL-Handbook.pdf`