New York City
College of technology

Component Subsystem Design II

Lab 1

VHDL Components and Port Maps

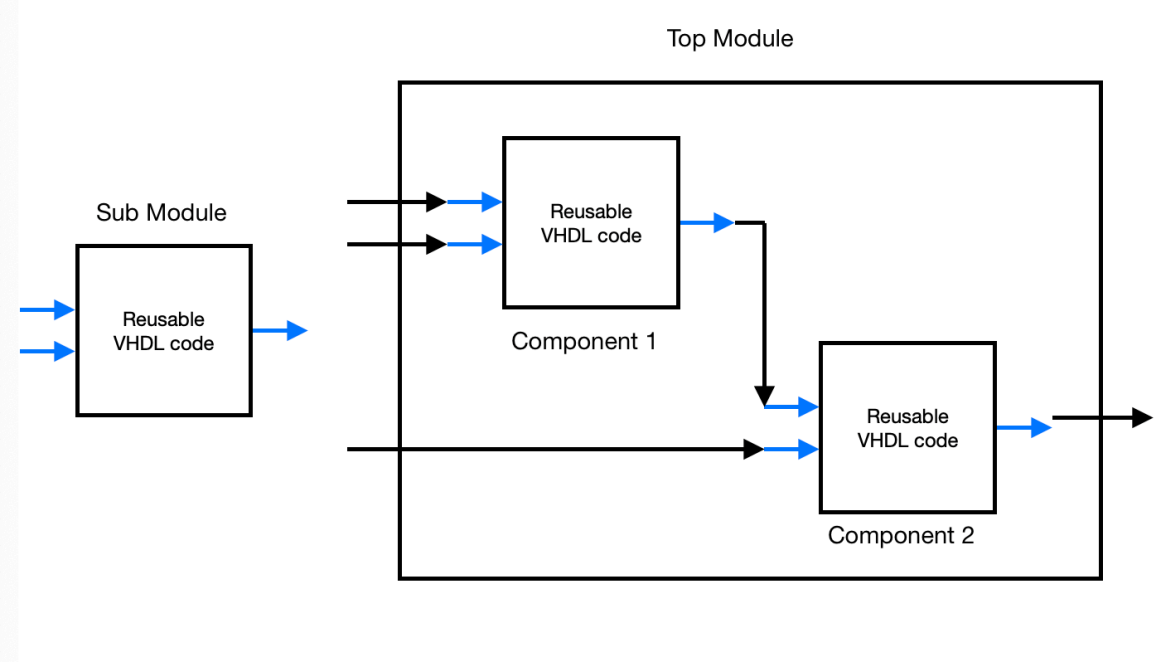
CET 4805 SECTION:

INSTRUCTOR: Prof Y. Wang

VHDL Components and Port Maps

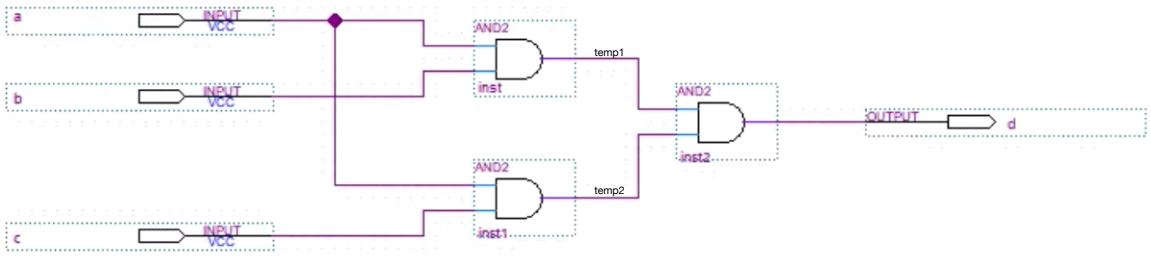
Using Components written in VHDL are a very important part of designing embedded systems. A component is a reusable VHDL module (block of code) that can be declared in other digital logic circuits using component declaration of the VHDL code. Instead of coding each component of the design in a single VHDL code we can divide the code into smaller modules as component and combine them using the port map technique. Port Map is the process of mapping inputs/ outputs of components in the main VHDL file. Think of this process as using functions in high level programming languages such as C++, Where the component is the function and port mapping is calling the function to the main program.

Port Map Block Diagram



Using the port map diagram as a template we will design the following circuit with and gates in place of the Sub Module.

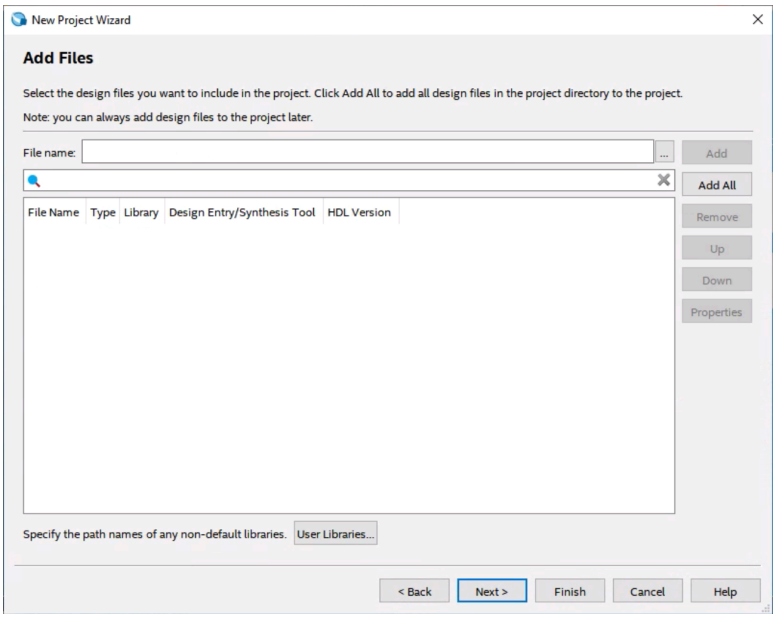
The following system has three inputs (a, b, c) and one output (d). It has two internal lines that are called temp1 and temp2 the system will reuse the 2 input and gate three times.

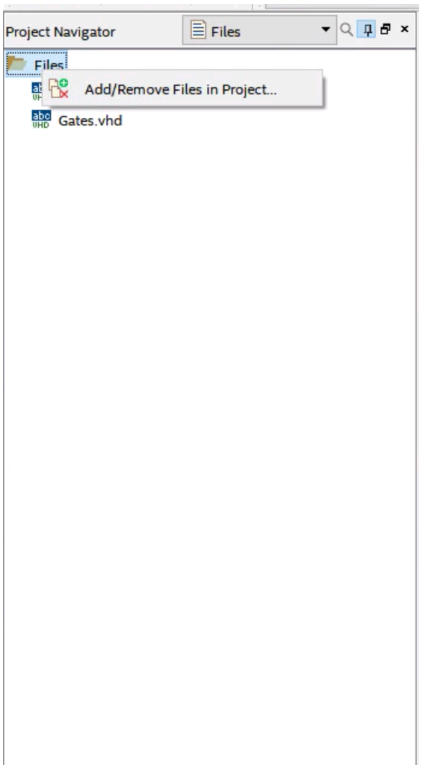


A. Open the Quartus Prime software and create a new VHDL file without starting a new project, Save this VHDL file as **and_gate.vhd**. The VHDL file can be saved in a separate section other than your project folder and added to the main file **Gate.vhd** later on. Below we can see the system above translated int VHDL code.

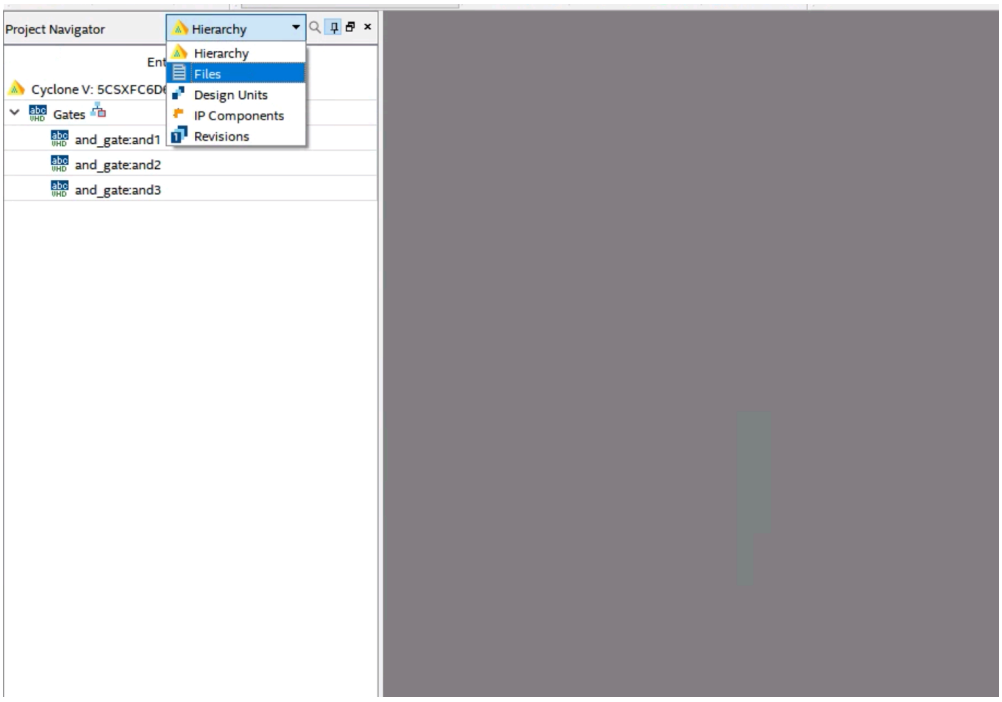
```
1
2   Library ieee;
3   Use IEEE.STD_LOGIC_1164.ALL ;
4   Use IEEE.STD_LOGIC_ARITH.ALL ;
5   Use IEEE.STD_LOGIC_UNSIGNED.ALL ;
6
7   entity and_gate is
8      port (   x , y : in std_logic;
9                  z : out std_logic);
10  end and_gate ;
11
12  architecture behavior of and_gate is
13
14  begin
15     process (x,y)
16        begin
17
18        if (x = '1') and (y = '1') then
19
20        z <= '1' ;
21
22        else
23
24        z <= '0' ;
25
26        end if;
27
28     end process ;
29
30  end behavior ;
31
```

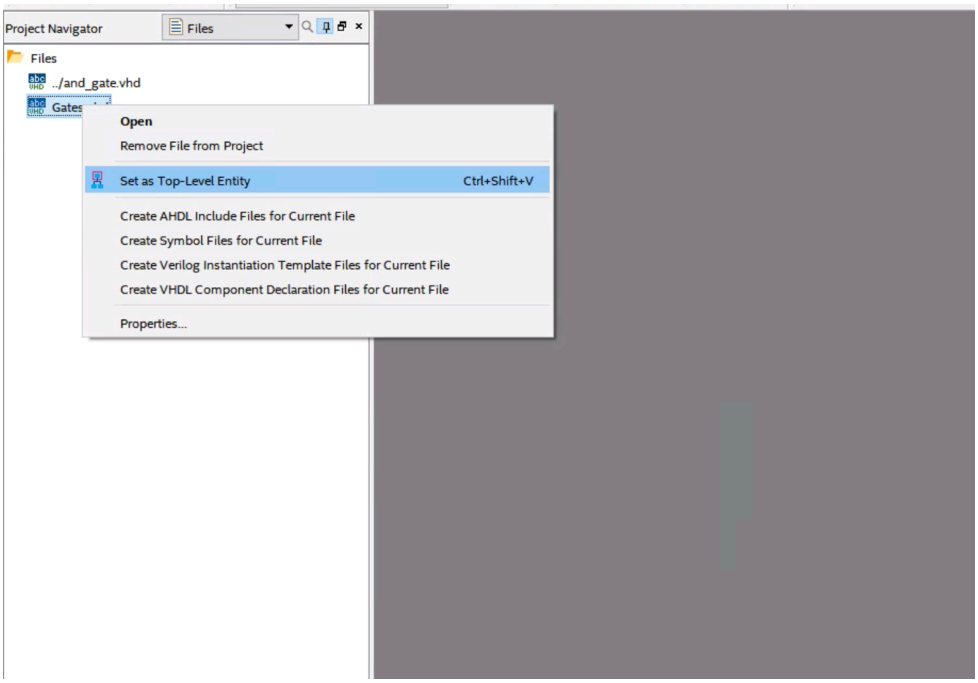
B. After creating the **and_gate.vhd** file we need to start a new project. Go to new project wizard and start a new project, name this project Gates. when you reach this screen you will click on […] and add the **and_gate.vhd** file to this new project. Files may also be added under the project navigator by right clicking and adding the file. *By default the VHDL will look in the same directory for the file and includes it directly.*





C. After creating a new program we will need to create a new vhd file. Name the file **Gates.vhd**, this will be our top level entity (top module) where we will be able to call and reuse the **and_gate.vhd** component (sub module). After creating the **Gates.vhd** file set it as the top level entity by setting the project navigator to files and right clicking on the Gates.vhd file. This must be always done on your main file where your port mapping is done to ensure the compiler knows which file to start the compile. it will automatically check the rest of the files because they are referenced in the main file.

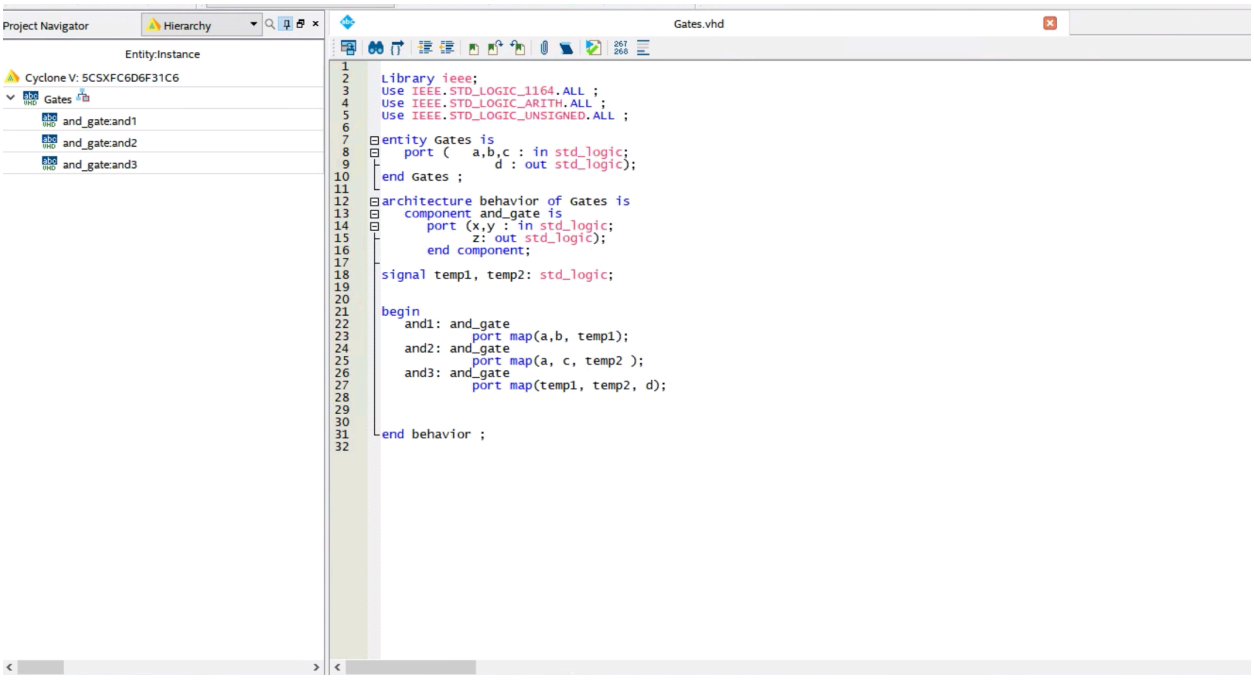




D. In order to use a component in your main file you will need to add it to your main code, this is done by using the code below, if you have multiple components they would need to have their own component in the main file by using the code below as a reference. All the input and output variables used in the original component file must be included when calling your component to the main file.

```
component and_gate is
    port (x,y : in std_logic;
          z: out std_logic);
    end component;
```

E. In the main file we will need to declare gates and set the port maps. We can see the *inputs a, b, c* and *output d* these will be mapped to the hardware inputs and outputs and be used to send signals to the component(s). Notice when port mapping the same amount of inputs used in your component file must be sent into the port map and the same amount of outputs are needed to receive data from the component. The component is port mapped 3 times because we used an and gate 3 times in our design, the same three components can also be seen in the project navigator under the top entity Gates.



F. Once Gates.vhd and and_gate.vhd are created, compile and assign the switches to inputs a, b, and c and LED to output d using the pin planner under assignments using the pins below.

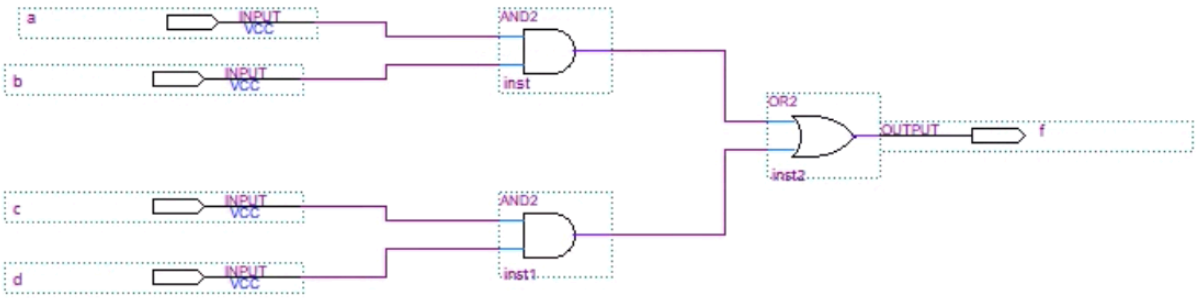
Table 3-6 Pin Assignment of Slide Switches

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
SW[0]	PIN_AB30	Slide Switch[0]	Depend on JP3
SW[1]	PIN_Y27	Slide Switch[1]	Depend on JP3
SW[2]	PIN_AB28	Slide Switch[2]	Depend on JP3
SW[3]	PIN_AC30	Slide Switch[3]	Depend on JP3
SW[4]	PIN_W25	Slide Switch[4]	Depend on JP3
SW[5]	PIN_V25	Slide Switch[5]	Depend on JP3
SW[6]	PIN_AC28	Slide Switch[6]	Depend on JP3
SW[7]	PIN_AD30	Slide Switch[7]	Depend on JP3
SW[8]	PIN_AC29	Slide Switch[8]	Depend on JP3
SW[9]	PIN_AA30	Slide Switch[9]	Depend on JP3

Table 3-8 Pin Assignment of LEDs

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
LEDR[0]	PIN_AA24	LED [0]	3.3V
LEDR[1]	PIN_AB23	LED [1]	3.3V
LEDR[2]	PIN_AC23	LED [2]	3.3V
LEDR[3]	PIN_AD24	LED [3]	3.3V
LEDR[4]	PIN_AG25	LED [4]	3.3V
LEDR[5]	PIN_AF25	LED [5]	3.3V
LEDR[6]	PIN_AE24	LED [6]	3.3V
LEDR[7]	PIN_AF24	LED [7]	3.3V
LEDR[8]	PIN_AB22	LED [8]	3.3V
LEDR[9]	PIN_AC22	LED [9]	3.3V

Your part



1. Start the Quartus prime software go to **File > New project wizard**. Create a new project in a new folder on your flash drive and assign the name ***BooleanEq***, assign the Cyclone V for the device family and select the **5CSXFC6D6F31C6** under available devices, which is the FPGA chip used on the DE10-Standard

2. Create a new VHDL design file **(File > New)** by highlighting the VHDL file in the pop up. And click OK. **Add in the VHDL code for and_gate.vhd** and save the file as part of your project. Alternatively you can add the one created earlier to this new project.

3. Create a new VHDL design file **(File > New)** by highlighting the VHDL file in the pop up. And click OK. **Create the VHDL code for of_gate.vhd** and save the file as part of your project. You will use **and_gate.vhd** as a template to create an or_gate component.

4. Create a new VHDL design file **(File > New)** by highlighting the VHDL file in the pop up. And click OK. **Create the VHDL code for BooleanEq.vhd** and save the file as part of your project. Be sure to set this file as your top level entity. Use **Gates.vhd** as a template to create the **Boolean.vhd** entity

5. Combine the files together (entity and components) and refer to the schematic to create your design using VHDL.
6. Compile and simulate Boolean.vhd. you should see the waveform output (20μs period).
7. Use the pin planner to configure your pins
8. Download the design onto the FPGA board and test your circuit works correctly
9. Submit your report to Blackboard.