LAB #3

Display the output of combinational circuit using library building technique of VHDL programming

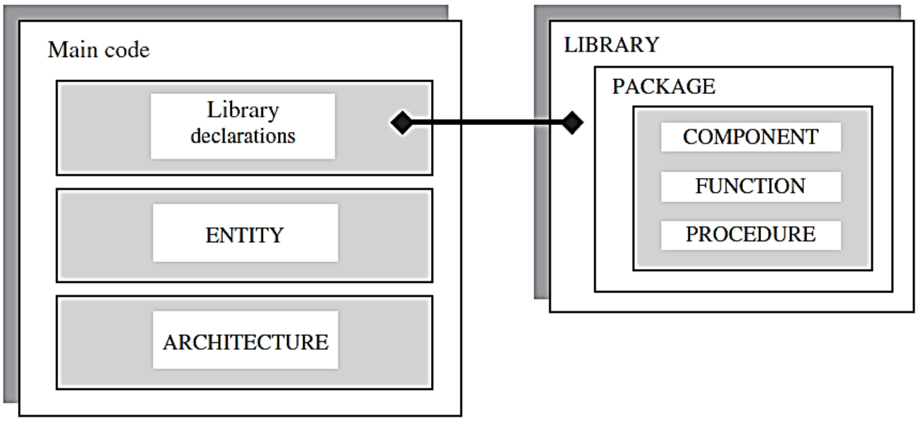
Objective

- To learn to utilize the library functions in VHDL
- To learn how to create own library in VHDL

Pre-Lab

Creating your own library components

Just like other programing languages, VHDL also provide a functionality to create your own library called package. The package contains components, function, procedures, constants and types. In this lab, we will learn how to create our own library.



We will learn this concept by walking you through an example (as pre-lab) and later on asking you; the student, to create another library to solve the in-lab task.

Pre-Lab Task

Task 1: Creating your own library: Step-by-Step Practice

Consider the combinational circuit given in the figure 1.

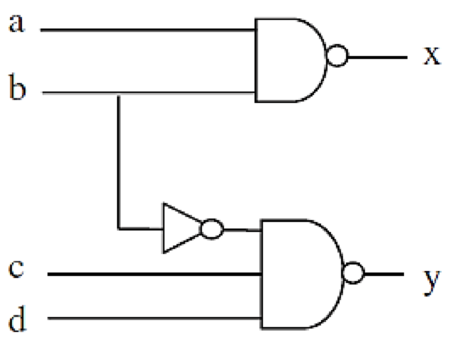


Figure 1 : Combinational circuit

The circuit in figure 1 can be implemented as

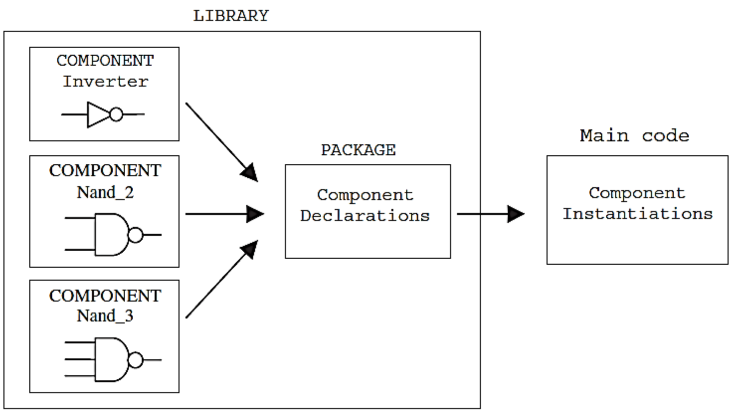


Figure 2: contents of user-defined library

1) Entity

The firsts part in creating a library is to define the entities that needs to be grouped together as a package. The entities required are

i. An inverter

```
-- Code for myinverter.vhd file

library IEEE;
use ieee.std_logic_1164.all;
entity myinverter is
port(
        inv_in: in std_logic;
        inv_out: out std_logic
);
end myinverter;
architecture bhv of myinverter is
begin
    inv_out <= not inv_in;
end bhv;
```

ii. A 2-input NAND

```
---------------------Code for nand_2.vhd

library IEEE;
use ieee.std_logic_1164.all;
entity nand_2 is
port(
        nand2_in1: in std_logic;
        nand2_in2: in std_logic;
        nand2_out: out std_logic
);
end nand_2;
architecture nand_2 of nand_2 is
begin
    nand2_out <= NOT (nand2_in1 AND nand2_in2);
end nand_2;
```

iii. A 3-input NAND

```
---------------------Code for nand_3.vhd
library IEEE;
use ieee.std_logic_1164.all;
entity nand_3 is
port(
        nand3_in1: in std_logic;
        nand3_in2: in std_logic;
        nand3_in3: in std_logic;
        nand3_out: out std_logic
);
end nand_3;
architecture nand_3 of nand_3 is
begin
    nand3_out <= NOT (nand3_in1 AND nand3_in2 AND nand3_in3);
end nand_3;
```

2) Package

Once the entities are created now, we need to create the package (library in our case)

```
---------------------Code for my_components.vhd
library ieee;
use ieee.std_logic_1164.all;

package my_components is
component myinverter is
port( inv_in: in std_logic; inv_out: out std_logic);
end component;

component nand_2 is
port     (   nand2_in1: in std_logic;
             nand2_in2: in std_logic;
             nand2_out: out std_logic
         );
end component;

component nand_3 is
port(
         nand3_in1: in std_logic;
         nand3_in2: in std_logic;
         nand3_in3: in std_logic;
         nand3_out: out std_logic
);
end component;
end my_components;
```

3) The top-entity (wrapper file)

Now we need to define another vhdl file that will be called a wrapper file. This file will integrate components from our package and create a circuit.
Remember to set this file as top-level entity.

```vhdl
-- -- Code for circuit1, wrapper file

library ieee;
use ieee.std_logic_1164.all;
use work.my_components.all; -- declaring our Library

entity circuit1 is
port( a,b,c,d : in std_logic;
        x,y: out std_logic
    );
end circuit1;

architecture struct of circuit1 is
signal w: std_logic;
begin
    --calling the components
    U1: myinverter PORT MAP
                (inv_in  => b,
                 inv_out => w);
    U2: nand_2 PORT MAP
                (nand2_in1 => a,
                 nand2_in2 => b,
                 nand2_out => x);
    U3: nand_3 PORT MAP(w,c,d,y);
end struct;
```

iv. Mapping Options

A ***port map*** is used to define the interconnection between instances.
PORT MAP is simply a list relating the ports of the actual circuit to the ports of the predesigned circuit (component being instantiated). Such a mapping can be positional or nominal, as illustrated in the example below, which employs the nand3 circuit again.

```vhdl
-------- Component instantiation: -----------------------
nand3_1: nand3 PORT MAP (x1, x2, x3, y); --positional mapping
nand3_2: nand3 PORT MAP (a1=>x1, a2=>x2, a3=>x3, b=>y); --nominal mapping
```

Use the following pin assignments to test the code on FPGA.

a	b	c	d	x	y
PIN_AD27	PIN_AC27	PIN_AC28	PIN_AB28	PIN_G19	PIN_F19

In-Lab Tasks

Lab Task 1: Designing a 4-bit ALU (Arithmetic Logic Unit) Using VHDL Packages

The goal of this task is to implement a **4-bit ALU (Arithmetic Logic Unit)** using VHDL packages. The ALU should perform a variety of arithmetic and logic operations based on control inputs(opcode). You will create a reusable package for arithmetic and logic functions and use it to build the ALU.

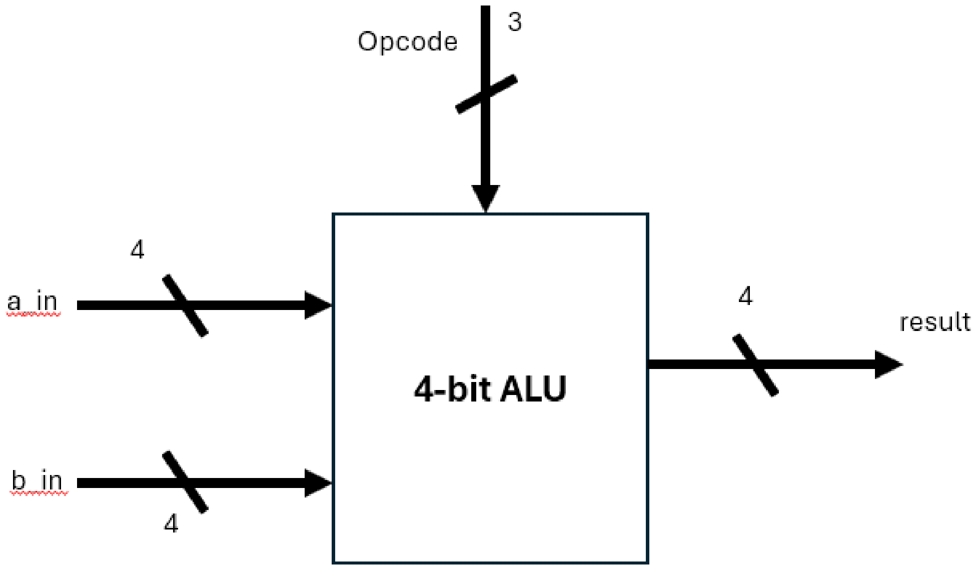


Figure 3: A simple ALU implementation

1) Create a Package for **Arithmetic and Logic Operations**: You will define a package that includes:
    - Basic arithmetic operations (ADD, SUBTRACT).
    - Basic logical operations (AND, OR, XOR, NOT).
2) Design a 4-bit ALU: Using the package, you will implement a 4-bit ALU. The ALU should take two 4-bit inputs (A and B), perform operations based on a 3-bit control signal (opcode), and output the 4-bit result (Y).

3) Operations to Implement and include in package:
    a) ADD: Y = A + B (full/half adder)
    b) SUBTRACT: Y = A – B (full/half subtractor)
    c) AND: Y = A AND B
    d) OR: Y = A OR B
    e) XOR: Y = A XOR B
    f) NOT: Y = NOT A

Note: In standard ieee.std_logic_1164 package there is an implementation of 1-bit universal gates. For this lab create your own 4-bit version of universal gates and use it to implement the ALU.

Pin assignments can be chosen from the Altera user manual file.

Rubric for Lab Assessment

The student performance for the assigned task during the lab session was:			
Excellent	The student completed assigned tasks without any help from the instructor and showed the results appropriately.	4	
Good	The student completed assigned tasks with minimal help from the instructor and showed the results appropriately.	3	
Average	The student could not complete all assigned tasks and showed partial results.	2	
Worst	The student did not complete assigned tasks.	1	

Instructor Signature: ______________________ **Date:________________**