Lab 12: Using Packages and Libraries

Goal of this lab is to create an arithmetic unit module that makes use of VHDL packages and libraries. You will design an arithmetic unit that performs basic operations like addition, subtraction, multiplication, and division. The arithmetic operations will be encapsulated in a package that you will create, and then you'll implement a module that uses this package.

A testbench is provided to check the functionality of the package.

The ArithmeticUnit module shall has the following ports

Port	Direction	Type	Width
op1	IN	signed	OP_WIDTH
op2	IN	signed	OP_WIDTH
opcode	IN	op_type	-
result	OUT	signed	OP_WIDTH

`OP_WIDTH` is an integer constant and `op_type` is a custom enumerated type, which must be defined in the package.

Exercise

1. Create the package

Go to `~/labs/lab12/` and open `src/alu_pkg.vhd` with a text editor.

```
kate src/alu_pkg.vhd &
```

Define a package `ArithmeticPkg` that includes the following:

- A constant `OP_WIDTH` representing the width of the operands. Set its value to `8`.
- An enumeration type `op_type` for the available operations (`ADD, SUB, MUL, DIV`).
- A function `compute` that performs the selected operation on two operands. The function should get as argument the two signed operands and an operation code, and return the result in form of a signed vector of size `OP_WIDTH`.

2. Create the Top module

Open `src/alu.vhd` with a text editor.

```
kate src/alu.vhd &
```

- **Design the `ArithmeticUnit` module** using the package you created. The package should be compiled in the `alu_lib` library.
- The module should use the function defined in the package to perform the selected operation on the input operands.
- Implement the necessary ports for the module as described above

3. Compile the package into a library

The `ArithmeticPkg` should be compiled in the `alu_lib` package. In Vivado simulation this can be done using the option `-work <library_name>` of the `xvhdl` command.

Open `run_sim.sh` to compile `alu_pkg.vhd` in the `alu_lib` library.

4: Simulate the Design

Run the simulation to check your design.

```
./run_sim.sh
```

(Optional) 5. Use VHDL records

Create a new VHDL record type `t_aluinput`, that groups all the input signals into the arithmetic unit module. Modify the `compute` function, the `alu.vhd` and the testbench `sim/tb_alu.vhd` accordingly to use this new record.