4-bit Arithmetic Unit

End Project of Digital Design, authored by:

- Mohamed Hassan 641717
- Mhammad Nour Altahan 643572

1.Introduction

A simple Hardware Unit written in VHDL that represents a 4-bit device that can do 4bit arithmetic operations such as:

1- **Addition**
2- **Subtraction**
3- **Division**
4- **Multiplication**

The IEEE standard already offers a library to do arithmetic operations. But our idea was to create a primitive form of a processor's **ALU** (without the Logic, so basically just AU) completely and solely on the **Register Transfer Level**
The idea is to implement these operations completely from scratch using only the clock and combinatoric elements (logic gates).

Note: The device can also do other operations, which are internally used to implement the 4 main operations listed above from scratch. These operations can be very easily added to the main FPGA driver code (au_main.vhd), but due to our vision of pin Assignment on the used FPGA board, and to keep everything within the time constraint we have, we decided to not include these operations in the final FPGA driver code (main code)

More information here: 3. System Architecture

2. Distribution of Tasks

The project was distributed as the following:

- Mohamed Hassan:
  - Addition
  - Division
- Mhammad Nour Altahan:
  - Subtraction
  - Multiplication

Each Team member was also responsible for designing and creating the necessary core Entities required to do said operations. <link to the unit explaining dependencies>

3. System Architecture

The entire system is built using a component-based approach. Meaning that very primitive components are created, then hierarchically assembled to form a bigger, more complex component that solves a more complex task. Eventually the 4 main arithmetic components are created and put together in an Entity that drives them all together and create the FPGA driver code, or basically the "application"

Here is a list of the components, and what they are made of:

- ***Core****:* the most basic component
  - Half_adder
  - Full_adder
- ***Primitive****:* next level of abstraction after *Core*, it is the main building block to all other components, higher in the abstraction level

Bin_4bit_adder	$x + b \Rightarrow c$
Bin_4bit_subtractor	$x - b \Rightarrow c$
Bin_4bit_negator	$x \Rightarrow (-x)$
Bin_4bit_abs	$x \Rightarrow \|x\|$
Bin_4bit_comparator	x ? y (bigger, smaller, equal)

- ***Operation***: higher level of abstraction the *Primitive*:

bin_4bit_signed_divider	$x / y \Rightarrow c$
bin_4bit_signed_multi	$x * y \Rightarrow c$

- ***Application***:
  - Au_main

This system architecture is very flexible, since we can very easily extend the application by adding more *Operation, Primitive* or *Core* components