Digital System Design
Exercise 1

Yousef Baroud

1 Task 0

Execute the following steps:

- Copy the exercise material to your computer:

  ```
  scp -r /usr/local.nfs/pas/teaching/dsd/dsd_ws_2018_2019/ex1 .
  ```

- Start Vivado:

  ```
  source /import/pas.local/sw/Xilinx/Vivado/2017.4/settings64.sh
  vivado&
  ```

- In Vivado:
  File –> New Project
  Project name: ex1
  Project location: *point to ex1 (where you copied the exercise material)*
  Do not tick the **Create project subdiretory check box!**
  Next…**Next** till the project is created.

- Add design and simulation sources to the project
  ⊙ Right click on **Sources** window
  Add or create design sources
  Add files `mux.vhd, comp_mux.vhd`
  Do not tick **Copy sources into project!**

2 Task 1

Given this entity (given in mux.vhd):

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.all;

entity mux is
  port (
    a        : in  std_logic_vector(3 downto 0);
    b        : in  std_logic_vector(3 downto 0);
    c        : in  std_logic_vector(3 downto 0);
    d        : in  std_logic_vector(3 downto 0);
    sel      : in  std_logic_vector(1 downto 0);
    selected : out std_logic_vector(3 downto 0));
end entity mux;
```

, implement a 4-to-1 MUX that is described by the following table:

sel	selected
00	a
01	b
10	c
11	d

using the `when…else` construct, `with…select` construct, `if…else`, and `case…when` construct.

Elaborate and synthesize the design and study the schematic/netlist.

3 Task 2

Given the following entity:

```
entity prior_enc is
  port (
    r      : in  std_logic_vector(3 downto 0);
    code   : out std_logic_vector(1 downto 0);
    active : out std_logic);
end entity prior_enc;
```

, implement the following truth table using `with…select` constructs:

4 Task 3

Given the following entity:

r	code	active
1—	11	1
01–	10	1
001-	01	1
0001	00	1
0000	00	0

```
entity comp_mux is
  port (
    in0    : in  std_logic_vector(3 downto 0);
    in1    : in  std_logic_vector(3 downto 0);
    op     : in  std_logic_vector(1 downto 0);
    output : out std_logic_vector(3 downto 0));
end entity comp_mux;
```

, implement the functionality described as follows. `op` defines the operation of the architecture. When `op` is all zeros, `output` is assigned the sum of the `in0` and `in1`. When `op` is "01", `output` is assigned the absolute value of the difference between `in0` and `in1`. When `op` is "10", `output` is assigned the result of bit-wise ORing of `in0` and `in1`. Otherwise, `output` is assigned a shifted version of `in0`. The sign of `in1` determines the shift direction; left when `in1` is negative and right when `in1` is positive. The magnitude of the shift is determined by the absolute value of `in1`.

5 Task 4

VHDL describe a (synthesizable) 1-bit full adder. Design a N bit ripple carry adder (by instantiating N 1-bit full adders). Let your design be parameterizable by deploying generics (`for...generate` construct).