# EXERCISE 2
# DESIGN, SIMULATION AND SYNTHESIS OF SEQUENTIAL LOGIC AND IMPLEMENTING ON THE ZEDBOARD

## Universal Counter

- ✓ There are many styles to design sequential logic. Even when all of them may behave in the same manner during simulations, each of these styles affects synthesis of the actual hardware in different ways. The task here is to design a loadable, resettable, bidirectional counter in two different ways. Both of them should be simulated to ensure the correctness of the description and finally synthesized.
- ✓ The requirements are (see also the table below):
- ✓ Counting is allowed only when the input **enable** is '1', otherwise it must be paused. Resetting occurs when the input reset is '1'.
- ✓ Counting from 0 to 15 (on the output **count**) when input **down/up** is '0', and from 15 down to 0 when the input is '1'.
- ✓ On overflow (underflow) the counter must generate '1' on output **over** and continue with counting from 0 upwards (from 15 downwards).
- ✓ '1' on the input load loads the counter parallelly (from input **data**), the input **enable** must be '1' at the same time.

| reset | enable | load | down/up | data | $count^t$ | $count^{t+1}$ | $over^{t+1}$ |
|---|---|---|---|---|---|---|---|
| **1** | **-** | **-** | **-** | **-** | **-** | **0** | **0** |
| **0** | **0** | **-** | **-** | **-** | **Q** | **Q** | **0** |
| **0** | **1** | 0 | **0** | **-** | **15** | **0** | **1** |
| **0** | **1** | 0 | **0** | **-** | **Q** | **Q+1** | **0** |
| **0** | **1** | 0 | **1** | **-** | **0** | **15** | **1** |
| **0** | **1** | 0 | **1** | **-** | **Q** | **Q-1** | **0** |
| **0** | **1** | **1** | **-** | **D** | **-** | **D** | **0** |

You are required to write the module architecture within **a single process.**

We recommend using variables for count.

Implement a synchronous reset.

**Part1:**

- ✓ Write a test bench which covers all important behaviours of the counter in a sequential manner: reset, enable, counting down, counting up.
- ✓ Simulate the counter to check they operate correctly.

**Part2:**

- ✓ Implement the code in part1 on the zedboard, setting up the switches as inputs, LEDs as outputs and RESET button as reset.
- ✓ Reminder: Zedboard internal clock is 125MHz, you have to use a frequency divider to shorten the counting slot.