Exercise 03: *cocotb* Test

Objective: *Introduction to the cocotb tests.*

Task: *Development of a cocotb test using the Saturation Filter.*

Locate the *cocotb* exercise in `<ROOT>/exercises/E03_sat_cocotb_test`.

Open the `sat_default_test.py` and analyze the file. Then, run the test and visualize the waveforms, running the following commands:

```
# Run the tests
(.venv) [<username>@<servername> E03_sat_cocotb_test]$ make WAVES=1

# visualize the waveforms
(.venv) [<username>@<servername> E03_sat_cocotb_test]$ gtkwave sim_build/sat_filter.fst

# visualize the waveforms in the background by adding an ampersand
(.venv) [<username>@<servername> E03_sat_cocotb_test]$ gtkwave sim_build/sat_filter.fst &
```

If you are running with FST format as the waveform format.

If you are running with VCD, then uncomment:

```
  // Generate waves files.
/*
  initial
  begin
    $dumpfile ("sim_build/sat_filter_waves.vcd");
    $dumpvars (0, sat_filter);
  end
*/
```

and run:

```
# Run the tests
(.venv) [<username>@<servername> E03_sat_cocotb_test]$ make

# visualize the waveforms
(.venv) [<username>@<servername> E03_sat_cocotb_test]$ gtkwave
sim_build/sat_filter_waves.vcd

# visualize the waveforms in the background by adding an ampersand
(.venv) [<username>@<servername> E03_sat_cocotb_test]$ gtkwave
sim_build/sat_filter_waves.vcd &
```

Part 1: Random Test

Now, create a random test. Using the `sat_default_test.py` file as reference, create a *cocotb* test inside the `sat_random_test.py` file. The test should have the following specifications:

- Generate 10 sets of random input data;

- Ensure that the `in_valid` signal is high when setting the `in_data`;
- Assign the data to the input ports of the DUT (the *Saturation Filter*) on the rising edge;
- Wait at least one clock cycle before changing the data again.

Read about the Coroutines and Tasks in the official documentation of *cocotb*. Some examples can also be found there.

Observe the test results and the waveforms.

- Does the waveforms look as expected?
- What happens when changing the waiting time after assigning the data?

Rerun the test for different parameters of the DUT, for example:

```
# Re-run the tests
(.venv) [<username>@<servername> E03_sat_cocotb_test]$ make clean
(.venv) [<username>@<servername> E03_sat_cocotb_test]$ make THRESHOLD=5
```

Part 2: Constraints

Now, update the `sat_random_test.py` test but constraining the input value using the *PyVSC* library. For example, the input data can be constrained in order to generate only positive values below 10.

Read more in the *PyVSC* Data Types documentation and *PyVSC* Constraints documentation.