**Cairo University, Faculty of Engineering, Computer Department**
**VLSI Design and Automation, CMP305, Fall 2024**
**Date: 3-November-2024, Deadline: 12-November-2024**

**Universal Verification Methodology Assignment**

# Overview

In this assignment, you will design a 4-bit synchronous up/down counter with enable and reset functionality as the Device Under Test (DUT). You will also create a UVM testbench to verify the correct operation of the DUT, including components such as a driver, monitor, and scoreboard.

# DUT Specifications

Your DUT is a 4-bit synchronous up/down counter with the following characteristics:

- **Inputs:**
  - **clk** (Clock): Drives the counter.
  - **reset** (Synchronous Reset): Resets the counter to a specific state based on the counting direction.
  - **enable** (Enable Control): When high (1), the counter operates; when low (0), it holds its current value.
  - **up_down** (Up/Down Control): When high (1), the counter counts up; when low (0), it counts down.
- **Output:**
  - **count** (4-bit output): Shows the current count value.

**DUT Behavior:**

- **Synchronous Reset:** When reset is asserted:
  - Sets the counter to 1111 if counting down (up_down is low).
  - Sets the counter to 0000 if counting up (up_down is high).
- **Enable Control:**
  - If enable is 1, the counter operates based on up_down.
  - If enable is 0, the counter holds its current value regardless of up_down.
- **Counting:**
  - With up_down high, the counter increments on each clock cycle when enable is asserted.
  - With up_down low, the counter decrements on each clock cycle when enable is asserted.
- **Roll-over:**
  - If the counter reaches 1111 and up_down is high, it rolls over to 0000.
  - If the counter reaches 0000 and up_down is low, it rolls over to 1111.

# Assignment Tasks

## 1. Implement the DUT

Write the RTL code for the 4-bit synchronous up/down counter as per the specifications above. Ensure that it behaves correctly for each of the inputs, including reset, enable, and up_down controls.

### 2. Develop the UVM Testbench

Create a UVM testbench with the following components:

- **Driver:** Drives randomized control signals (up_down, enable, and reset) to the DUT and applies clock cycles.
- **Monitor:** Observes and logs the DUT's inputs and output, capturing each state change of the count output.
- **Scoreboard:** Implements a reference model that mimics the counter's behavior. The scoreboard should include assertions to check that the DUT's output matches the expected count values. It should log errors when mismatches are detected.

### 3. Verify Counter Behavior

Ensure that your UVM testbench includes but not limited to the following scenarios:

- Up counting and down counting behavior with enable high.
- Reset behavior, where the counter correctly resets to 1111 or 0000 based on up_down.
- Enable functionality, where the counter holds its state when enable is low.
- Roll-over functionality at maximum and minimum count values.

## Submission Requirements

Submit the following files by the deadline:

- RTL code for the DUT.
- UVM testbench files, including driver, monitor, and scoreboard.
- A simulation report that describes the test scenarios covered and summarizes the verification results.

## Grading Criteria

- Correct functionality of the DUT and adherence to specifications: 20%
- Completeness and correctness of the UVM testbench: 20%
- Quality of the simulation report (coverage of test scenarios, clarity, and correctness): 50%
- Code readability and organization: 10%