## LAB 5 – 3300L

### Fall 2024 / Dr. Van Blerkom

For this lab, we will design and implement a digital combination lock. We will use the pushbuttons to enter the code, and we will use a switch to indicate that we want to set a new combination. The combination will be 4 entries long. To implement the functionality, you will use a finite state machine.

1. **States:**
   - Idle – wait for the first entry
   - OneCorrect – wait for entry 2
   - TwoCorrect – wait for entry 3
   - ThreeCorrect – wait for entry 4
   - FourCorrect – open the lock (LED turns Green), wait for clear
   - OneWrong – wait for entry 2
   - TwoWrong – wait for entry 3
   - ThreeWrong – wait for entry 4
   - FourWrong – don't open the lock (LED turns Red), wait for clear
   - ProgramMode – wait for the first entry to set the code
   - SetOne – wait for entry 2
   - SetTwo – wait for entry 3
   - SetThree – wait for entry 4
   - SetFour – Wait for clear (LED turns Blue)
2. **Inputs:**
   - Programming Switch (SW[0]) determines if we are setting the combination or not. (signal prog_sw)
   - Center Pushbutton to clear back to Idle state.
   - Left, Right, Up and Down Pushbuttons are the choices for each combination lock entry.
3. **Outputs:**
   - Tri-color LED will show Green when the lock is open, Yellow when entries are being made, Red when the wrong combination was entered, and Blue when the combination is set to a new value.
   - Use four LEDs to show the current FSM state – this is good for debugging.

4. **Functionality:**
   - The lock's combination consists of four entries of some combination of (Left, Right, Up, Down) – for example, one combination could be Left, Left, Up, Down.
   - The center pushbutton can always be used to go back to the Idle state, to start entering the combination from the beginning.
   - If all four entries are correct, the LED turns green. If any of the entries is wrong, the LED will turn red after the fourth entry. After the four entries are made, the clear button needs to be pressed to go back to the beginning.
   - If the Programming Switch is on, then each entry is saved to become the new combination.

Some starter code is available in the Github repository; consisting of the following modules:

**button_pulse**: this module debounces the push-button, and produces a one-clock cycle pulse when a button is pressed.  All of the inputs from the push-buttons should go through this module first.

**pwm_gen**: this module produces a 25% and 12.5% duty cycle PWM output, and is used by the led_color module.

**led_color_rygb**: this module produces the proper led_r, led_g, and led_b signals for the tri-color LEDs, depending on the input color[1:0]. The color codes are given as parameters in the module.

**store_combo**: this module outputs the valid combination, and will save the new combination if it is being set.