Lab 8: CCP Module

- **Link**
  - Document
  - Video

- **Basic** (50%)
  - **Description**

    Use RB0 as a motor control button to implement the following motor control behavior:
    1. Initial position: 0°.
    2. Each button press advances the motor by 90° following the cyclic sequence:

       0° → +90° → 0° → -90° → 0° → +90° → …
    3. Each subsequent button press continues this sequence in the same pattern.

  - **Standard of grading**
    1. Properly handle the CCP1CON<5:4> bits when configuring the PWM duty cycle.
    2. Both C and Assembly language implementations are acceptable.
    3. The button input must be **edge-triggered with debounce**, meaning the action is triggered only once upon button release, preventing multiple activations while the button is held down.

- **Advance** (30%)
  - **Description**

    Use RB0 as a motor control button to implement the following motor control behavior:
    1. Initial position: -90°.
    2. Each button press advances the motor by 135°.
    3. When the motor reaches +90° or -90°, it must automatically reverse direction and rotate backward.
    4. The motor must rotate gradually, increasing its angle position step by step, i.e., increasing CCPR1L:CCP1CON<5:4> values progressively rather than jumping directly to the final angle.

  - **Standard of grading**
    1. Properly handle the CCP1CON<5:4> bits when configuring the PWM duty cycle.
    2. Both C and Assembly language implementations are acceptable.
    3. The button input must be **edge-triggered with debounce**, meaning the action is triggered only once upon button release, preventing multiple activations while the button is held down.
    4. The motion must be executed incrementally, with smooth transitions between angles. Setting only the final position is not acceptable.

- **Hard** (20%)
  - **Description**

    Use RB0 as a start/stop trigger button, and employ a Timer other than Timer2 to implement the following motor control behavior:

    1. Initial position: -90°. The system starts in the idle (stopped) state.
    2. **Start** trigger:

       When the button is pressed and released, the motor starts moving gradually from -90° to +90°, and then reverses direction back to -90°, repeating periodically.

       The motion should follow the sequence: (-90° → +90° → -90° → +90° → ...)

       The complete round trip (-90° → +90° → -90°) should take approximately 5 seconds.
    3. **Stop** trigger:

       When the button is pressed and released, the motor stops immediately and holds its current position.
    4. Each subsequent button press toggles between running and idle states.
    5. An additional LED indicator must be used to represent the PWM duty cycle level. The LED brightness should vary continuously as the motor moves between −90° and +90°.

  - **Standard of grading**

    1. Properly handle the CCP1CON<5:4> bits when configuring the PWM duty cycle.
    2. Both C and Assembly language implementations are acceptable.
    3. The button input must be **edge-triggered with debounce**, meaning the action is triggered only once upon button release, preventing multiple activations while the button is held down.
    4. The motion must be executed incrementally, with smooth transitions between angles. Setting only the final position is not acceptable.
    5. Use Timer (e.g., Timer0, Timer1, Timer3) to handle the task. Timer2 may only be used for PWM. Any other approaches are prohibited.

- **Hint**

The following steps should be taken when configuring the CCP module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPRxL register and CCPxCON<5:4>bits.
3. Configure the CCPx pin as an output by clearing the corresponding TRIS bit.
4. Set the TMR2 prescale value, then enable Timer2 by configuring the T2CON register.
5. Configure the CCPx module for PWM operation.