Aaron Wong

Wongx654

Yahya Alhinai

alhin010

**Lab 4 Report**

Stages:

1. Testing the functionality of the Output Compare.
2. Testing the interrupt function with the IC activated by the button presses.
3. Edit the IC interrupt functions to calculate periods of button presses with overflow.
4. Save the periods of 4 presses in a buffer array of 4 elements then average them one array for IC1 and one for IC2.
5. Edit Timer 2 interrupt function to periodically check every second if any button has been pressed 4 times.
6. If any of the players wins toggle the LED corresponding to them if they win and set the servo to zero state.
7. Ignore presses with pulse widths less than 2 ms.
8. Adding the speed variable instead of using a step variable to shift the servo depending on who pushes faster in the T2 interrupt depending on the average taken from the buffer in the IC interrupts.

|  |  |
| --- | --- |
| 1. | Alternate between a 25% and 75% duty cycle on an LED |
| 2. | Using the debugging in MPLAB create a breakpoint in the interrupt function and test if pressing the buttons sends the program to there |
| 3. | Tested when there is a big delay between button presses |
| 4. | Using MPLAB’s variable watch, check to make sure the buffer and averages are correct with button presses. |
| 5. | Tested if statement in T2 Interrupt executing only when 4 button presses. |
| 6. | Tested the win condition for each player winning |
| 7. | Using debouncing to ignore presses with pulse widths less than 2 ms |
| 8. | Tested to see the servo move to the right side when each player is pressing faster |

Problems we had when coding this lab was first understanding the functionality of the OC and how it worked. When calculating the speed variable, the game was not working at 100% as it should have been.