University of Nevada, Las Vegas
Department of Electrical and Computer Engineering

CPE301L - Microcontroller Systems Design Lab

Final Project Report
Competitive Mini Game

Yanai Avila
Spring 2024

1. Introduction

Board Used: UNLV 328P board (board overview)

Language: C

Project Description: For my final project, I implemented a competitive mini-game. This 2-player mini-game resembles the arcade game called Cyclone where a player needs to stop the cycling light when it reaches the indicated red LED. For my implementation, an LED flashes back and forth between 7 different LEDs and the player has to press a button when one of the LEDs in the middle turns on. Unlike in Cyclone where the difficulty increases with each round, my implementation allows the player to adjust the difficulty using a potentiometer.

Game overview: A light moves back and forth between 7 LEDs at a fast rate chosen by the players. Each player needs to press their designated button when LED 5 is lit up. If the player presses their button when LED 5 is on, that player gains a point. The player that gets to 5 points first wins the game. Points are tracked on the serial monitor.

2. Implementation

- A **timer** (TIMER1 COMPA) is used to have each of the LEDs light up one after the other at a constant rate.
- **2 interrupts** (INT0, INT1) are used when the player presses their designated. Player 1 uses the SW1 pushbutton and player 2 uses the SW4 pushbutton.
- The **USART** interface allows for the players' points to be outputted to a serial monitor. I used the MAX232N dual driver/receiver chip for this one-way communication.
- A user-defined function called **LED_back_forth(delay)** takes in a delay value to control the rate at which the LEDs blink after each other. The higher the value, the faster the rate. The potentiometer is used to choose this delay.
- **PORTB** is the output port for the LEDS

3. Code

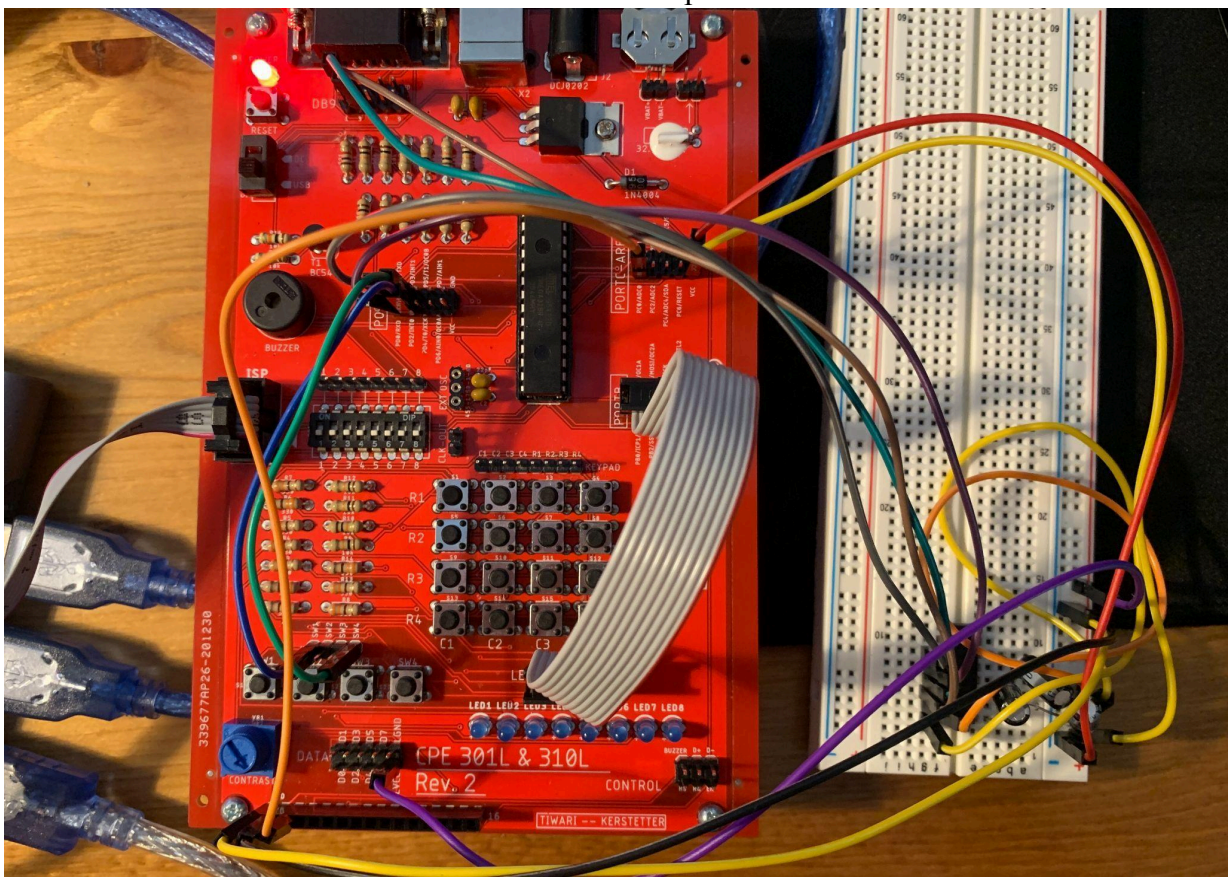
Main code

```
225  // MAIN PROGRAM
226  int main(void)
227  {
228      USART_Init(MYUBRR); // Initialize USART
229      ADC_Init(); // Initialize ADC
230
231      // PORTS
232      DDRB |= 0xFF; // port B as LED port (output port)
233
234      DDRC = 0; // Make Port C an input for ADC input
235      PORTC = 1; // Pull-up resistor
236
237      // Set PD2 (INT0) and PD3 (INT1) as inputs with pull-up resistors for buttons
238      DDRD &= ~((1 << PIND2) | (1 << PIND3));
239      PORTD |= (1 << PIND2) | (1 << PIND3); // Enable pull-up resistors
240
241      // Timer
242      TCCR1B |= (1 << WGM12) | (1 << CS12) | (1 << CS10); // Set prescaler to 1024
243                                                          // Mode: CTC
244      TIMSK1 |= (1 << OCIE1A); // Output compare A match interrupt
245      OCR1A = 1;
246
247      sei(); // Enable global interrupts
248
249      // For interrupts
250      EICRA |= (1 << ISC01) | (1 << ISC11); // Falling edge generates interrupt
251      EIMSK |= (1 << INT0) | (1 << INT1); // Enable INT0 and INT1 interrupts
252
253      while (1)
254      {
255          // Runs forever
256      }
257  }
```

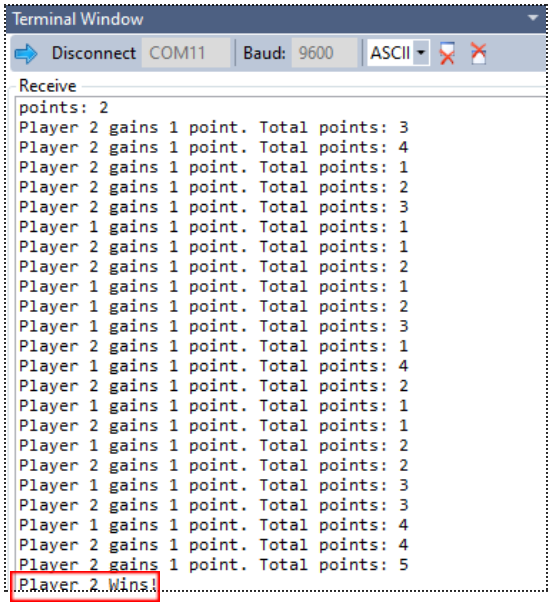
Code link: https://github.com/YanaiAvila/CPE301L/blob/main/CPE301L-Final-Project-Code.c

5. Board Setups and Output

Board setup



Microchip Studio Output



4. Video
Video link: https://youtu.be/WUGZxaSniOU

5. Future Improvements
The only thing about my game that I was not able to figure out was implementing the potentiometer to change the difficulty of the game. The difficulty is hard coded to the easiest setting as it is. However, I have variables for the 4 different difficulties, and choosing between them does change the rate at which the LEDs blink, so implementing the potentiometer would not be too difficult if given more time.