EGR-289

Final Project: Portable GPS with Physical time element

Lab Report

Harnoor Singh

Abstract–one sentence

This device combines GPS for real-time location display on a screen with a servo mechanism for physical time indication, and offers a versatile solution for dynamic time and location representation.

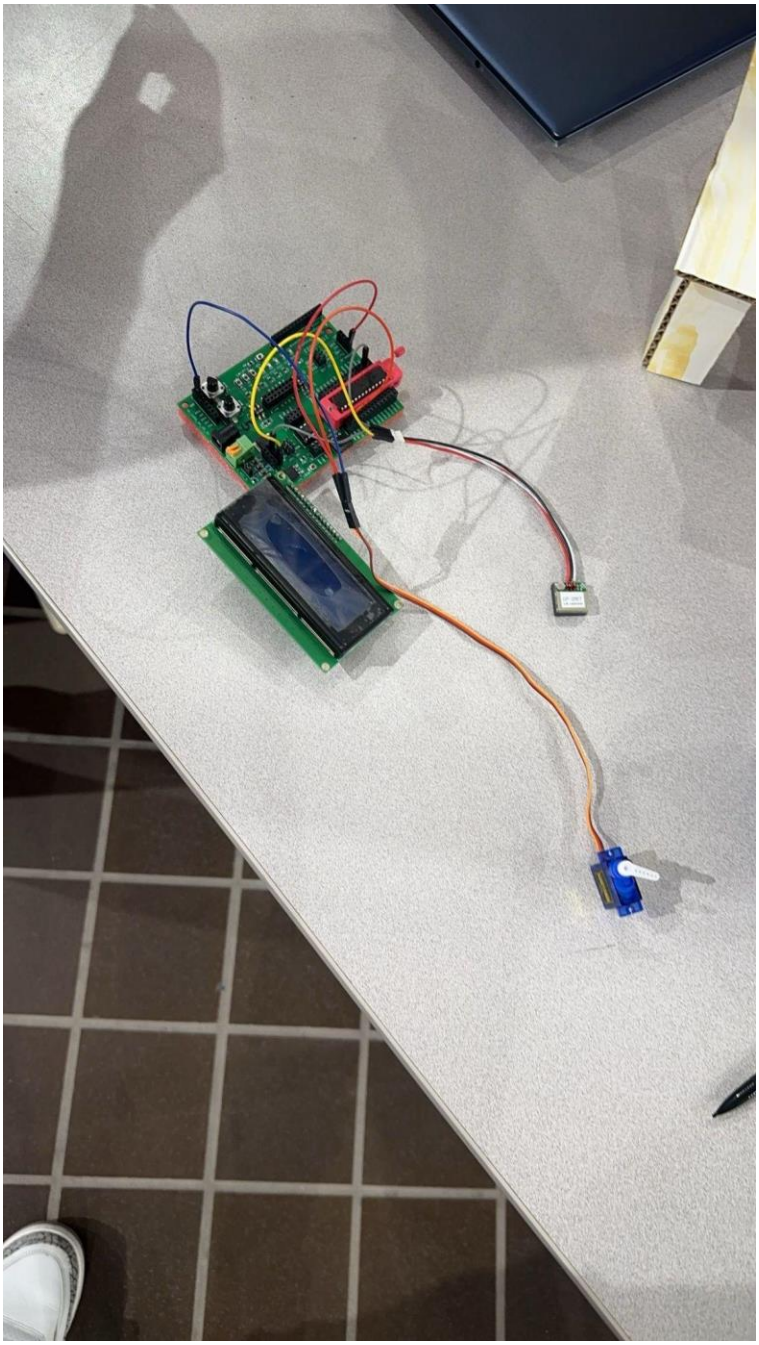
Problem statement

The problem lies in the need for a device that seamlessly integrates GPS technology to provide real-time location display on a screen while incorporating a servo mechanism for physical time indication, as the current market lacks a comprehensive solution that offers customizable options for both form factor and durability.

Description

This device integrates GPS technology to display real-time location on a screen while utilizing a servo mechanism to physically indicate the current time, offering a multitude of ways to show time and location, and allows users to manipulate the size and durability of the device.

Pictures



Code

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include "c:\avr\freq_328.h"
#include "c:\avr\i2c.h"
#include "c:\avr\lcd_16x2_i2c.h"

char serial_receive();
void print_info(char* data, int& h, int& m) ;

void cycle(int);



int main(void) {
	DDRB = 0b1110;



	class lcd_16x2_i2c lcd;


	freq_8mhz();
	UBRR0H = 0;
	UBRR0L = 103;
	UCSR0A |= (1 << U2X0);
	UCSR0B |= (1 << RXEN0);

	char a;
	do {
		a = serial_receive();
	} while (a != '$');

	const int n = 80;
	char c[100];
	for (int i = 0; i < n; i++) {
		c[i] = serial_receive();
```

```
}

int ho, mi;
print_info(c, ho, mi);


cycle(ho);

        /*
while (1) {
        for (int i = 0; i < 100; i++) {
                PORTB = 0b1111;
                _delay_ms(2000);

                PORTB = 0b0111;
                _delay_ms(2000);

                PORTB = 0b0011;
                _delay_ms(2000);

                PORTB = 0b1001;
                _delay_ms(2000);
        }

        //int rotated_value_ho = 360 * ho;
        */
        _delay_ms(5000); // 5-second pause

        cycle(mi);
        /*
        for (int i = 0; i < 100; i++) {
                PORTB = 0b1111;
                _delay_ms(2000);

                PORTB = 0b0111;
                _delay_ms(2000);

                PORTB = 0b0011;
                _delay_ms(2000);
```

```
                        PORTB = 0b1001;
                        _delay_ms(2000);
                }

        //      int rotated_value_mi = 360 * mi;

                _delay_ms(1000);
        }
        */
        return 0;
}

char serial_receive() {
        while ((UCSR0A & (1 << RXC0)) == 0);
        int value = UDR0;
        return static_cast<char>(value);
}

void print_info(char* data, int& h, int& m)
{
  class lcd_16x2_i2c lcd;
        int comma_count = 0;
        int i = 0;
        while (comma_count < 1)
          { if (data[i] == ',')     comma_count++;
                 i++;
          }

  h = (data[i  ]-0x30)*10 + (data[i+1]-0x30);
  m = (data[i+2]-0x30)*10 + (data[i+3]-0x30);

  lcd.dd(h);
  lcd.dd(m);
}

void cycle(int time){
        int i = 0;
        while (i < time){
                for (int i = 0; i < 200; i++){
                        PORTB = 0b10;
```

```
                _delay_ms(1);
                PORTB = 0b00;
                _delay_ms(19);
        }
        i++;
        if (i == time)
                return;
                for (int i = 0; i < 200; i++){
                        PORTB = 0b10;
                        _delay_ms(2);
                        PORTB = 0b00;
                        _delay_ms(19);
                }
        i++;
    }
}
```