**HSI Final Project 5**

Alireza Mohammadzadeh and Md Junain Muntasir

ID: 147106231 and 122291230

Date: August 14, 2024

**Contents**

[**Code** 2](#_Toc174471671)

[**HSI Discussion** 6](#_Toc174471672)

[**Circuit Details** 6](#_Toc174471673)

[**Picture of circuit**: 7](#_Toc174471674)

[**Picture of eagle pcb:** 8](#_Toc174471675)

[**Bill of Materials:** 8](#_Toc174471676)

[**Link to website:** 9](#_Toc174471677)

[**HSI Summary** 9](#_Toc174471678)

[**Breadboard Setup**: 10](#_Toc174471679)

# **Code**

/\*

HSI255 Project 5

Name: Md Junain Muntasir & Alireza Mohammadzadeh

Date: 14 August, 2024

Purpose: program begins with the LEDs off and the buzzer off. When the program starts, it will explain the

program and ask the user for-

Time: ask the user how many seconds they want the program to run, and the program will run that long.

Buzzer activations: the program will end after the sensor has been blocked the number of times the user specifies.

After the user makes their selections, the program will then begin monitoring the status of the optical sensor.

The LED will be lit to indicate that the sensor is not blocked, and the buzzer will be off. When the sensor is

blocked, the LED will turn off and the buzzer will turn on. When the sensor is unblocked, the LED turns back on

and the buzzer turns off. The console window updates the status of the sensor 4 times per second no matter the status.

\*/

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <Windows.h>

#include "C:/Program Files (x86)/LabJack/Drivers/LabJackUD.h"

int main() {

LJ\_ERROR lj\_cue;

LJ\_HANDLE lj\_handle = 0;

double sensorAIN1;

int counter = 0;

int exitStrategy, activations, buzzerActivations = 0, sensorBlockedCount = 0, firstBlock = 1;

int duration = 0;

// Initialize LabJack

lj\_cue = OpenLabJack(LJ\_dtU3, LJ\_ctUSB, "1", 1, &lj\_handle);

lj\_cue = ePut(lj\_handle, LJ\_ioPIN\_CONFIGURATION\_RESET, 0, 0, 0);

// Configure PWM signal for buzzer on FIO4

lj\_cue = AddRequest(lj\_handle, LJ\_ioPUT\_CONFIG, LJ\_chTIMER\_COUNTER\_PIN\_OFFSET, 4, 0, 0);

lj\_cue = AddRequest(lj\_handle, LJ\_ioPUT\_CONFIG, LJ\_chTIMER\_CLOCK\_BASE, LJ\_tc1MHZ\_DIV, 0, 0);

lj\_cue = AddRequest(lj\_handle, LJ\_ioPUT\_CONFIG, LJ\_chTIMER\_CLOCK\_DIVISOR, 1, 0, 0);

lj\_cue = AddRequest(lj\_handle, LJ\_ioPUT\_CONFIG, LJ\_chNUMBER\_TIMERS\_ENABLED, 1, 0, 0);

lj\_cue = AddRequest(lj\_handle, LJ\_ioPUT\_TIMER\_MODE, 0, LJ\_tmPWM8, 0, 0);

lj\_cue = Go();

// Turn off LED on DAC0 initially

lj\_cue = ePut(lj\_handle, LJ\_ioPUT\_DAC, 1, 5.0, 0);

// Welcome message and ask for exit strategy

printf("Welcome to the HSI255 Project program.\n");

printf("Choose exit strategy:\n");

printf("1. Time (seconds)\n");

printf("2. Number of buzzer activations\n");

printf("Choose your exit: ");

scanf("%d", &exitStrategy);

if (exitStrategy == 1) {

printf("Enter the duration in seconds: ");

scanf("%d", &duration);

}

else if (exitStrategy == 2) {

printf("Enter the number of buzzer activations: ");

scanf("%d", &activations);

}

else {

printf("Invalid selection. Exiting program.\n");

return 1;

}

printf("Press any key to start the program...\n");

getchar(); // To capture the enter key after scanf

getchar(); // To capture the actual key press

// Main loop

while ((exitStrategy == 1 && counter < duration \* 4) || (exitStrategy == 2 && sensorBlockedCount < activations)) {

// Read sensor state

lj\_cue = AddRequest(lj\_handle, LJ\_ioGET\_AIN, 1, 5.0, 0, 0); // Optical sensor on AIN1

lj\_cue = Go();

lj\_cue = GetResult(lj\_handle, LJ\_ioGET\_AIN, 1, &sensorAIN1);

if (sensorAIN1 > 1.0) { // Sensor is unblocked

lj\_cue = ePut(lj\_handle, LJ\_ioPUT\_DAC, 1, 0.0, 0); // Turn on LED on DAC0

lj\_cue = ePut(lj\_handle, LJ\_ioPUT\_TIMER\_VALUE, 0, 0, 0); // Turn off buzzer on FIO4

printf("Sensor is unblocked. LED is ON. Buzzer is OFF.\n");

}

else { // Sensor is blocked

if (firstBlock) {

sensorBlockedCount++;

firstBlock = 0;

printf("Sensor Blocked. Count: %d\n", sensorBlockedCount);

}

lj\_cue = ePut(lj\_handle, LJ\_ioPUT\_DAC, 1, 5.0, 0); // Turn off LED on DAC0

lj\_cue = ePut(lj\_handle, LJ\_ioPUT\_TIMER\_VALUE, 0, 32768, 0); // Turn on buzzer on FIO4

printf("Sensor is blocked. LED is oFF. Buzzer is ON.\n");

}

Sleep(250); // (4 times per second)

counter++;

if (sensorAIN1 > 1.0) {

firstBlock = 1;

}

}

// Turn off LED and buzzer

lj\_cue = ePut(lj\_handle, LJ\_ioPUT\_DAC, 1, 5.0, 0); // Turn off LED on DAC0

lj\_cue = ePut(lj\_handle, LJ\_ioPUT\_TIMER\_VALUE, 0, 0, 0); // Turn off buzzer on FIO4

printf("Program ended. LED and buzzer are turned off.\n");

Close();

return 0;

}

# 

# **HSI Discussion**

This project uses the optical sensors buzzers LEDs and buttons. One of the challenges of making this project was Seneca pcb taking a week just to tell us our pcb has errors. Second is finding the correct components on eagle. The third issue was trying to figure out the correct lab jack ports to use. The code was not too difficult as it is a combination of the codes of previous labs. This circuit combines what lab 6 and 8 do with the counter to end the loop from the optical sensor lab and the time for the code to end from the buzzer lab.

# **Circuit Details**

Ports used:

* DAC1
* AIN1
* FIO4
* GND
* VS

Parts used:

* Resistor 330ohm
* Transistor NPN
* Buzzer
* LED
* Optical sensor

# **Picture of circuit**:

A red and white electronic device with wires

Description automatically generated

# **Picture of eagle pcb:**

A blue and green circuit board

Description automatically generated

# **Bill of Materials:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Part | Supplier | Sup. P/N | Value | Specs | Qty | Cost per/ | Cost |
| Resistor | Digikey |  | 330ohm | 5%,1/4 W | 5 | 0.15$ | 0.75$ |
| NPN Transistor |  |  | 300MHz | 30V | 1 | 0.45$ | 0.45$ |
| Buzzer |  |  | 4KHz | 75dB @ 3V, 10cm | 1 |  | 1.28$ |
| Optical sensor |  |  | 8µs, 50µs | (3mm) | 1 |  | 2.43$ |
| LED |  |  | 4.6mm | 20mA | 1 |  | 0.81$ |
|  |  |  |  |  |  | Total | 5.72$ |

**Link to website:** <https://www.digikey.ca/en/products/detail/sunled/XLM2MR11W/4901623?utm_adgroup=General&utm_source=google&utm_medium=cpc&utm_campaign=PMax%20Shopping_Product_Zombie%20SKUs&utm_term=&productid=4901623&utm_content=General&utm_id=go_cmp-17855401585_adg-_ad-__dev-c_ext-_prd-4901623_sig-Cj0KCQjwiOy1BhDCARIsADGvQnAljFGWBkNyL-BEyLw5Pi7xGdSDOrO-ZSMvUKMXznDOfv39e1o6o3AaAuFuEALw_wcB&gad_source=1&gclid=Cj0KCQjwiOy1BhDCARIsADGvQnAljFGWBkNyL-BEyLw5Pi7xGdSDOrO-ZSMvUKMXznDOfv39e1o6o3AaAuFuEALw_wcB>

# **HSI Summary**

project integrates hardware and software components to create a responsive system that monitors and reacts to changes detected by an optical sensor. The hardware setup involves constructing a circuit on a breadboard, adhering to the given schematic. Key components include an optical sensor connected to AIN1, an LED connected to DAC1, and a buzzer connected to FIO4.

**Circuit Details:**

1. **Optical Sensor:**
   * **Connection:** The sensor is connected to AIN1 and configured with a pull-up resistor.
   * **Function:** It detects the presence or absence of an object, outputting a signal that the program monitors to determine the sensor's state.
2. **LED:**
   * **Connection:** An LED is connected to DAC1, with the anode connected to DAC1 and the cathode connected to the ground through a current-limiting resistor.
   * **Function:** The LED serves as a visual indicator, turning on when the sensor is unblocked and off when the sensor is blocked.
3. **Buzzer:**
   * **Connection:** The buzzer is connected to FIO4, configured to operate using a PWM signal.
   * **Function:** It emits a sound when the sensor is blocked, providing an audible alert.
4. **PWM Signal Configuration:**
   * The PWM signal for the buzzer is set up with specific parameters to control the duty cycle, ensuring the buzzer sounds appropriately when activated.

### **Breadboard Setup**:

The circuit is assembled on a breadboard, ensuring proper connections and placement of components:

* **Power Rails:** Connected to the LabJack U3 HV for power supply.
* **Optical Sensor:** Placed with appropriate connections to AIN1 and configured for pull-up.
* **LED and Resistor:** Connected in series between DAC1 and ground, ensuring the LED operates within safe current limits.
* **Buzzer:** Connected to FIO4 with PWM configuration to control its activation.

**Program Overview:**

The program starts by initializing the LabJack device and configuring the pins and PWM signal. It explains the operation to the user, who selects an exit strategy: either a time-based or event-based duration. The main loop monitors the sensor state and updates the LED and buzzer accordingly. If the sensor is unblocked, the LED turns on, and the buzzer remains off. When the sensor is blocked, the LED turns off, and the buzzer activates.

The project demonstrates a practical application of interfacing sensors and actuators with a microcontroller, emphasizing both hardware assembly on a breadboard and software logic to achieve desired outcomes. This hands-on experience reinforces understanding of electronic components, circuit design, and embedded programming.