**Optical Sensor Alarm with Buzzer**

Muhammad F. Khan

School of Electronics & Mechanical Engineering, Seneca College

HSI255: Hardware and Software Interfacing

Goby Gunabalasingam

**Optical Sensor Alarm with Buzzer**

The aim of this lab was to use an optical sensor to control and monitor the voltage supply to a Led and Buzzer. Whenever the sensor was to be disrupted, both the led and buzzer would turn on. The physical part of the lab required resistors, a photo interrupter, a piezo buzzer, a LED, and some jumper wires.

**Gathering PCB Components**

Five resistors were among the first items gathered. Two 330-ohm resistors, one 10-kiloohm resistor, one 1-kiloohm resistor, and one 470-ohm resistor The Optical Sensor, a Piezo Buzzer, a Green transparent case LED, an NPN transistor, and 5 female header pins were the remaining components. DigiKey was used to collect all of these pieces, which were then compiled into a list. The total cost of these components came to $8.69 in total.

**Process of Creating the Schematic**

The Fritzing program was used to create the circuit's schematic and connect the components. To begin, a 330 ohm resistor was connected to ground in parallel with the optical sensor's collector terminal, while the emitter terminal of the sensor was connected to ground. The sensor's anode was also linked to ground, and a 10 kiloohm resistor was used to link the cathode to the voltage source. As a current limiting resistor, another 330 ohm resistor was connected to ground in series with the LED, which was connected to a flexible I/O. Since the buzzer had no polarity, the right lead was connected to the collector terminal of the NPN transistor via a 1 kiloohm resistor, while the left lead was connected directly to the voltage source. A 470 ohm resistor was used to link the NPN transistor's base to an analog output, while the emitter was connected to ground. In the end, the schematic came out fairly similar to the project instructions, but with a few added details such as labelling of the Analogs Outputs and Inputs and the values of the resistors.

**Process of Creating the PCB**

The PCB was created with the help of PADs Layout and Router. Pads Layout was used to insert all of the circuit board's elements, and router was used to attach them. To begin, 5 female header pins were used to represent the following: VS, GND, FIO4, FIO6, and AIN2. These five laid the basic domain for the PCB. The resistors were placed in fairly the same way as the schematic, but a couple issues did arise as the connections for the ground kept getting aligned incorrectly. After some tweaking, the design came out smaller than expected. The name was then inserted on the top left in a Silkscreen Top layer.

**Process of Building the Circuit**

The circuit was built on a breadboard using the components from the Digi Key LabKit. At first, the schematic was directly followed however when it came time to insert the buzzer, a little adjustment had to be made. Instead of using the right lead for the source voltage, the left was used, and the right was for the NPN collector terminal. In addition, originally the LabJack DAC1 was going to be used as the analog output, but it was giving a slightly slow response, so instead a flexible I/O was used. Aside from that, the breadboard came out exactly like the schematic.

**Programming the LabJack**

The circuit was programmed in Visual Studio 2019 using a LabJack library along with 3 others which included the stdio.h, stdlib.h, and windows.h. For the first part, 7 variables were declared and initialized. Two of them were the error handlers, and the other 5 consisted of three doubles variables and two integer variables. Their names being dblValueDAC1, dblValueAIN2, dblValueFIO4, dutycycle, and counter and they were all set to 0. Section 2 was used to set up the handle and reset all ports to factory condition. For sections 3 to 8, the buzzer was set up using the PWM sequence of 5 AddRequests. First, a timer was configured to use FIO7, then a timer clock base of 48MHz/Div was set. As for the divisor, it was tested with multiple values until one where the buzzer did not create an irritating sound was found, and that was 48. Sections 6 to 8 involved enabling a timer, setting the timer mode to 8 bits, and calling the Go function. Next came the while loop which was set using a counter. The condition was set up so that if the counter surpassed 60, the loop would be terminated. This was due to method 2's requirement on the Project Instructions, which stated that the program had to run for 15 seconds. After that, an ADD-GO-GET function sequence was used to display the value of AIN2 in the terminal window. In order to turn the buzzer on, the sensor would have to be interrupted and so to do that AIN2 was originally set to 0 so that every time the sensor was blocked, its value would increase causing an if else statement to initiate in which the dutycycle would rise to 50% if the value of AIN2 exceeded 1. Section 11 was used to set the dutycycle with the PWM sequence. For the LED, the same if-else statement was used, but instead of the duty cycle, FIO4 was set to high whenever the sensor was interrupted. After the else statement, a counter iteration was added, and a sleep function that would display the value every 0.25s because of the requirement in the instructions. The last few sections were for pausing the system after the program stopped, resetting all LabJack terminals and closing it.

**Final Outcome**

The final outcome of this project was surprisingly clean. The gathering of components and making the schematic turned out to be the easiest part, and making the actual circuit was fairly easy as well. But it did take some time to get used to the UI of PADS which ended up taking more time than necessary. In the future, it would be a better idea to get experience using a software early. It saves much more time and can help produce a better product. Overall, this project was an exciting experience, and taught many lessons which could help with other Labs and Projects in the near future.

Table 1. The total cost of the components required to build the Printed Circuit Board

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Parts** | **Supplier** | **Sup. P/N** | **Value** | **Specs** | **Qty** | **Cost/** | **Cost** |
| Resistor 1 | Yageo | CFR-25JB-52-330R | 330Ω | 5%, 1/4W | 2 | $ 0.15 | $ 0.30 |
| Resistor 2 | Yageo | MFR-25FBF52-10K | 10kΩ | 1%, 1/4W | 1 | $ 0.14 | $ 0.14 |
| Resistor 3 | Yageo | MFR-25FBF52-1K | 1kΩ | 1%, 1/4W | 1 | $ 0.14 | $ 0.14 |
| Resistor 4 | Yageo | CFR-25JB-52-470R | 470Ω | 5%, 1/4W | 1 | $ 0.15 | $ 0.15 |
| Transisitor | Central Semiconductor Corp | 2N3904 PBFREE | 40V(Max Range) | 2N3904(NPN) | 1 | $ 0.64 | $ 0.64 |
| Buzzer | TDK Corporation | PS1240P02BT | 30V(Max Range) | Transducer | 1 | $ 0.89 | $ 0.89 |
| PhotoInterrupter | Lite-On Inc. | LTH-301-07 | 30V(Max for NPN) | Through Beam | 1 | $ 1.65 | $ 1.65 |
| LED | Cree Inc. | C503B-GCN-CY0C0791 | 527nm | Led Green Clear | 1 | $ 0.33 | $ 0.33 |
| Female Header Pins | Preci-Dip | 801-87-006-10-001101 | N/A | Through Whole | 5 | $ 0.89 | $ 4.45 |
| Total |  |  |  |  |  |  | $ 8.69 |

Diagram, schematic

Description automatically generated

Fig. 1. The updated schematic diagram of the Optical Sensor Alarm with Buzzer circuit

Diagram

Description automatically generated

Fig. 2. Top part of printed circuit board

Diagram, schematic

Description automatically generated

Fig. 3. Bottom part of printed circuit board

A close-up of a circuit board

Description automatically generated with low confidence

Fig. 4. Breadboard build of the circuit