CS 362 Arduino Project

Holiday Safety Tree

Team Members

Cesar Hernandez, Ubemio Romero

Light Arduino

Purpose

Our goal for this project was to construct a tree made up of LED's that would light up in time for the holiday season. Along with holiday music playing in the background, but with the LED's being controlled a photoresistor so that it knows when the room is dark enough for the tree to light up. We also decided to integrate a thermostat in case of a holiday tree fire it can detect the temperature and it would then enter a panic mode where it will sound an alarm and the LED's will enter panic mode, so it notifies the owner that there is a fire around the tree. We also decided that an LCD screen integration would be helpful so that the owner knows what the temperature readings are at. We also wanted the tree to have more flair in the LED's instead of the usual always ON or always OFF mode, so we decided it would be great if we can create some pattern that will change in a set amount of time.

Utilities/Idea

In this project we utilized two Arduinos: One Arduino which we call the Light Arduino is to control the LED's and make use of the photoresistor and the other Arduino which we call the Temperature Arduino (T. Arduino, for short) oversees having the LCD screen, buzzers and thermostat. We begin from the first day where we gathered all the Arduino kits and supplies and we at first had planned to make the tree to be covered in LED's thru its whole base (Fig:1.0), but after reflecting and revising our soldering skills we agreed we are not skilled enough in it to do such a thing so we had to go about lighting the tree up a different way so we decided that we should use the spherical ornaments(Fig:1.1)and place a mix of LED’s inside each one. So, we start at the basics testing and making sure the basics are up to code we begin our work on a breadboard to see how the circuit will and should go about (Fig:1.2).

Fig:1.0 Fig:1.1

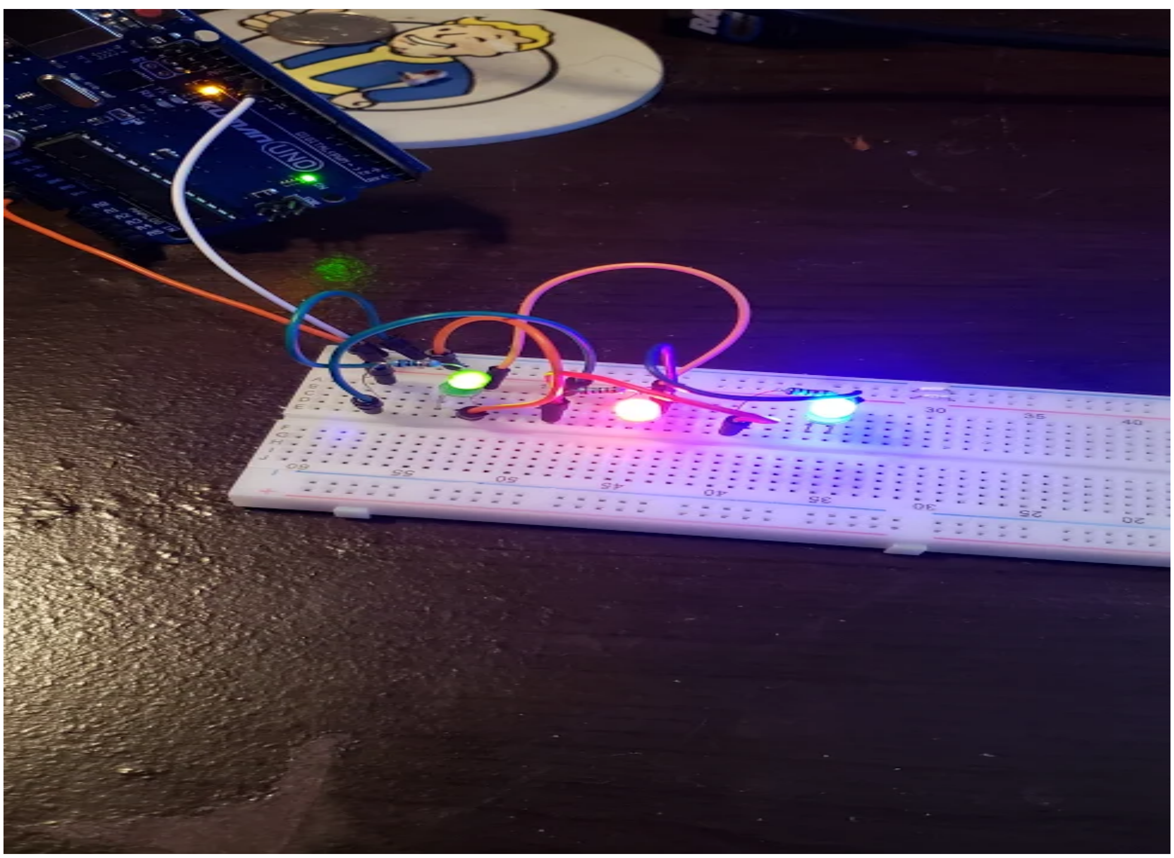


Fig: 1.2

Design

After having it work properly on the breadboard, we were ready to start soldering these trios of LED's together to have them ready to insert them into the ornaments. We decide to use the Red, Green, and Blue LED's that came in our Arduino kits we then needed some exposed wire so that we can solder on it to connect the three LED's. Our first problem came in the shape of the wire we had provided in our kits apparently if we tried to strip the colored plastic part from the wires it would not wore because we were expecting clear metal wire like on the end of it instead we were met with many small strand of what appeared to be copper wire(Fig1.3). After this discovery we decided we cannot use those wires for that so we deiced we can still use them to connect the ends to go to the Arduino, but after many attempts to solder the tip of the wire to the resistor it would just repel and slide off all solder that it would come in contact with just like water and oil. Here we concluded that these wires are solder proof hence we cannot use them at all, unless we use connected them to Dupont wire (Fig:1.4) these wires in our kits help make us of the other wires we had in our kits to help connect them and help close the gap and distances of wires we must solder.

Fig: 1.3

Fig: 1.4

Troubleshooting

After this hurdle was addressed and fixed, we began soldering these trios of LED's together we lined up all three LED's together and then soldered them on clear metal wire that we got by stripping the plastic part from them and note this wire was obtained thru another classes kit for labs. We soldered the LED's power sides all on one wire and then on the other we soldered 220 OHM resistors and then the resistors to another separate wire. Then result was Fig: 1.5, and after our first prototype was finished we took it to the Arduino to test to see if it work lighting up (Fig: 1.6) and after the success we began the mass production of more of them once we had a couple we hung them on the tree to see how the end result would appear (Fig: 1.7). From here we knew for sure that this was not the way to go about it, so we decided to buy spherical ornaments that were clear and plastic. Unfortunately, we were unable to find some that met our requirements as many of the clear ones were either too small or too big and made of glass. We thought that perhaps the glass would melt like in the shows of how they melt glass to reshape it but then we also thought of the case that it might shatter with a certain amount of temperature, so to be safe we stuck to the plastic only rule. After purchasing some plastic ornaments that weren't clear but met the plastic and size criteria we decided that perhaps we can make holes for the LED's to pop out thru inside the ornament. So, we decided to do the top part of the ornaments melt the top off to be able to access the inside (Fig: 1.8).

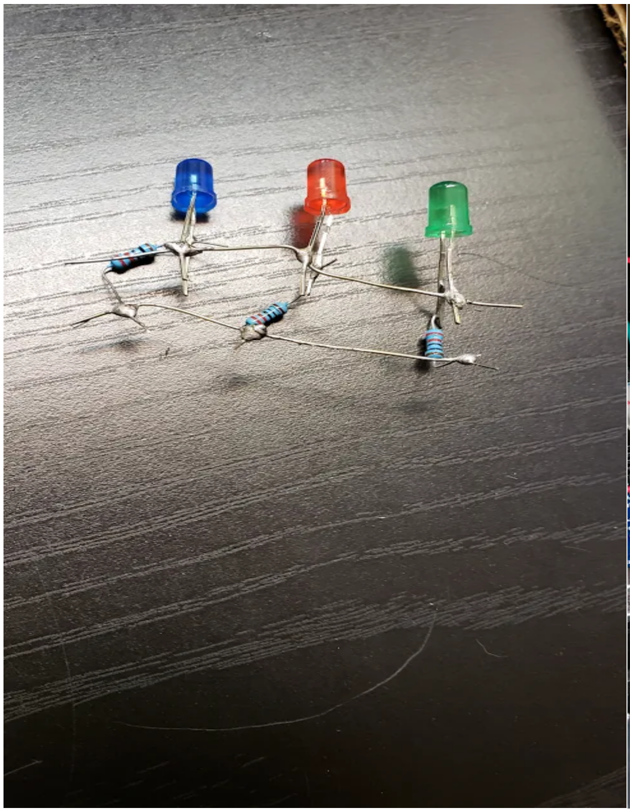


Fig: 1.5

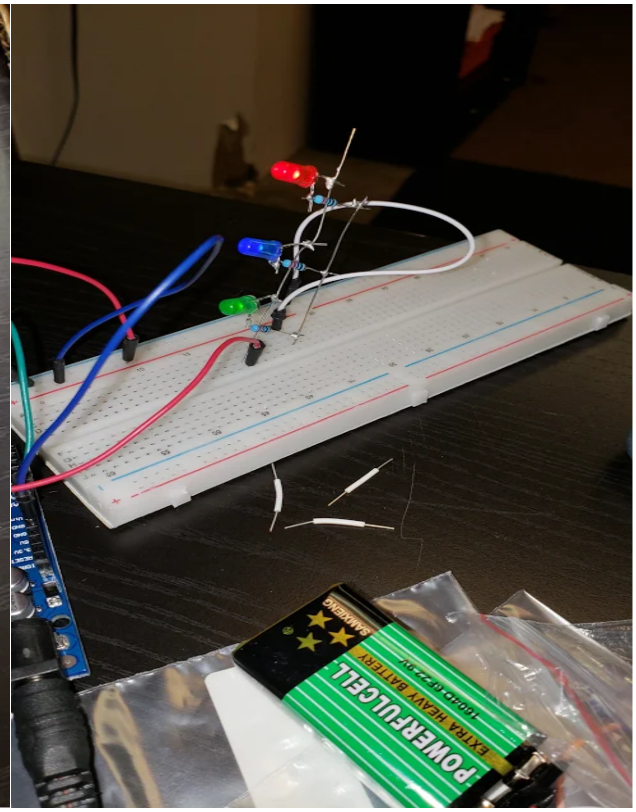


Fig: 1.6

Fig: 1.7

Fig: 1.8

Implementation

After having access to the inside of the ornaments we had to design a method of organizing the ends of our lights to work properly we needed and exit hole and a power hole for the ornaments to work proper and it needed to be constant and done the same for all the other ornaments. So, we decided that the power wire would exit on a side from the sphere and the ground wire would come out thru the bottom so that they are distinct from one another. We then melted the holes on the roof of the ornaments to make space for the LED's to pop out from so that their illumination can be seen, then we needed some way to close of the lid back onto the ornaments and the answer was a glue gun (Fig: 1.9).

Once we had a full-on ornament in our hands all that was left was to test it to see if it would light up and work we connected the ground wire and power wire to their designated areas and the results came in our favor as the ornament lit up brilliantly (Fig: 2.0).

Fig: 1.9

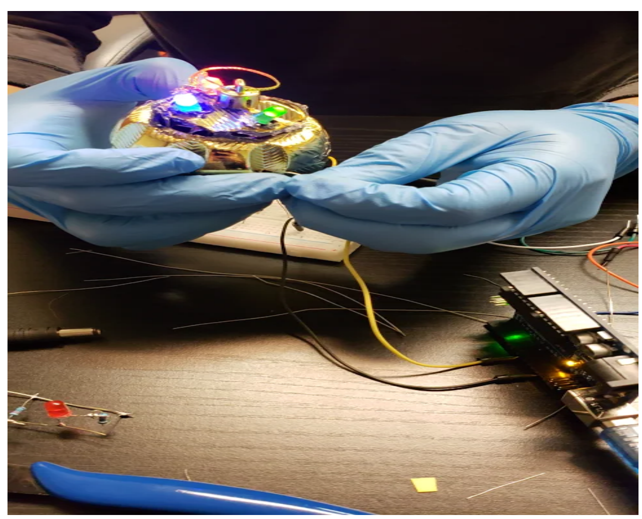


Fig: 2.0

Next up was to finish up all the rest of them and continue the cycle that we did to make the original prototype at this point we ended up fitting 5 ornaments into our tree, so we hung them up in the tree and then began to solder all their grounds together as it doesn't matter to share a common ground. From here we moved on to soldering the power ones together we started with a pair to check if they would both power on and they did, we were positive they would but then came the tricky part if it would if three of them shared power. Here to prevent a soldering mistake we grabbed the third ornament's power wire and connected it to the pairs power with just a contact within wires to see if they would power up and they did then we brought the 4th one to see if they still would and they did. The 5th ornament was the black sheep of them all as it contacting the others power connection made the power dim a lot and if not completely. Therefore, our solution to this was that the power had reached the max amount it can supply to the others, so this one by itself would be our center and have its own power supply provided to it. The quadruple ornament received one pin and the lone wolf one received another pin to itself both of them digital (Fig: 2.1).

Code

Once we had the trees wire setup basically all complete next up for the Light Arduino was to code up some patterns so that the LED's when they are lit up have some flair to them. The meat of the code comes in the functions that were created that are called in the loop of the main code, for example we made a function called allOn() which makes all the LED's stay on and another called allFlash() which makes all the LED's flash. The way we came up with making these work congruently was to have a global counter that would aid in knowing when it's time to end the call of allOn() or allFlash(). It would increment at the end of each loop in each function and when it reached a certain count it would return to the main loop to be able to call the next function hence start the next pattern. The important part for our counter to work correctly was that it needed to be set back to zero in the last function once it reached a specific number so that way it would be able to start back at the original pattern rather than just stay in the last pattern till the power is cutoff, so by resetting the counter it allowed us to reset the pattern back and help the loop connect itself properly.

Next up was setting up the photoresistor to work with the LED's to let them know when to light up and when not. Starting back in the basics from lab 5 we go back to see how to get the photoresistor to work properly, we set it up on the breadboard check how it is and open the serial monitor to check that it is working and responding correctly. Once those values looked legitimate we soldered its resistor and the photoresistor correctly and soldered that part to common ground we also soldered a wire in between both connections to have it connect to A0, its analog pin. Moving on to solder its power side that would connect to its power supply as it seemed no to work when we connected it to the shared power of the ornaments and apart from that it needs its own power to be constant it cannot be manipulated by the patterns in the code.

Finally, the light Arduino had all its components ready and operational the last part was to code up the photoresistor to work with the LED's. With the help from the labs the code wasn't hard to make but to place it was another story. To make the code we simply had the analog pin read in values and assign them to a variable that we then interpreted to a certain value where we deemed dark enough to turn on and lit enough to turn off. Then came the part that it would turn off only certain times in the code the bug was that it had to be checking the light in the room not only in the main loop but within each function/pattern constantly so that we can ensure that it reacts as soon as it reads that its dark or lit. The final touch on the light Arduino was finished and it was working correctly in the dark/reacting to the light (Fig: 2.2, below)

END RESULT OF LIGHT ARDUINO

Temperature - Arduino A.K.A T.Arduino

Purpose

The purpose of this Arduino is to involve the functions of reading the temperature of the room and working alongside with the buzzer.

Utilities

In order to accomplish this, we used the LCD that was previous solidarize and connect it using the breadboard. The LCD will be used to display the current temperature reading. When the temperature is above, say, 75 degrees Fahrenheit it will cause the buzzer to behave and react on a constant pitch tone. The tone will sound like some sort of panic mode alarm. This will be easily distinguished and understandable because if the temperature is below 75 degrees Fahrenheit then the buzzer will behave normal and start to play a Christmas carol.

In our case we decided that it will play the Jingle Bells melody. We have also included a potentiometer to control the volume of the buzzer. That way if we want the melody to play at. A high or low volume, this will be possible. In addition to these functions, if the temperature is too hot the LCD will also display a message saying that it is hot, likewise it will show a message if it normal. Thus T. Arduino is now an indicator and outputting sound to represent whether or not it meets room temperature conditions.

Design.

In the T. Arduino design, we have standard LCD pins (13,12,5,4,3,2) to get the LCD to work accordingly with the potentiometer to balance out the light being displayed in the LCD.

Furthermore, T. Arduino also include 2 Ground pins. It has a ground pin above pin 13 that goes directly into the breadboard to ground the necessary resistors, LM35 and that of the LCD. The other ground is the ground being shared between both Arduinos (T. Arduino Light. Arduino). This ensures the ground serial communication between both Arduinos.

In addition to ground, T. Arduino does have a 5v pin that gives power to the LCD, the LM35, and the buzzer. Since we are using a buzzer, the buzzer is connected to pin 8. The buzzer implantation design is fairly simple, from the gathered information of LAB 2 we we're successfully implement the buzzer without much trouble along with the potentiometer.

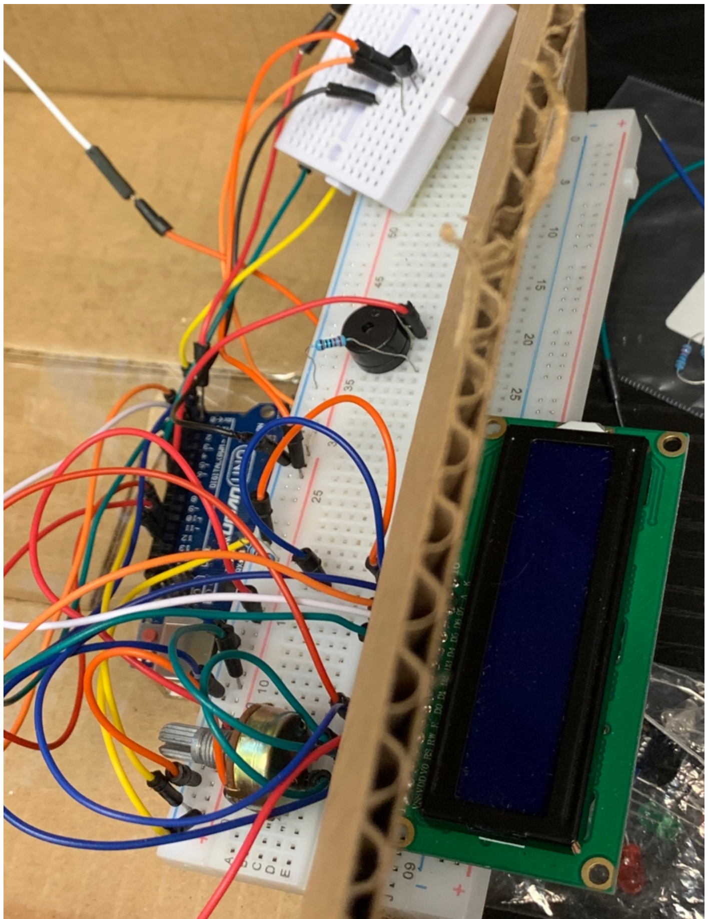
Lastly, T. Arduino uses an extended breadboard an addition to the regular breadboard where the LCD and the buzzer are all connected in. T/he extended breadboard is only for the LM35. The LM35 is a device that will help us receive and read in temperature values that from the room it's in. The LM35 has a ground wire and a power wire. But just like the Photoresistor, it also needs an analog pin. Therefore, the LM35's middle pin goes directly into pin A0, which establishes analog signal into the T. Arduino.

Design Decision's

We took some decisions into how our design would look like. T. Arduino has two boards. One board has the LCD, the buzzer, and the two potentiometers. The small board holds the LM35 device. Our design will be inside a box. We decided to make an opening on top of the cardboard box directly on top of the buzzer to make it look like a speaker, our own way of distributing bass. On one of the sides of the box we cut through the middle horizontally to place the LCD outside of the box and visually present the messages being read. That way we hide the unnecessary wires.

Troubleshooting

We did encounter a couple of errors. Some troubleshooting had to occur to test that the buzzer was working correctly. At first sound would come out but no melody would be presented. Until our wires we're placed correctly. LCD did not give us any errors since we have been using them throughout the semester. The LM35 design had some major issues. Before implementing the LM35 we were using another device called J1Con3 with a 10k resistor soldier in provided from the Arduino kit. That did not work at all because we we're getting values that did not correspond to the room temperature. We made sure by using the house's thermostat. Therefore, we proceeded to use an LM35. The LM35 did not require a resistor, just ground, power, and an analog input. However, when connecting the wires and putting everything together. If not careful a small amount of smoke would arise from the LM35. That was due to polarity. Meaning ground has to go with ground and power with power. At first, we made a mistake by doing the opposite when connecting the wires accordingly due to no direction from the LM35 how it should be connected. It was no reason why the LM35 read such high temperature values. The device itself got really hot.



This is design we went with for the following T. Arduino.

Main breadboard has LCD, Buzzer and potentiometers.

Small board has the LM35 device.

Code

For the LCD code implementation, this was done by going back into the Lab 3. This was made possible from a variety of labs that we worked on throughout the semester. Therefore, we did not have trouble implementing this code and reused old code.

The buzzer is broken down from the LCD code. The LCD main priority was to display the initiated value of temp. From temp we can assign it to a specific value and if that value is less than or greater than it will tell the buzzer to play a certain melody.

The Jingle Bell melody code is composed of chars and an array. This was orchestrated to follow a certain beat from. The given chars to the corresponding beats that of the integer of the array. For this special case scenario since, we did not have a lot of practice orchestrating music we had to do some digging from online and use a composed beat. From there we would use and edit the code to play the melody to the way we would like it to behave.

Finally, the temp code was also fairly simple. Since we did work with time in one of the labs, we had the same idea to initialize some values. Convert the Fahrenheit formula into code. But before that ensure that we interpret the voltage value into code. All simple initializers. From there use the read values from the LM35 and work our way up to display working degree Fahrenheit values.

Serial Communication

For this current section There is one problem that we do face and that's the establishment of both Arduinos communication. Now we did incorporate that the LED's would turn off if pressed from a button. This process is communicating between both Arduinos. For example, the data of the pin is going to the, Light Arduino from the T. Arduino. It will be input/send through the push button and output/received through the LED's. However, in case for some reason this does not work again like we had it running our alternative solution will be to use the same method expect this time the output will be the buzzer. Currently our buzzer doesn't turn off, so we figured that maybe we should have a present from serial communication to turn off the buzzer when the user please by pressing the push button.

References

https://www.youtube.com/watch?v=Zu3TYBs65FM	Soldering tips to improve soldering skills
https://www.youtube.com/watch?v=lQroCVV1QZ0	How to correctly solder an LED, improve from here to do three
https://www.arduino.cc/en/Reference/AnalogWrite https://www.arduino.cc/en/Reference/AnalogRead	Used for analog code that required it to be read and wrote
http://playground.arduino.cc/Learning/PhotoResistor http://learn.adafruit.com/photocells	Photoresistor build and setup from labs
http://fritzing.org/projects/esp8266-and-temperature-sensor-lm35	Help with setting up the LM35 component
https://www.arduino.cc/en/Tutorial/Button	Button lab setup to see how it was setup and suppose to work
https://www.arduino.cc/en/Tutorial/LiquidCrystalScroll	LCD lab pin setup
https://gist.github.com/elubow/7844436	JingleBells tune for the buzzer