PROJECT NAME: DRAGONPOD

Designed by Jianzhi Long, Yanye Li, Tingyu Wang

Introduction

Statement of Purpose

Usually, people's mood is affected by the surrounding, and they their preference of music changes accordingly. For example, their choice of music in a sunny day is different from that in a rainy night. Therefore, we propose to design a smart background music player that plays the music which fit in the mood of listener based on the surrounding environment. The unique selling point of the product is that it automatically select songs for users. Sometimes, a new song could elevate the user's state of mind which he or she could not possibly know without the smart BGM player.

Features and Benefits

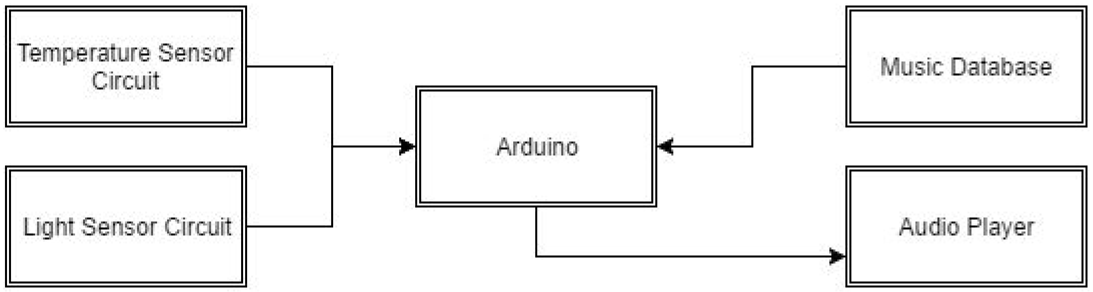
The product uses light and temperature sensor to collect data from the surrounding and Arduino IDE based algorithm will be used to process the information collected and select the corresponding music to play. The benefit of the Dragonpod is it is beneficial for improving mental health. When the user is too bothered by troubles to find something to make them feel better, the Dragonpod will provide that for them, and music is known to improve people's mood.

Design

System Overview

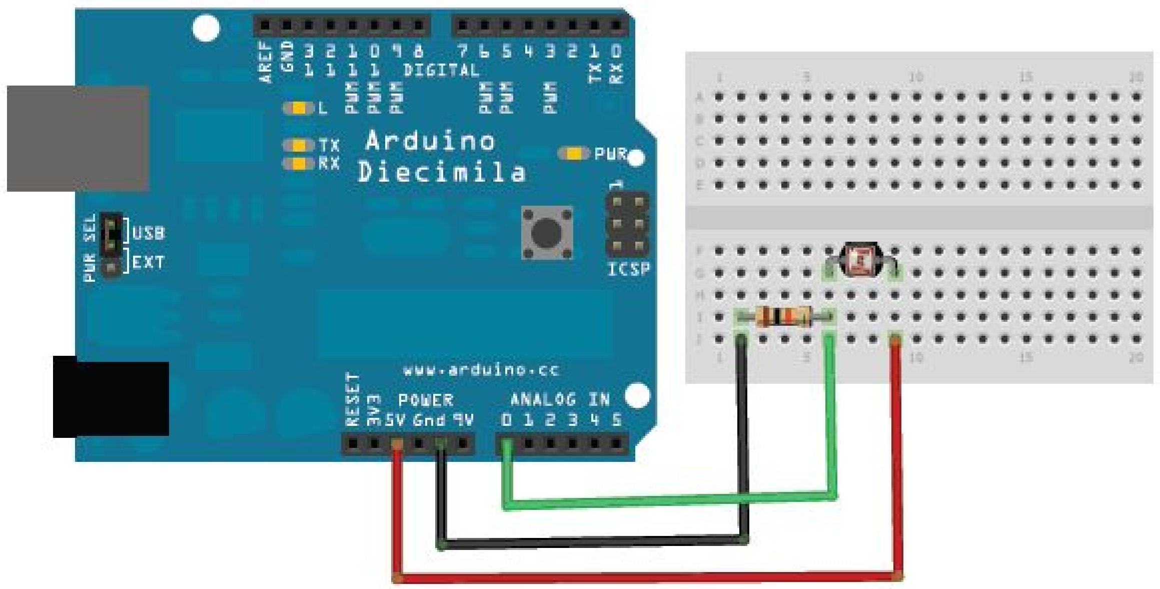
The temperature sensor measure the room temperature, which is affected by seasons, weathers and time in the day. The light sensor measures the light intensity in order to distinguish dim light and relatively intensive light. The combination of these factors could affect the user's mood. The algorithm then combine the information together and randomly select the music from the database that consists of music of a certain category that fits this surrounding. The categories of music corresponds to a certain range of temperature and light intensity combination. In this project, there are 15 songs in the database. Finally, music is played on the computer which is connected to Arduino.

Design Detail

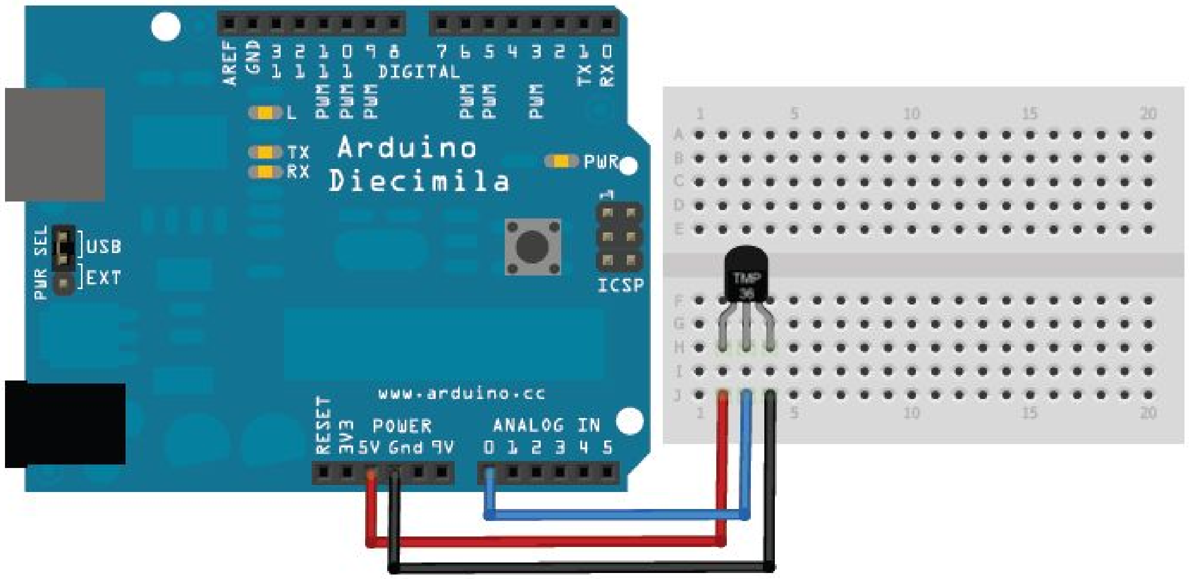


Circuit diagram

Light Sensor Circuit



Temperature Sensor Circuit



Results

After we put together all the components of the BGM design, we test on the range of the thermistor and photo sensor so that we can have the number of categories. Firstly, we test on the readings of the photocell under conditions of lab lights, the flashlight of IPhone and when we touch on the photocell to create a dim condition. The readings floats like 30 to 40 so we make three major division for the photocell----below 60, between 60 to 100 and above 100. Secondly, we did the same thing on the thermistor and record the values of the readings. Finally, we have a set of division of 15 conditions and since we only have 15 songs in the music library, we assign the proper music to the exact combined condition. Eventually, whenever we applied any of the 15 conditions, an identical song will be played.

Problems and Challenges

There were several problems we have encountered. There was a connection problem on the light sensor circuit that caused some time to be detected and solved. The most challenging problem was writing the algorithm. Although the TA had kindly provided us a song-selector program, we spent a lot of time to figure out the logic behind the coding. The use of the command window is also new to us.

Future Plans

In the future we plan to elaborate on the existing structure of the project by increasing the categories of different surroundings, which will be done by editing the code. Also, we are considering to implant more sensors for example, humidity sensors, thus making the device more accurate. Last but not least, we plan to enlarge the music database to hundreds or even thousands of songs of different categories, therefore providing more variety for the consumers.

Appendix

```
#include <CapacitiveSensor.h> //library that can be downloaded from
http://labs.arduino.org/Capacitive

int LEDpin = 9;
int count;//initialize the integer variable ‘count’
float volume = 0.5;//set for the value of the volume
int thermistorPin = A1;//set the portal pin for thermistor
int thermistorReading;//initialize the integer variable ‘thermistorReading’
int photocellPin = 0;//set the portal pin for the photocell
int photocellReading ;//initialize the integer variable ‘Photocell’


/**
 * Method run at the beginning of the code.
 */
void setup() {
  Serial.begin(9600);
  count = 0;//assign 0 as the initial value
  pinMode(LEDpin,OUTPUT);
}

/**
 * A loop that is iterated through continuously as part of the Serial.begin()
method.
 */
void loop() {
  photocellReading = analogRead(photocellPin);//read in the value from the
photosensor
  thermistorReading = analogRead(thermistorPin);//read in the value form the
thermistor



  if(thermistorReading<120 && photocellReading<60){//first class of division
    count=1;}//assign number of the song to the variable ‘count’
    else if(thermistorReading<170 && photocellReading<60){//2nd class of division
    count=2;}//assign number of the song to the variable ‘count’
     else if(thermistorReading<200 && photocellReading<60){//3rd class of division
```

```
   count=3;}//assign number of the song to the variable ‘count’
    else if(thermistorReading<900 && photocellReading<60){//4th class of division
   count=4;}//assign number of the song to the variable ‘count’
    else if(thermistorReading<170 && photocellReading<60){//5th class of division
   count=5;}//assign number of the song to the variable ‘count’
    else if(thermistorReading<120 && photocellReading<100){//6th class of
division
   count=6;}//assign number of the song to the variable ‘count’
   else if(thermistorReading<170 && photocellReading<100){//7th class of division
   count=7;}//assign number of the song to the variable ‘count’
    else if(thermistorReading<200 && photocellReading<100){//8th class of
division
   count=8;}//assign number of the song to the variable ‘count’
    else if(thermistorReading<900 && photocellReading<100){//9th class of
division
   count=9;}//assign number of the song to the variable ‘count’
    else if(thermistorReading<170 && photocellReading<100){//10th class of
division
   count=10;}//assign number of the song to the variable ‘count’
   else if(thermistorReading<120 && photocellReading<900){//11th class of
division
   count=11;}//assign number of the song to the variable ‘count’
   else if(thermistorReading<170 && photocellReading<900){//12th class of
division
   count=12;}//assign number of the song to the variable ‘count’
    else if(thermistorReading<200 && photocellReading<900){//13th class of
division
   count=13;}//assign number of the song to the variable ‘count’
    else if(thermistorReading<900 && photocellReading<900){//14th class of
division
   count=14;}//assign number of the song to the variable ‘count’
```

```
    else if(thermistorReading<170 && photocellReading<900){//15th class of division
   count=15;}//assign number of the song to the variable ‘count’
  Serial.print(count);//display on the computer of the variable ‘count’
  Serial.print(" ");
  Serial.println(volume);//display on the computer of the volume it’s playing currently
  Serial.println(photocellReading);//display on the screen the reading of the photocell



  digitalWrite(LEDpin, OUTPUT);
   delay(1000);
}
```

Reference

1. "Using a Photocell" [online] Available:

   https://learn.adafruit.com/photocells/using-a-photocell

2. "Using a Temp Sensor" [online] Available:

   https://learn.adafruit.com/tmp36-temperature-sensor/using-a-temp-sensor