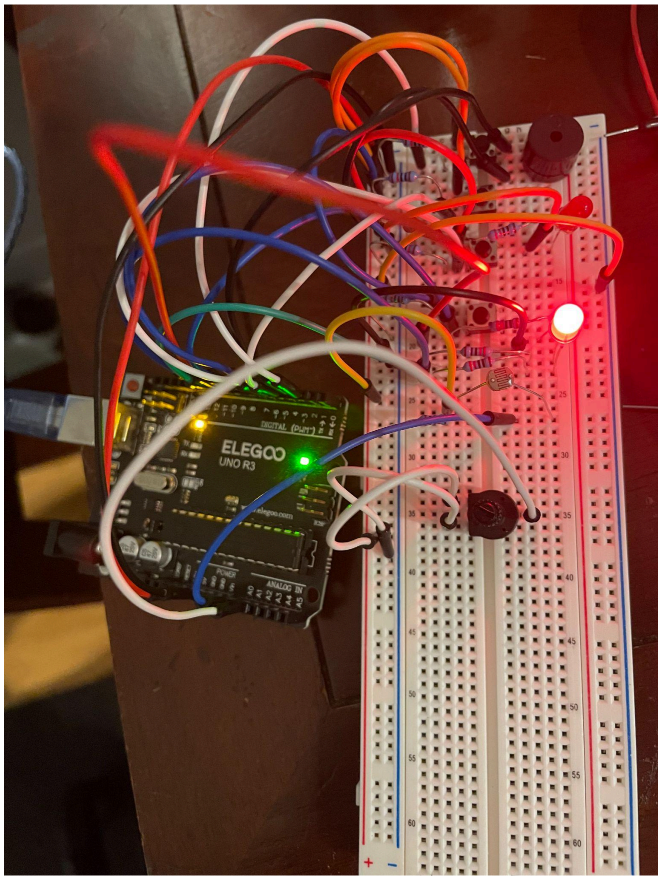
Arduino Final Project

EEL3003 Elements of Electrical Engineering
Ethan Dunton
11/23/24

I. Introduction



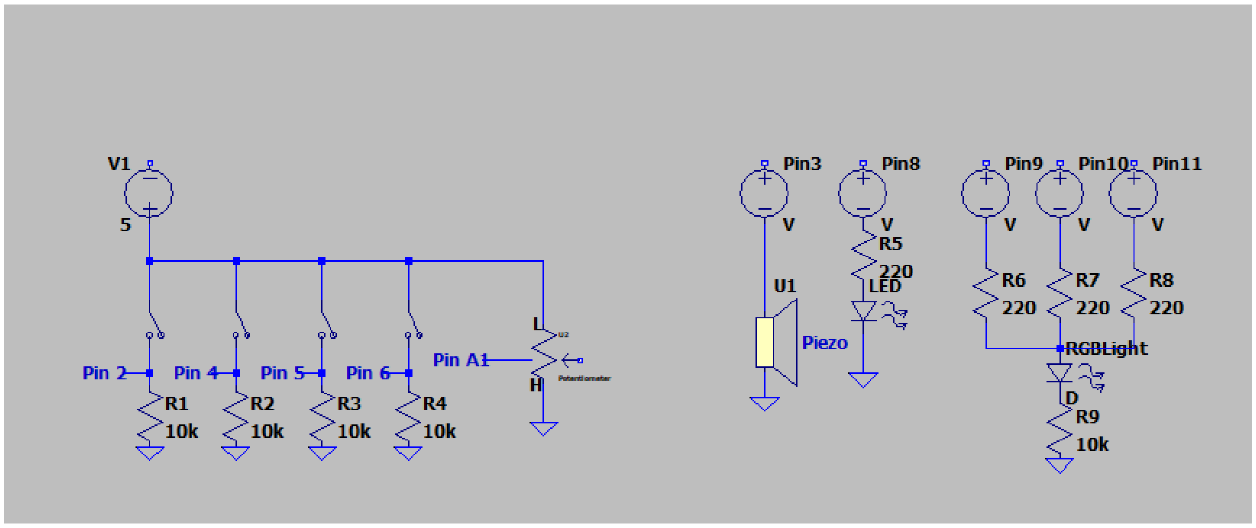
This project is essentially an extremely simple keyboard in that it is an instrument with properties of both a piano and a guitar. Its core function is that there are four buttons, each being assigned to a different note - specifically, D, E, G, and A. When pressed, each button plays that note on the piezo. This uses simple pull down resistors to ensure that the correct values are read at all times, and if the button is pressed, the pin it's connected to reads "High" which, from the code, sends a PWM signal to the piezo to play that note. There is some additional functionality here as well - there is a potentiometer "octave switch" that varies between three ranges. The low range keeps the notes at the original octave, the middle range raises them one octave higher, and the high range raises them two octaves higher. The current octave is indicated by the RGB switch, which is red at the basic octave, green at the middle octave, and blue at the high octave. Lastly, there is an additional "power chord mode" activated when the phototransistor is covered. When this is activated, any notes played will make not only the original note be played by the piezo, but also the note a fifth above. For example, playing an E in power chord mode will make both E and the B above it be played. When these are played together, it's difficult to hear that it's two notes and not just the fifth as even on real instruments, fifths sound very similar to the root note. However, when compared to just the fifth being played at the original frequency (as in, a D in power chord mode playing a D and an A versus just an A in regular mode), the power chord has a thicker sound which is shown in the video. An LED also lights up when in power chord mode.

II. Part List

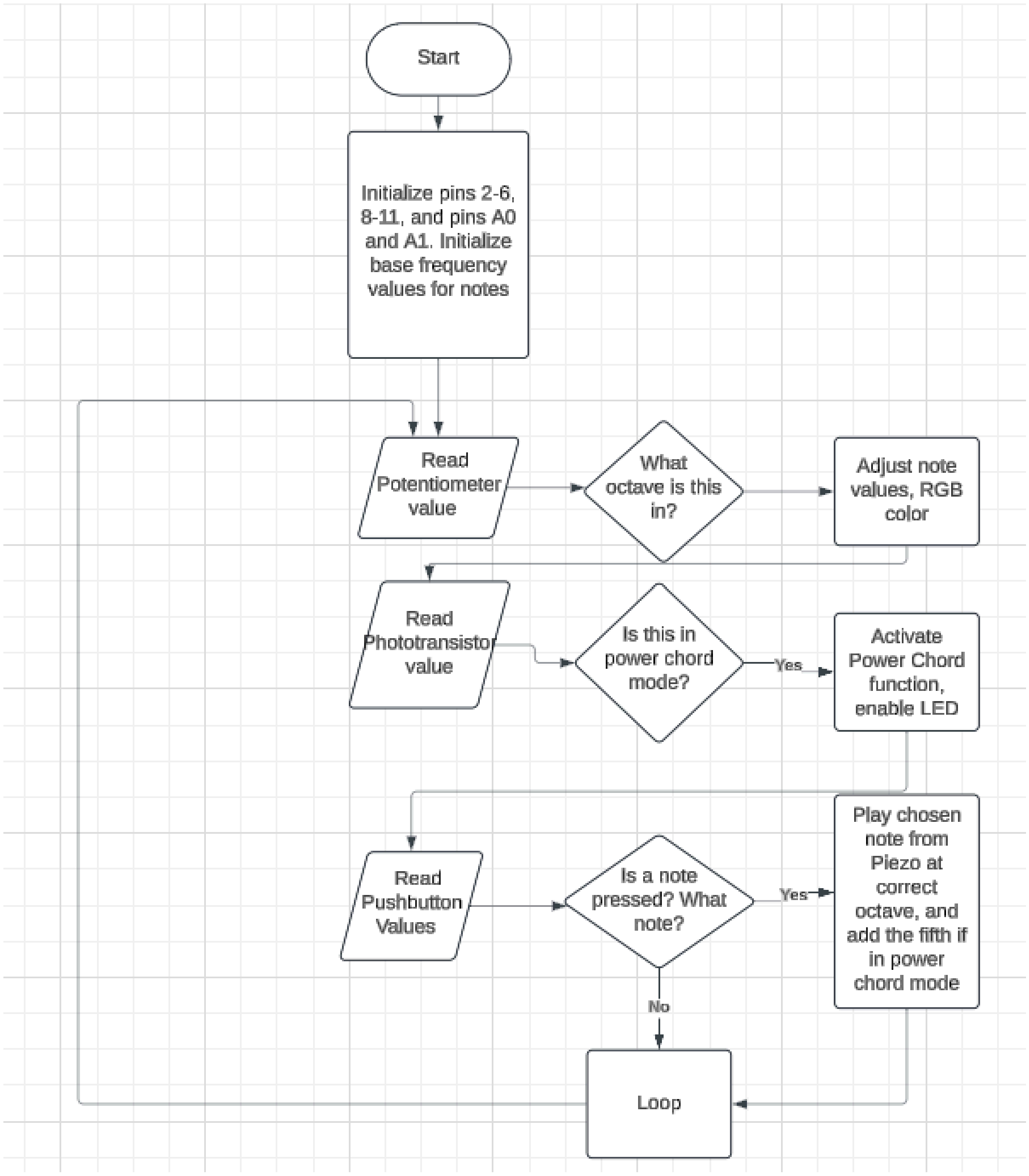
Highlighted parts are component types for project requirement.

Part	Quantity
USB Cable	1
Arduino UNO Controller	1
Breadboard	1
Wire	19
10k Ohm Resistor	5
220 Ohm Resistor	3
Pushbutton	4
RGB Variable Color Switch	1
Potentiometer	1
LED	1
Piezo Speaker	1
Phototransistor	1

III. Schematic



IV. Flowchart


Start
Initialize pins 2-6, 8-11, and pins A0 and A1. Initialize base frequency values for notes
Read Potentiometer value
What octave is this in?
Adjust note values, RGB color
Read Phototransistor value
Is this in power chord mode?
Yes
Activate Power Chord function, enable LED
Read Pushbutton Values
Is a note pressed? What note?
Yes
Play chosen note from Piezo at correct octave, and add the fifth if in power chord mode
No
Loop

V. Final Code

Note: It's likely obvious in this code that ChatGPT was used heavily for help coding. That being said, I walked it through my process step by step and had it help generate code piece by piece, and all the ideas and wiring were original.

```
// Pin assignments
int button1 = 2; // Button for note D
int button2 = 4; // Button for note E
int button3 = 5; // Button for note G
int button4 = 6; // Button for note A
int piezo = 3;    // Piezo buzzer
int potPin = A1; // Potentiometer pin (reversed input)
int photoPin = A0; // Phototransistor pin
int ledPin = 8;   // LED indicator for power chord mode
int redPin = 9;   // RGB red pin
int greenPin = 10; // RGB green pin
int bluePin = 11;  // RGB blue pin

// Base frequencies for notes (Hz)
int baseNoteD = 294;
int baseNoteE = 330;
int baseNoteG = 392;
int baseNoteA = 440;

// Variables for octave adjustment and power chord mode
int octaveOffset = 1; // Default octave is raised by +1
bool powerChordMode = false; // Power chord mode status

void setup() {
  // Set button pins as input
  pinMode(button1, INPUT);
  pinMode(button2, INPUT);
  pinMode(button3, INPUT);
  pinMode(button4, INPUT);

  // Set piezo pin as output
  pinMode(piezo, OUTPUT);

  // Set LED pins as output
  pinMode(ledPin, OUTPUT);
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
}

void loop() {
  // Read the potentiometer value and reverse its direction
  int potValue = analogRead(potPin); // Read value (0-1023)
  potValue = 1023 - potValue;         // Reverse the potentiometer input

  // Determine the octave based on the reversed potentiometer value
  if (potValue < 341) {
    octaveOffset = -1; // Low octave
  } else if (potValue < 682) {
    octaveOffset = 0; // Default octave
  } else {
    octaveOffset = 1; // High octave
  }

  // Check the phototransistor to activate/deactivate power chord mode
  int photoValue = analogRead(photoPin);
  powerChordMode = (photoValue < 500); // Threshold for covered
phototransistor

  // Update LED based on power chord mode
  digitalWrite(ledPin, powerChordMode ? HIGH : LOW);

  // Update RGB LED based on octave
  updateRGB(octaveOffset);

  // Adjust note frequencies based on octave offset
  int noteD = adjustOctave(baseNoteD, octaveOffset + 1); // Default octave
raised by 1
  int noteE = adjustOctave(baseNoteE, octaveOffset + 1);
  int noteG = adjustOctave(baseNoteG, octaveOffset + 1);
  int noteA = adjustOctave(baseNoteA, octaveOffset + 1);

  // Play notes based on button presses
  if (digitalRead(button1) == HIGH) {
    playPowerChord(noteD); // Play D
  } else if (digitalRead(button2) == HIGH) {
    playPowerChord(noteE); // Play E
  } else if (digitalRead(button3) == HIGH) {
    playPowerChord(noteG); // Play G
  } else if (digitalRead(button4) == HIGH) {
    playPowerChord(noteA); // Play A
  } else {
    noTone(piezo); // Stop playing if no button is pressed
  }
}

// Function to play a single note or power chord
void playPowerChord(int baseFrequency) {
  if (powerChordMode) {
    // Power chord: play the base note and its fifth
    int fifthFrequency = baseFrequency * pow(2, 7.0 / 12.0); // Fifth =
base * 2^(7/12)
    tone(piezo, baseFrequency);
    delay(10); // Small delay to overlap notes
    tone(piezo, fifthFrequency);
    delay(300); // Hold the chord
  } else {
    // Single note mode
    tone(piezo, baseFrequency, 300);
    delay(300); // Hold the note
  }
}

// Function to adjust frequency for the selected octave
int adjustOctave(int baseFrequency, int offset) {
  return baseFrequency * pow(2, offset); // Multiply by 2^offset
}

// Function to update RGB LED color based on the octave
void updateRGB(int octaveOffset) {
  if (octaveOffset == -1) {
    // Low octave: Red
    digitalWrite(redPin, HIGH);
    digitalWrite(greenPin, LOW);
    digitalWrite(bluePin, LOW);
  } else if (octaveOffset == 0) {
    // Default octave: Green
    digitalWrite(redPin, LOW);
    digitalWrite(greenPin, HIGH);
    digitalWrite(bluePin, LOW);
  } else if (octaveOffset == 1) {
    // High octave: Blue
    digitalWrite(redPin, LOW);
    digitalWrite(greenPin, LOW);
    digitalWrite(bluePin, HIGH);
  }
}
```

VI. References

“Arduino Piano Code” prompt. *ChatGPT*, 11/23/24, OpenAI, chat.openai.com.