Voltage-to-Frequency Electric Piano

Final Project Report

Course: EE45 – Electronics Systems and Laboratory
Institution: California Institute of Technology
Instructor: Dr. Emami
Submission Date: 06/12/25

Team Members:

Aditya Pillai
Jason Tran
Arjun Pradhan

1. Introduction

This project focuses on building an electronic piano using analog components. The design allows us to play 12 musical notes across a single octave, from A4 (440 Hz) to A5 (880 Hz). The core idea is that each button on the keyboard produces a unique voltage, and that voltage determines the pitch of the sound.

The design assumes that only one button is pressed at a time. This simplifies both the hardware and the logic, and it makes the system easier to test. The range of notes was chosen based on what the circuit can generate reliably, and the values for each component were selected to keep the output stable and within the right frequency range.

2. System Architecture / Block Diagram

The system includes five main parts. First is the set of push-buttons, which acts as the interface for playing notes. Each button is connected to a pin of an arduino micro that is connected to the MCP4725 Digital to Analog Converter breakout board. The Vout of the DAC, regulated through the arduino, goes into a voltage-to-frequency converter, which turns it into a waveform with a corresponding frequency. From there the signal passes through an audio amplifier and then into a speaker, which produces the sound.

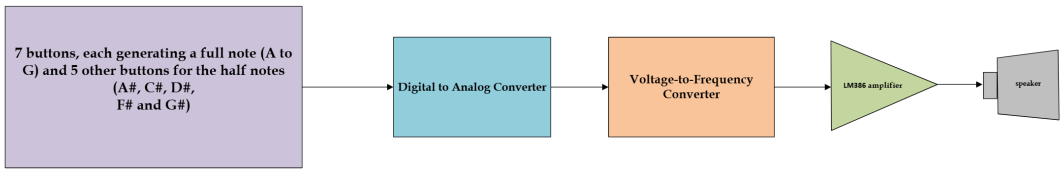

Figure 1: Block diagram of the full system. Each button generates a voltage corresponding to a musical note, which is converted into a frequency, amplified, and played through a speaker.

3. Theoretical Analysis

Key Input Interface and DAC

While our original design at first implemented a simple resistor ladder as the DAC, we pivoted to a more integrated approach by using an Arduino Micro board interfaced with a separate MCP4725 DAC chip. The primary reason we pivoted is because the digital DAC is able to specify our target voltage with extreme precision given the C++ code we wrote in the Arduino sketch. In this scenario, each button is fed into a data pin on the micro, which is then processed to output a constant voltage. Our DAC and its wired configuration is detailed below.

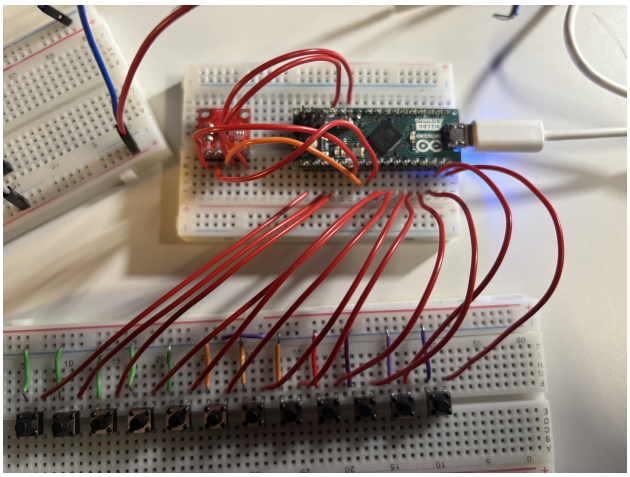

Figure 2: Digital DAC Module

The purpose of this block is to generate a unique DC voltage for each note/button. These DC voltages and their corresponding frequencies and notes are summarized in the table below when testing the actual circuit.

Note	Frequency (Hz)	V_{Out} (V)
A_4	440	0.35
$A\#_4/B\flat_4$	466	0.37
B_4	494	0.39
C_5	523	0.42
$C\#_5/D\flat_5$	554	0.45
D_5	587	0.475
$D\#_5/E\flat_5$	622	0.505
E_5	659	0.535
F_5	698	0.575
$F\#_5/G\flat_5$	740	0.615
G_5	784	0.65
$G\#_5/A\flat_5$	831	0.70

Table 1: DAC design for 5th octave

Voltage-to-Frequency Converter

The voltage-to-frequency converter is the core of the system. It uses a relaxation oscillator built from two op-amps: an integrator and a Schmitt trigger comparator. The integrator charges a capacitor linearly, and the comparator resets it via a diode once a threshold is reached. This produces a sawtooth waveform whose frequency increases with input voltage. Shown below is the schematic of the circuit that we utilized for the design:
To begin with analysis of this circuit, we treat one complete cycle as two linear phases:

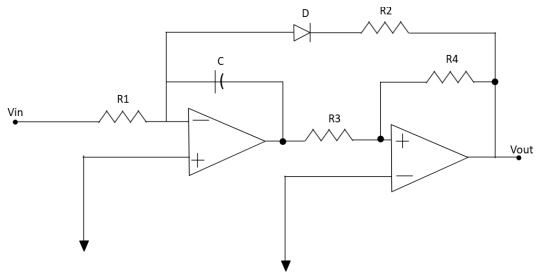

Figure 3: Schematic of Voltage-To-Frequency Converter

Comparator output	Diode state	Integrator slope $\dot{V}_x$
$V_{\text{out}} = +V_{CC}$	OFF	$\dot{V}_x = -\dfrac{V_{\text{in}}}{R_1 C}$
$V_{\text{out}} = -V_{CC}$	ON	$\dot{V}_x = -\dfrac{V_{\text{in}}}{R_1 C} - \dfrac{V_{\text{out}}}{R_2 C} = \dfrac{V_{CC}}{R_2 C} - \dfrac{V_{\text{in}}}{R_1 C}$

The slopes follow directly from KCL at the integrator summing-node (V_x is the integrator output):

$$\frac{V_{\text{in}}}{R_1} + \frac{V_{\text{out}}}{R_2} + C\dot{V}_x = 0 \implies \dot{V}_x = -\frac{V_{\text{in}}}{R_1 C} - \frac{V_{\text{out}}}{R_2 C}.$$

Schmitt-trigger thresholds. With the non-inverting input of the second op-amp tied to the divider R_3–R_4,

$$0 = V_+ = \frac{R_4}{R_3 + R_4} V_x + \frac{R_3}{R_3 + R_4} V_{\text{out}} \implies V_x = \pm \frac{R_3}{R_4} V_{CC} \equiv \pm V_H.$$

Hence the triangular ramp swings between $\pm V_H$ with

$$V_H = \frac{R_3}{R_4} V_{CC}, \qquad \Delta V = 2V_H.$$

Time per phase.

$$t_{\text{OFF}} = \frac{\Delta V}{|\,\dot{V}_x|_{\text{OFF}}} = \frac{2V_H R_1 C}{V_{\text{in}}},$$

$$t_{\text{ON}} = \frac{\Delta V}{\dot{V}_x|_{\text{ON}}} = \frac{2V_H C}{\dfrac{V_{CC}}{R_2} - \dfrac{V_{\text{in}}}{R_1}} = \frac{2V_H C R_1 R_2}{V_{CC} R_1 - V_{\text{in}} R_2}.$$

Period and frequency. Adding the two intervals,

$$T = t_{\text{OFF}} + t_{\text{ON}} = 2V_H C \left[\frac{R_1}{V_{\text{in}}} + \frac{R_1 R_2}{V_{CC} R_1 - V_{\text{in}} R_2} \right] = \frac{2V_H C V_{CC} R_1^2}{V_{\text{in}} \big(V_{CC} R_1 - V_{\text{in}} R_2 \big)}.$$

Substituting V_H and inverting gives the frequency:

$$\boxed{f(V_{\text{in}}) = \frac{1}{T} = \frac{R_4\, V_{\text{in}}}{2R_1 R_3 C V_{CC}} - \frac{R_2 R_4\, V_{\text{in}}^2}{2R_1^2 R_3 C V_{CC}^2}}.$$

Thus, with this equation we were able to design given components for our circuit. With that being said, we picked the following values for each of our components: $V_{CC} = 5V, R_1 = 100k\Omega, R_2 = 50k\Omega, R_3 = 10k\Omega, R_4 = 100k\Omega$, and $C = 10nF$. This produces the following expression for frequency as a function of voltage:

$$f(V_{in}) = 1000V_{in} - 1000V_{in}^2$$

Next, we needed to process this signal a bit by cutting off certain frequencies while also amplifying the signal. To do this, we first implemented a low pass filter, of which the schematic is shown below in figure 4.

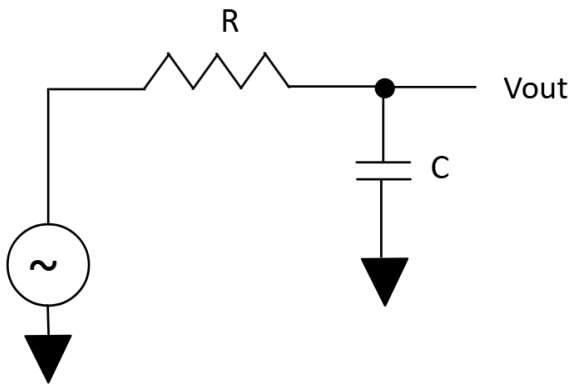

Figure 4: Schematic of Low Pass Filter

The cutoff frequency for the low pass filter is well known, so we won't be including a full derivation. That being said the relationship ends up being $f_{cutoff} = \frac{1}{2\pi RC}$. In this case we are targeting the 5th octave (440 Hz - 831 Hz). As such, we decided to cutoff at a bit above the highest frequency, so we chose $f_{cutoff} = 1$ kHz. Thus, given the above relationship for cutoff frequency in a low pass filter, we designed the components of the filter to be the following values: $R = 10k\Omega$ and $C = 15nF$.

Finally, we wanted to amplify our signal such that we get a loud enough sound to be played. To do this, we implemented a non-inverting amplifier another op-amp. The schematic for this circuit is shown below in figure 5.

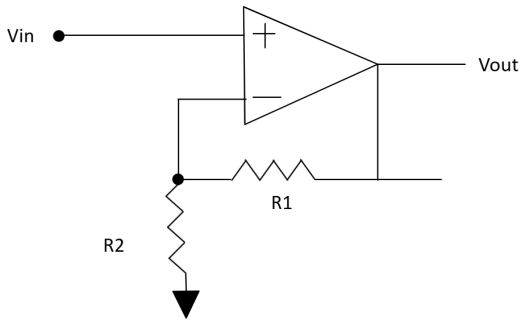

Figure 5: Schematic of Non-Inverting Amplifier

Again, the gain of the non-inverting amplifier is well-known, so we won't be including a full derivation. That being said, the gain abides by the following relationship: $A_v = 1 + \frac{R_1}{R_2}$. We thought a gain of 5 would be a reasonable amplification factor as to make the produced sound just loud enough that it would be pleasant for the ears. As such, we designed our values for each resistor to then be: $R_1 = 3k\Omega$ and $R_2 = 750\Omega$.

The final circuit was then constructed and we arrived at the final schematic with each of our designed values, which is shown below in figure 6.

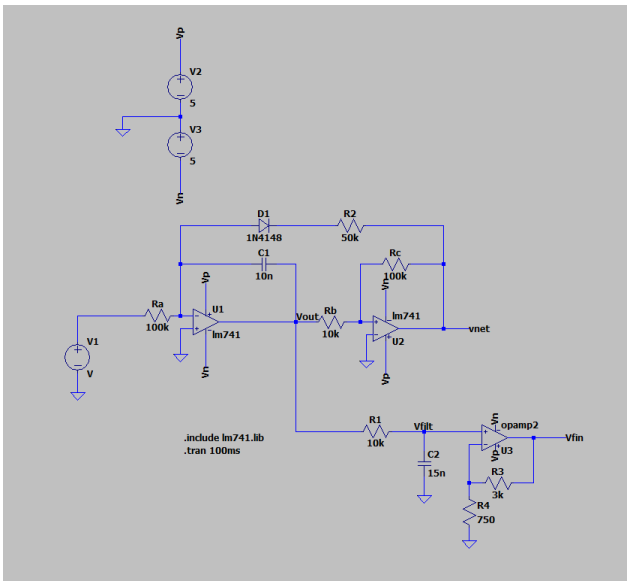

Figure 6: Full Voltage-To-Frequency Converter Schematic with Post Processing

4. Simulations

We tested the circuit using LTspice with input voltages corresponding to three representative notes: A, B, and C. Each of the notes correspond to specific frequency and voltage as summarized in table 1. The waveforms generated in simulation closely matched our expectations. These simulations are displayed below in figures 7, 8 and 9.

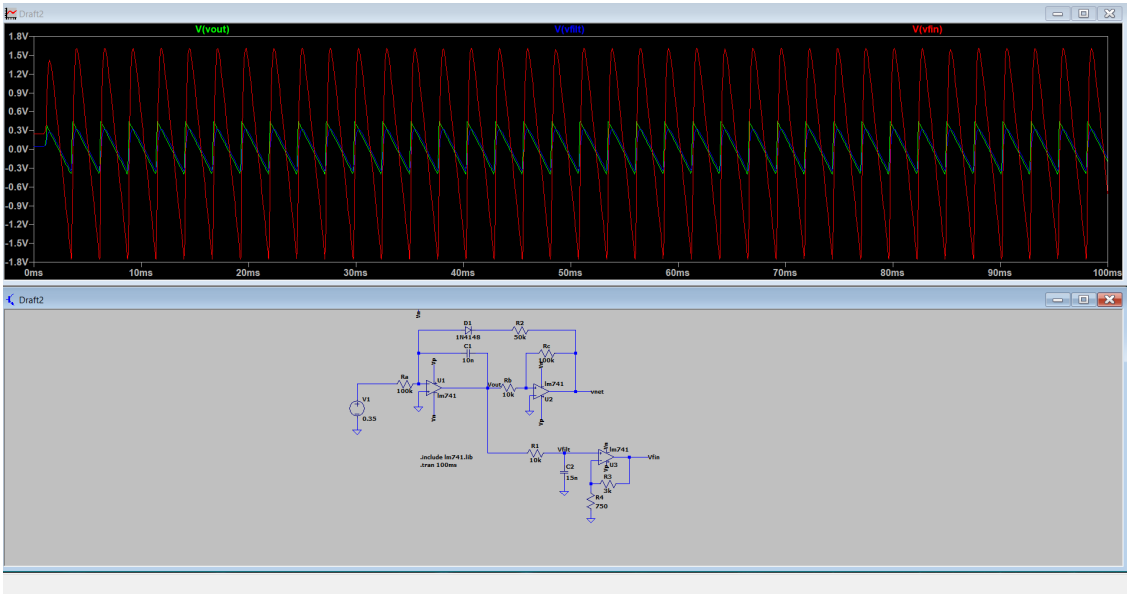

Figure 7: Simulating A note

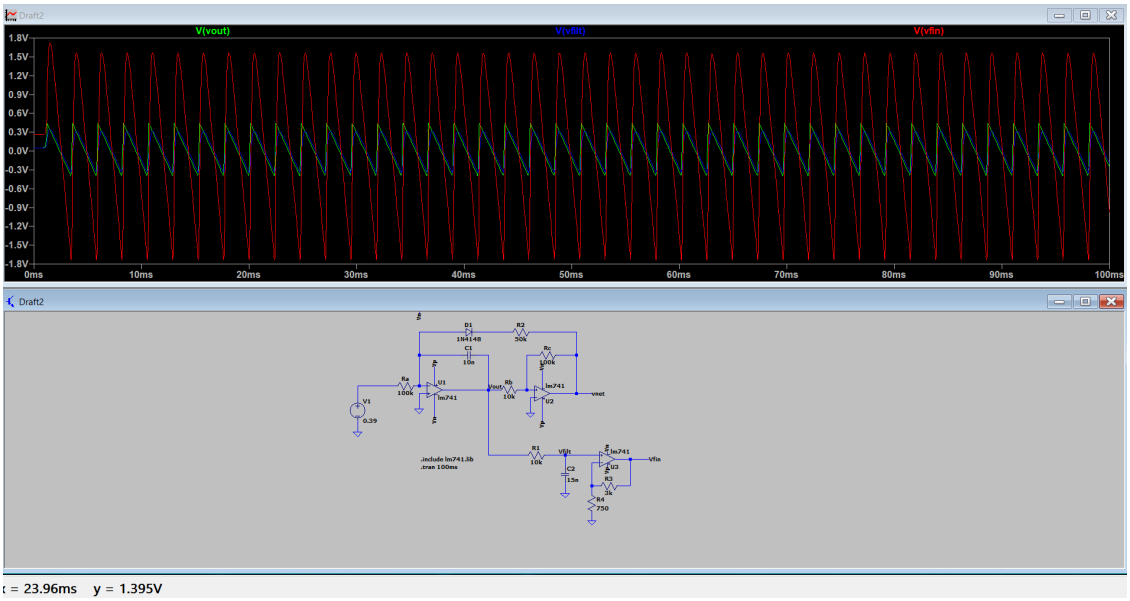

Figure 8: Simulating B note

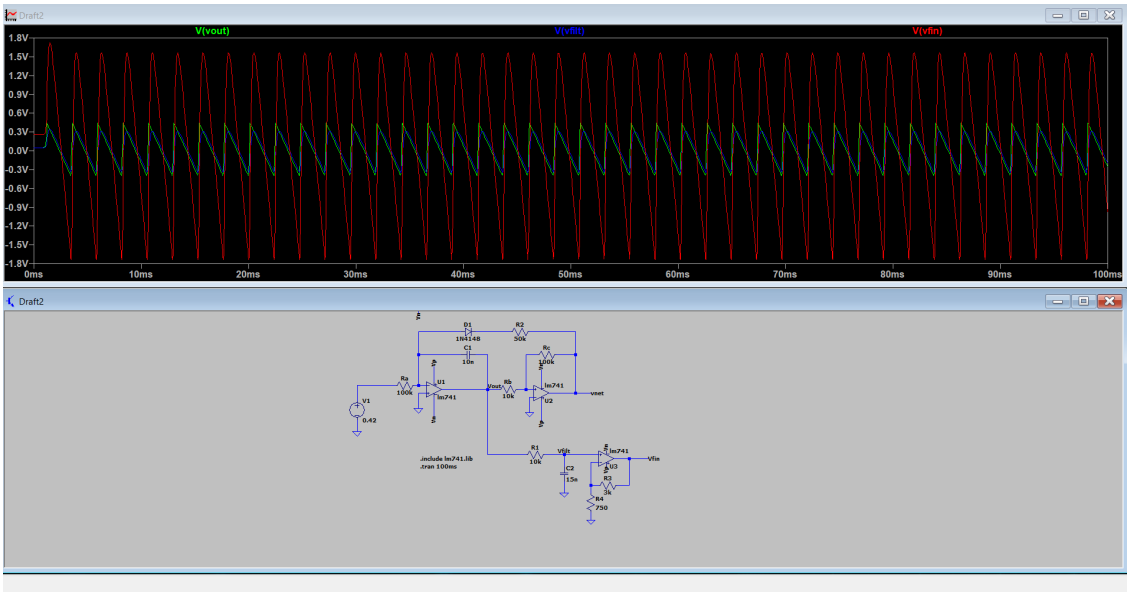

Figure 9: Simulating C note

5. Implementation

Amplifier and Speaker

We used an LM386 audio amplifier to drive the speaker. The gain was configured via a capacitor between pins 1 and 8, and the input/output were AC-coupled to remove DC offset. The entire system was powered by the lab bench supply.

Breadboard Integration

All subsystems were assembled on three breadboards. One containing the buttons, whichs ground and power were directly connected to a half-breadboard that contain the arduino micro and DAC. The VOut of the Dac, was directly connected to Vin of our V2F. Also, after our V2F, we have a low pass filter consisting of another UA741 op-amp and two resistors, 3 kOhm and 750 Ohm. Special attention was paid to grounding and wire layout to minimize signal issues.

6. Results / Plots / Codes

We verified output waveforms on the oscilloscope and confirmed they matched expected frequencies within tolerance.
Here is our Arduino Code used to regulate the voltages of the 12 buttons (keys):

```
#include <Wire.h>
#include <Adafruit_MCP4725.h>

Adafruit_MCP4725 dac;  // Create DAC object
```

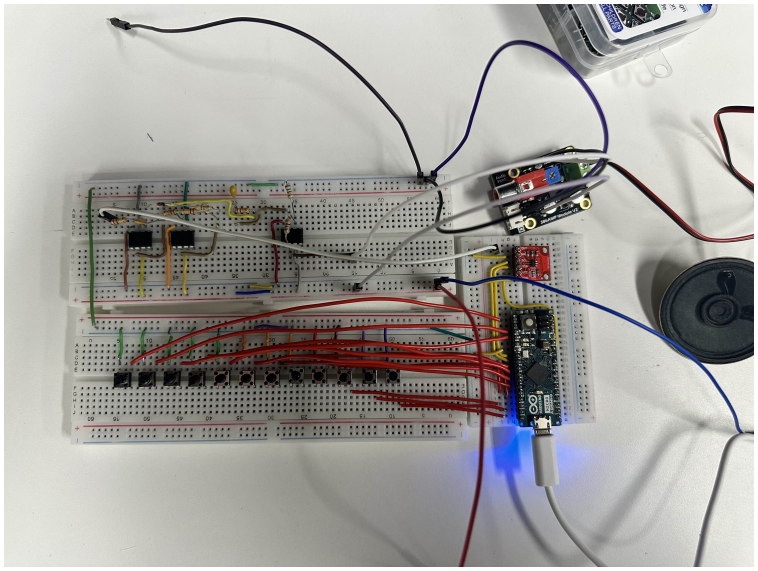

Figure 10: Entire Circuit Overhead View

Figure 11: Oscilloscope capture of the voltage-to-frequency (V2F) converter output. The yellow trace (Channel 1) shows the triangle wave generated by the integrator stage (output of the first UA741 op-amp). The blue trace (Channel 2) displays the corresponding square wave output from the comparator stage (second UA741 op-amp). The measured output frequency is approximately 833 Hz (Testing the waveform using the G Sharp Key), corresponding to the input DAC voltage.

```
5
6  const float vcc = 5.00;  // DAC supply voltage
7
8  const int button1  = 12;
9  const int button2  = 11;
```

```
const int button3  = 10;
const int button4  =  9;
const int button5  =  8;
const int button6  =  7;
const int button7  =  6;
const int button8  =  5;
const int button9  =  4;
const int button10 =  1;
const int button11 =  0;
const int button12 = 16;

void setup() {
  Serial.begin(9600);
  dac.begin(0x60);

  pinMode(button1,  INPUT_PULLUP);
  pinMode(button2,  INPUT_PULLUP);
  pinMode(button3,  INPUT_PULLUP);
  pinMode(button4,  INPUT_PULLUP);
  pinMode(button5,  INPUT_PULLUP);
  pinMode(button6,  INPUT_PULLUP);
  pinMode(button7,  INPUT_PULLUP);
  pinMode(button8,  INPUT_PULLUP);
  pinMode(button9,  INPUT_PULLUP);
  pinMode(button10, INPUT_PULLUP);
  pinMode(button11, INPUT_PULLUP);
  pinMode(button12, INPUT_PULLUP);

  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  if      (digitalRead(button1)  == LOW) outputVoltage(0.35,  1);
  else if (digitalRead(button2)  == LOW) outputVoltage(0.37,  2);
  else if (digitalRead(button3)  == LOW) outputVoltage(0.39,  3);
  else if (digitalRead(button4)  == LOW) outputVoltage(0.42,  4);
  else if (digitalRead(button5)  == LOW) outputVoltage(0.45,  5);
  else if (digitalRead(button6)  == LOW) outputVoltage(0.475, 6);
  else if (digitalRead(button7)  == LOW) outputVoltage(0.505, 7);
  else if (digitalRead(button8)  == LOW) outputVoltage(0.535, 8);
  else if (digitalRead(button9)  == LOW) outputVoltage(0.575, 9);
  else if (digitalRead(button11) == LOW) outputVoltage(0.615, 11);
  else if (digitalRead(button10) == LOW) outputVoltage(0.650, 10);
  else if (digitalRead(button12) == LOW) outputVoltage(0.700, 12);
  else {
    dac.setVoltage(0, false);
    digitalWrite(LED_BUILTIN, LOW);
```

```
  }
  delay(10);
}

void outputVoltage(float vout, int buttonNum) {
  uint16_t code = (vout / vcc) * 4095.0 + 0.5; // Conversion
  dac.setVoltage(code, false);
  digitalWrite(LED_BUILTIN, HIGH);
  Serial.print("Button ");
  Serial.print(buttonNum);
  Serial.print(" pressed  Output voltage: ");
  Serial.print(vout, 3);
  Serial.println(" V");
}
```

Listing 1: Full Arduino sketch that interfaces an MCP4725 digital-to-analog converter (via the Adafruit MCP4725 library) and produces discrete output voltages selected by 12 input buttons.

7. Conclusion

The final circuit successfully produced 12 playable notes using analog components. The DAC generated stable voltages, the voltage-to-frequency converter output frequencies in the correct range, and the amplifier delivered clear signals to the speaker. All modules worked together with minimal integration issues.

Most of the effort was focused on tuning the voltage-to-frequency converter. Once its behavior was understood, the rest of the system came together quickly. The project met its functional goals and offered a solid learning experience in analog design.