Lab #5: The Arduino music maker

Purpose

In this lab you will build an Arduino-controlled music system. The synthesizer will be a four note electronic synthesizer (one button for each note) with distortion control (potentiometer), and a percussion instrument motor (or other instrument of your design) using the servo-motor.

What you will need

1. Arduino board (and USB power cable)
2. Breadboard for building external circuits
3. Circuit components: jumpers, 5 buttons, speaker, resistors, potentiometer, servo-motor
4. Other tools and components: small screwdriver for attaching arm to the servo-motor armature or tape or wire-ties, carboard for making a motor mount

Note: Do NOT use digital pins 0 or 1 on the Arduino. These are special pins that are reserved for communication and using them may cause your board to malfunction in a very ambiguous way that is hard to diagnose. You should be able to use any of the other pins without problem.

Additional resources

https://pages.mtu.edu/~suits/notefreqs.html
https://www.choose-piano-lessons.com/kids-songs.html

Main tasks

1. Lay out the components on the board, being sure to leave space for five buttons, the potentiometers, and the speaker. There will not be a lot of extra space, but it all fits tightly on the standard breadboard if you are very conservative with your space.
2. Add appropriate resistance for each of the circuit elements. Unlike a previous lab where we used the potentiometer in series with a fixed resistance, today we will be using the potentiometer by itself. The total value of our potentiometers is fixed at 10kΩ (between the two legs on the same side), and the leg on the other side represents the intermediate point between two resistors in series.
3. Connect the breadboard to the Arduino so that you monitor the status of the five buttons.
4. Develop your code to create four distinct tones and allow these to be distorted using the potentiometer control.
5. Connect the servo-motor to the board: red = +5V, black = ground, white = signal (from a digital pin).
6. Develop the final bit of code that allows the servo-motor to be controlled as part of your music machine.
7. Build a structure to mount your motor (this can be done using cardboard) and make some kind of instrument. A simple percussion instrument might be the easiest.

Tips

1. To get all of these components to fit on the board using a design where the components straddle the center as we have been doing we need to pack the components together fairly tightly. Each button takes up 3 rows itself. Leave one row between each button and the next. You will have to bend the legs of the series resistor to go between the button and this open row. You can even pack things closer by using the middle row between the

two legs of each button, but then things tend to be so close that it becomes hard to press individual buttons, so some small amount of space is helpful here.

2. Any value of resistance will be fine for these circuits, but use something small like a 220 Ω resistor for the speaker (larger resistances will reduce the volume). You can (and should) use larger values for the button circuits. The potentiometer does not need an additional circuit.
3. To calculate the timing for the tones you can start by using any old values for the delay times (you will want to use the microsecond delay function given that auditory tones are at fairly high frequency and we need better than millisecond resolution here). After you are done testing you should make some intentional tones. Pick a four-note song or tune that you want to play and program these buttons to play those tones. You can use the link above to access a list for conversion of musical notes to frequencies.
4. The servo-motor is a special motor that has limits to its motion. For our motors, these limits are defined as 0 degrees and 180 degrees.
    a. This unit has three wires: red is +5V, black is ground, and white is the signal from the board that determines the angular position of the motor.
    b. To communicate with the board you need to load the a communication and control library and establish communications with it using the latch command.
        i. Load this library with: **#include <Servo.h>**
        ii. Create a servo-motor object (like a variable) with: **Servo George** (or whatever you want to call your servo motor).
        iii. Establish communications with your motor in the setup() function by including this command: **Servo.latch(pin)** where **pin** is the pin to which the white wire is connected on the Arduino board.
        iv. Tell the motor to move using: **George.write(position)** where **position** is the angular position of the motor (in degrees).
        v. The motor does not adjust its position instantaneously. Between each move command you will need to include a little bit of a delay so that you give the motor time to complete its move action before telling it to move again. A delay of a few hundred microseconds seems to be fine for small motions of about 10 degrees or so.
5. You will need to attach some kind of arm to the motor armature. You should be able to make this work with tape or wire ties, but the small screws that come with the armatures are probably best. **These screws are very small and can easily be lost, so be sure to work over a safe place where you have something to catch these screws.** If using the screws, you will have the best success in attaching the popsicle stick (or other such thing) to the armature if you make two through-holes in the wood so that the screws thread into the plastic of the armature. If you do this be sure to make a hole that is just about the same size as the threads, but smaller than the head of the screw (so that it doesn't fall into the hole). To mark these holes on the wood you can hold the plastic armature down on the wood and make a small mark on the wood surface using a paperclip or other small and pointed object.
6. Your motorized instrument need not be a percussion instrument. It could, for example, be some kind of string instrument that is plucked by the motor. Please feel free to experiment and come up with something clever.

Lab report
Your lab report should include all of the normal elements of a full report, which includes and introduction, a setup and description of the apparatus, an analysis and calculations section, and a conclusion. In addition to your lab report, submit your Arduino code file and a short video recording of your device in action.

Introduction
State what you are making and the general goals of this work.

Setup
Please include the following elements in your report:

1. Circuit diagrams for each of your circuits. Remember to label your input and output pins as you have designed in your circuit.

2. Include a sketch of your motorized instrument design and define the range of motion (in degrees) needed to activate this instrument. Describe the materials used for the construction of this device and the approximate setup that enables this.

Data and analysis
There is not a lot of data to analyze for this project. Please show the following things.

1. Given your schematic for button #1, show through a calculation whether your measurement should transition from 0 to 1 or from 1 to 0 when the button is pressed.

2. Make a calculation of the voltage measured by the Arduino at the potentiometer as a function of the knob's position, knowing that the total resistance of the potentiometer is fixed at 10 kΩ.

3. State what notes your synthesizer plays when the buttons are pressed and show how these notes translate to the specific time delays used in your code.

4. Choose a distortion range other than +/-128 or +/-150 (which were presented as calculations in class). Define the transformation equation that allows you to create your distortion range using integer math.

Conclusions
Please reflect on this project overall, and comment on the following two things in particular.

1. Explain why the notes that we get out of our synthesizer when two buttons are pressed is not a smooth combination of tones that we would get if we were playing a real-life musical instrument or on a professional synthesizer (I comment on this toward the end of the video from session #2).

2. If you were to continue to expand this project and work toward making a more complicated or higher quality music maker, what three things would you do to improve it?