Project 4: iPoduino

Due date: December 2nd at 11:59 PM
Link to Slides: Lecture 4: Microcontrollers

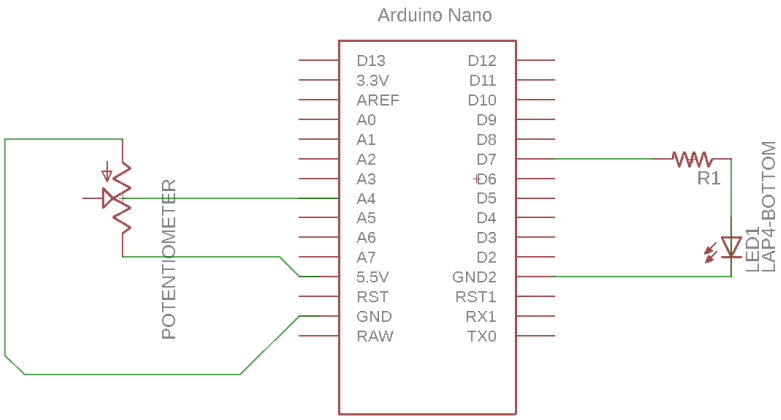
1 | About the Project

This is an introduction to microcontrollers and the Arduino IDE. Practice using Arduino syntax, as well as uploading and testing your code, when programming your Arduino Nano to control the blinking rate of an LED. Then learn about PWM (Pulse Width Modulation) and manually create PWM with varying duty cycles. Finally, combine what you have learned to build an Arduino music player, an iPoduino.

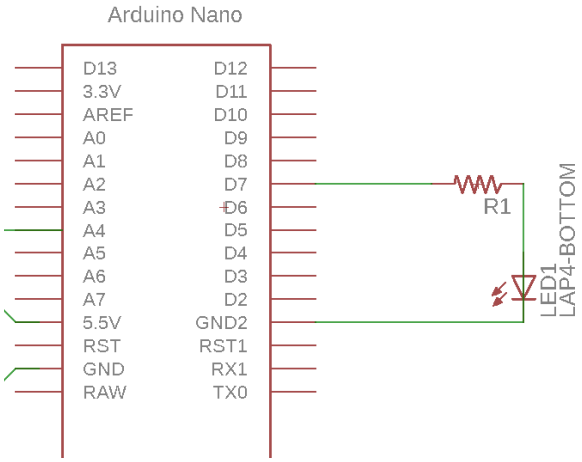
2 | Parts Needed

x1	Breadboard
x1	Potentiometer
x1	Speaker
x1	LED
x1	130 Ohm Resistor
x1	Arduino Nano
x1	Mini USB Cable

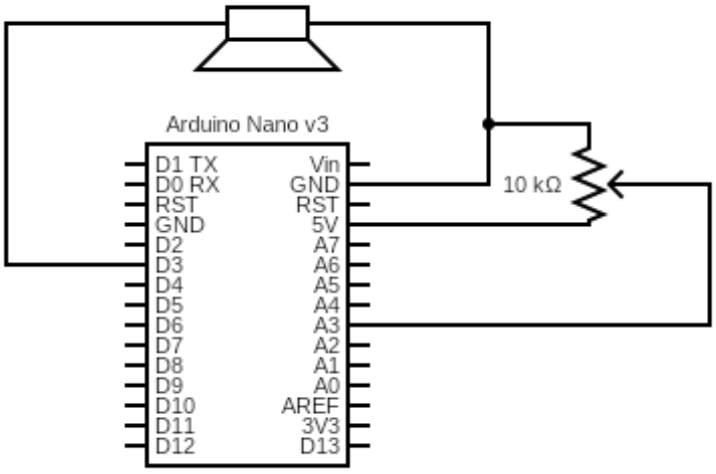
3 | Schematics



Checkpoint 1: Potentiometer and LED Schematic



Checkpoint 2: LED Schematic



Project 4: Speaker Schematic

4 | How to Approach

1. Complete checkpoint 1. Focus on familiarizing yourself with Arduino syntax and uploading/testing your code on the Arduino.
2. Complete checkpoint 2.
3. Build the circuit for the iPoduino on your breadboard.
4. Program a song to play through the speaker
    a. Some helpful functions include:
        i. `tone(pin, frequency)`: generates a 50% duty cycle PWM square wave at a specified frequency.
        ii. `noTone(pin)`: stops PWM wave generated by tone().
        iii. `play(note, duration)`: helper function that combines tone() and noTone() so that you don't need to do it ;).

1.
```
#define SPEAKER [INSERT PIN HERE]
int NOTE_DUR = 137; // This value can be modified
                    // to change the tempo

void play(int note, int dur) {
    tone(SPEAKER, note);
    delay(dur * NOTE_DUR);
    noTone(SPEAKER);
    delay(dur * NOTE_DUR / 3);
}
```

   b. Be sure to include header files `pitches.h` or `basicpitches.h` in the same folder as your sketch and at the top of your code.
   c. Each song should be a minimum of 10 beats (notes).
5. Program the potentiometer interface
   a. Switch between songs by reading the voltage from the potentiometer **after** the reset button is pressed.
   b. Be able to switch between at least 3 songs.
   c. For organizational purposes, it may be helpful to create a separate function for each song.
   d. Hint: setup( ) happens once per reset, loop( ) repeats itself infinitely
6. Program the display
   a. Code a visual display that will indicate which song is playing.
   b. There are different ways you can use the LED to show that a different song is playing, such as changing the brightness of a single LED, using three different color LEDs, etc.

5 | Checkpoint 1

Build the potentiometer and LED circuit. See the schematic for the LED circuit above. Program the Arduino Nano to control the rate that the LED blinks, using a potentiometer.

Pseudocode:

```
// define input pin for potentiometer
// define pin for LED

void setup() {
  // declara pins (OUTPUT or INPUT?)

}

void loop() {
  // Get value from potentiometer
  // Turn LED on
  // Make it stay on depending on the value read from potentiometer
  // Turn LED off
  // Make it stay off depending on the value read from potentiometer
}
```

6 | Checkpoint 2

Manually create PWM. Write 2 different duty cycles to change the brightness of an LED. (Show us 2 brightness for your LED, you aren't allowed to use 0% duty cycle NOR 100% duty cycle). Make sure that the pulse frequency is 1kHz (one cycle lasts 1000 microseconds).

You are only allowed to use in `loop()`:

- `digitalWrite(pin#)`
- `delayMicroseconds(#us)`

7 | Deliverables

Complete both checkpoints and the iPoduino.

- Upload sketches (code) for both checkpoints and iPoduino.
- Video of both checkpoints and iPoduino
    - Video of LED changing brightness with potentiometer dial
    - Video/photo of two different LED brightness
    - Video of ipoduino