

Digital Audio with Teensy - workshop

1. Bits

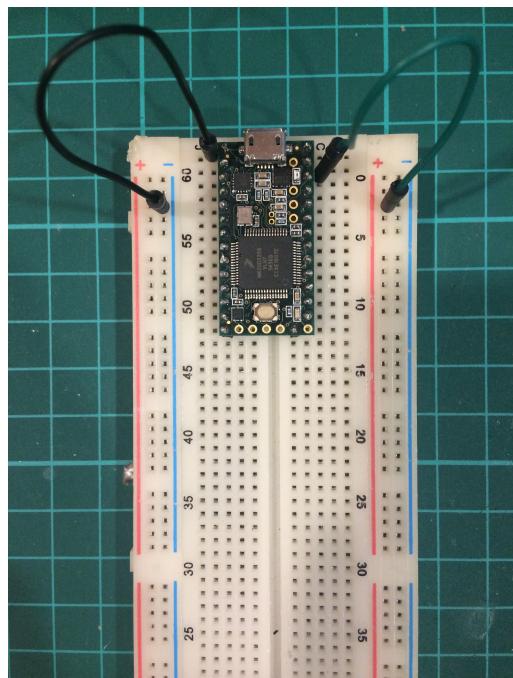
- Teensy board + spec sheet
- Breadboard
- Breadboard jumper cables
- USB cable
- Audio cable
- Switch
- 2 potentiometers

2. Installation

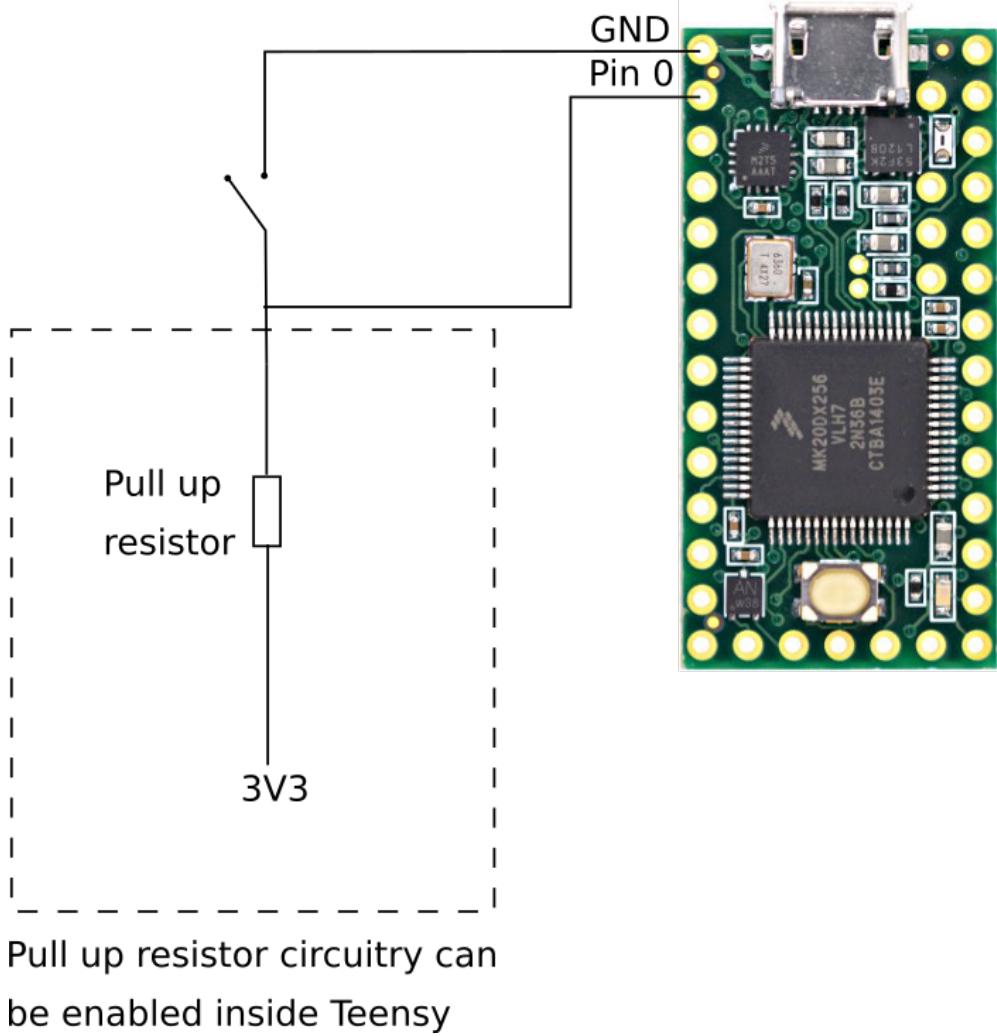
- Install Arduino <https://www.arduino.cc/en/Main/Software>
- Install Teensyduino https://www.pjrc.com/teensy/td_download.html
- Navigate to <https://github.com/cutlasses/TeensyWorkshop> click ‘Clone or Download’ and select Download ZIP
- Unzip the GitHub download to its own folder, keeping the same folder structure.

3. Add the Teensy to the breadboard

- Mount the Teensy as shown
- Connect a patch cable from the GND pin, to the ground rail (blue line)
- Connect the 3.3V (3V3) out to power rail (red line)



4. Connect button as shown



5. Download test code

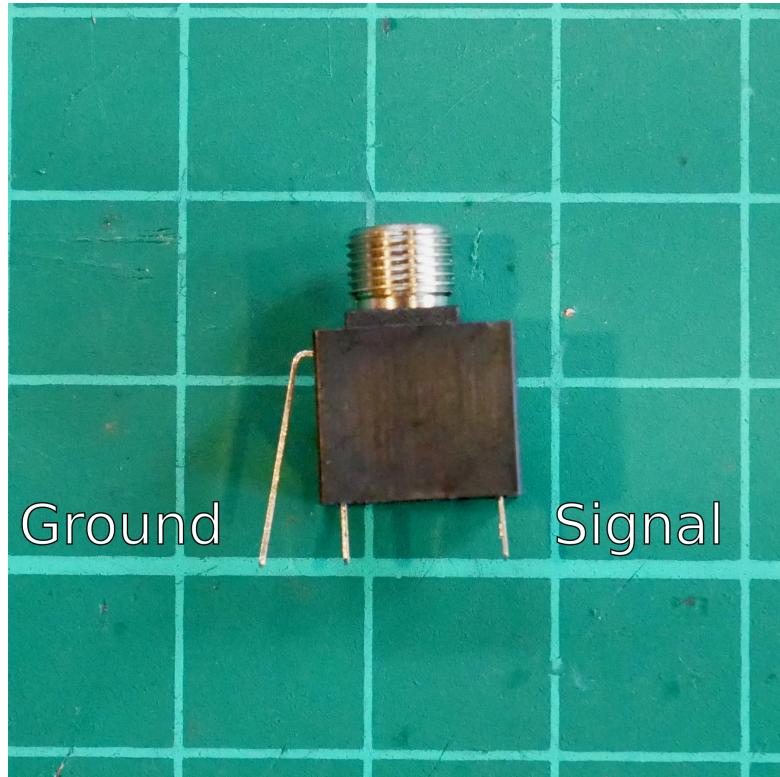
- Open workshop_part1 code in Arduino
- Plug USB cable into Arduino
- Compile and Download to Teensy by clicking the right facing arrow (second button from the left)



- In Tools->Board ensure Teensy 3.1/3.2 is selected
- In Arduino select Tools->Port and select Teensy
- Open Tools->Serial Monitor
- Press button, you should see text appearing in the Serial Monitor

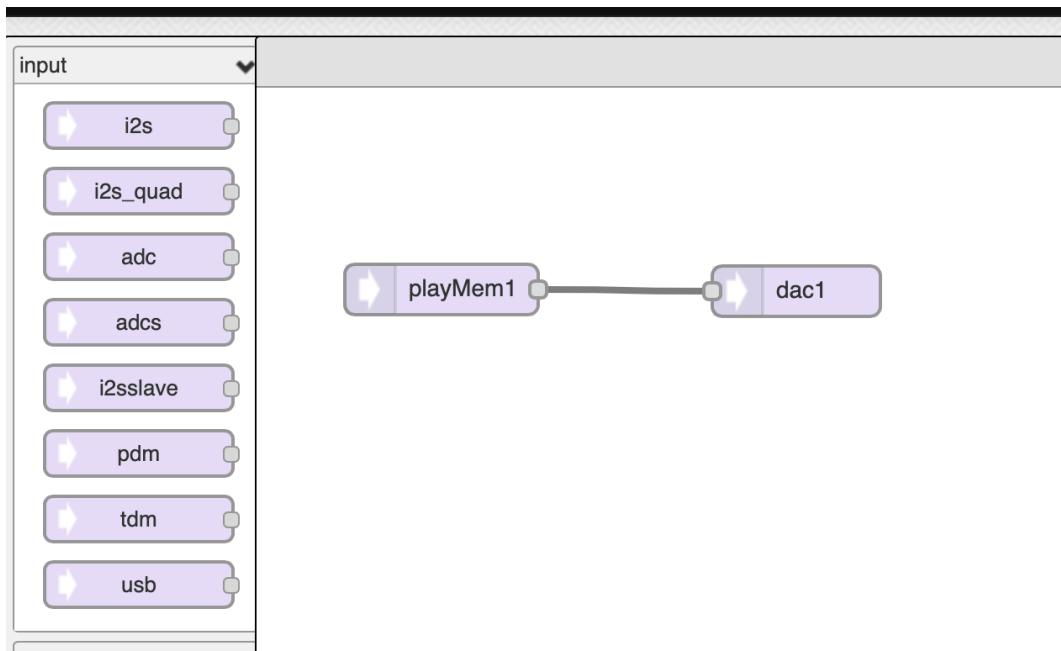
6. Add audio components to breadboard

- Insert jack socket into breadboard
- Connect Signal to DAC pin, and Ground to ground rail



7. Use Audio Design Tool to add audio out

- Connect to <https://www.pjrc.com/teensy/gui/index.html>
- Add the components as shown



- Click Export
- Copy into Arduino Sketch (under #include <bounce.h>)

8. Add code to trigger sample

At the top of the file add

```
#include "AudioSamplePiano_c3_44k.h" // include the sample data for the piano sound
```

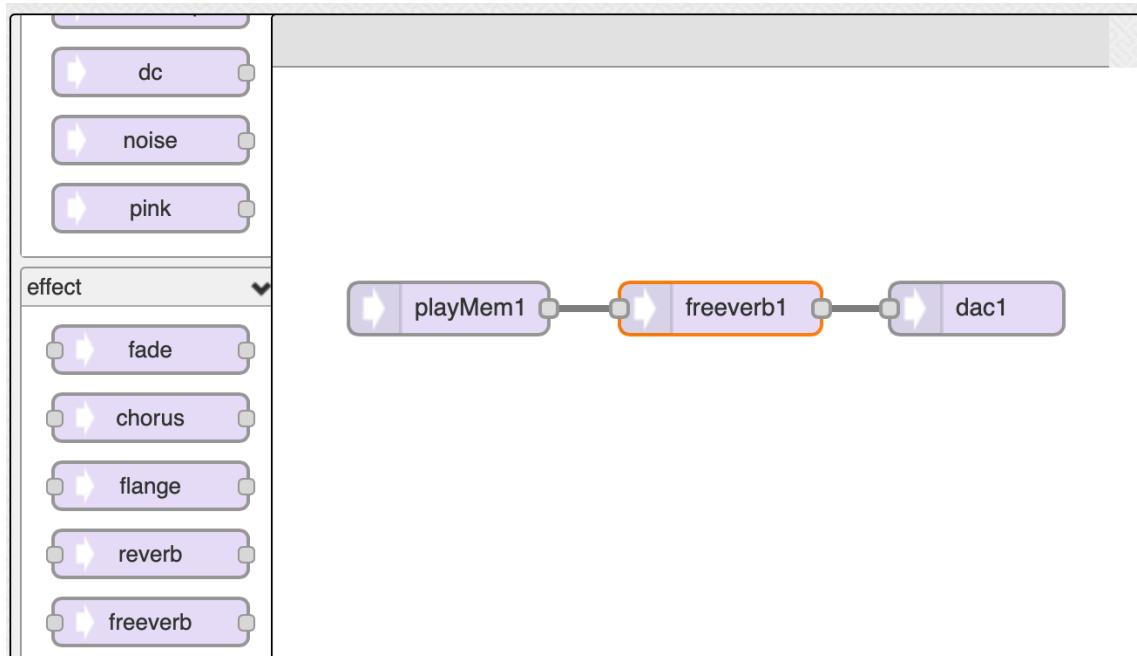
In the button section add

```
// is the button pressed down
if( button_bounce.fallingEdge() )
{
    Serial.println("Button down!");

    playMem1.play( AudioSamplePiano_c3_44k );
}
```

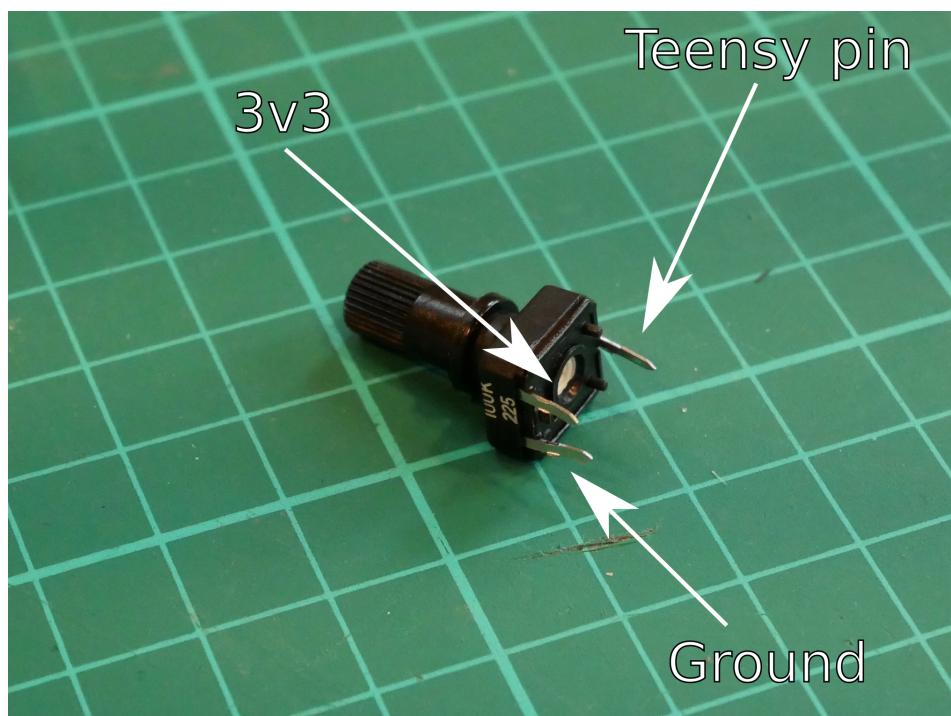
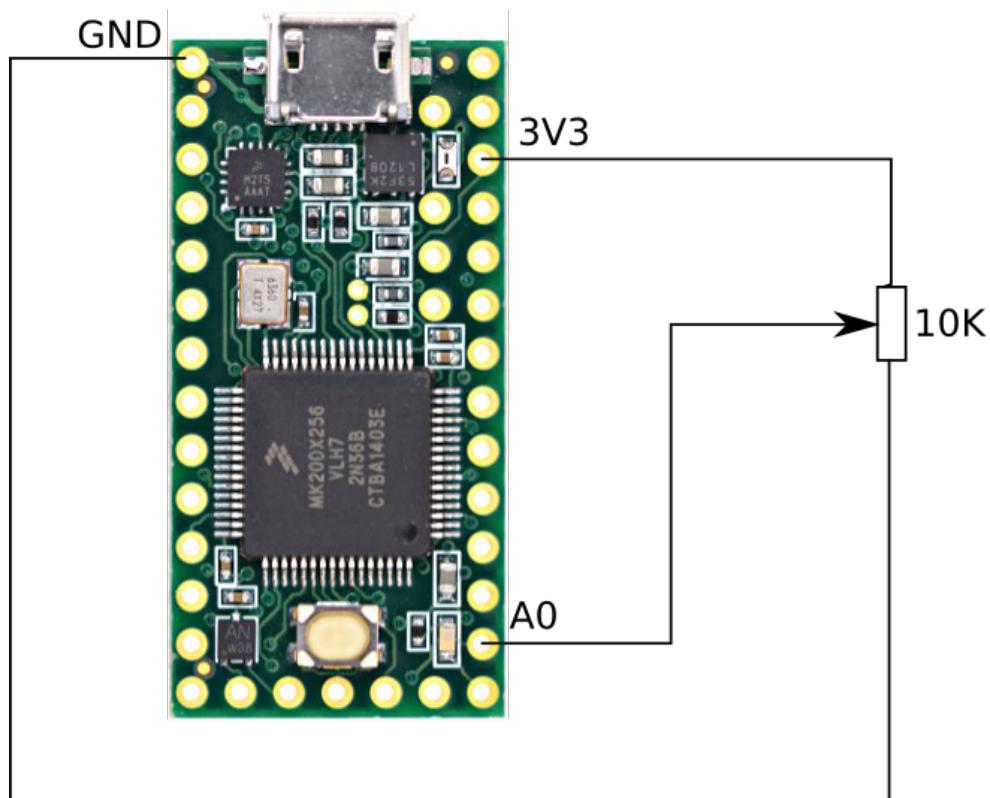
9. Add an audio effect

- In the Audio Design Tool add an effect in between playMem1 (which creates the sound) and dac1 (which outputs the sound)
- Export the code and copy over the previous export



10.Add potentiometers

- Connect as shown below to input A0. Your potentiometer will be configured as a potential divider. A0 will read a variable voltage from around 0V to around 3V3, dependent on how far you've turned the potentiometer
- Connect the second potentiometer to A1 (check your card). The other pins should go to the ground and power rails, as above.



11. Link potentiometers to effects chain

- Add the following code after the BUTTON_PIN definition (near the top of the file)

```
const int POT1_PIN(A0);  
const int POT2_PIN(A1);
```

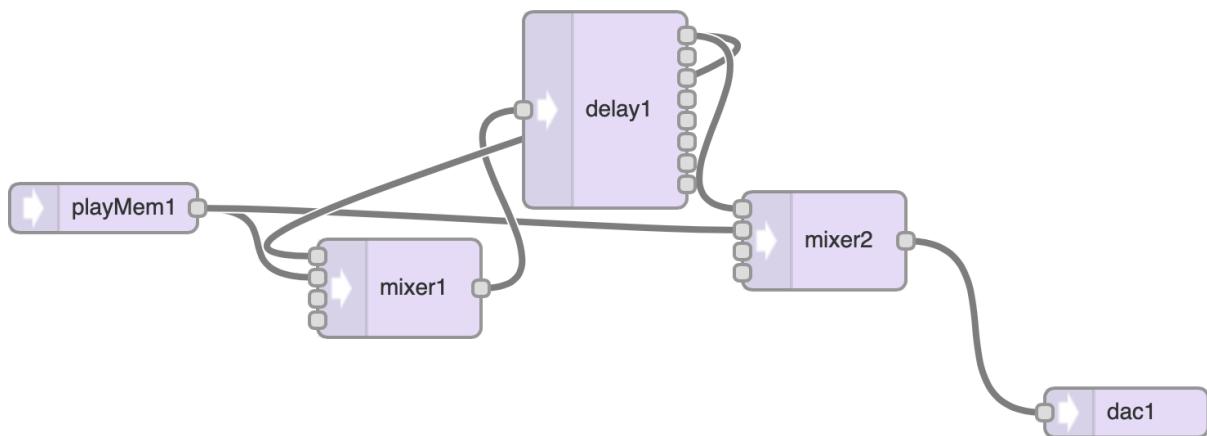
- Add the following code at the end of the update() function

```
float pot1_val = analogRead( POT1_PIN ) / 1024.0f;  
freeverb1.roomsize( pot1_val );  
  
float pot2_val = analogRead( POT2_PIN ) / 1024.0f;  
freeverb1.damping( pot2_val );
```

12. Experiment

- Create a new effects chain using the Audio Design Tool
- Map the potentiometers to control the effects as above (use the documentation in the design tool for help)

Example - Delay with Feedback



- Increase the audio memory to 128

```
AudioMemory(128);
```

- Add this to the setup()

```
// mix dry and wet signal (50% of each)  
mixer2.gain( 0, 0.5f ); // wet  
mixer2.gain( 1, 0.5f ); // dry
```

- Add this to loop when reading potentiometers

```

const int min_delay_time = 50;
const int max_delay_time = 300;

static float old_pot1_val_raw = 0;
float pot1_val_raw = analogRead( POT1_PIN );

// is the difference large enough to change the delay time?
if( abs( old_pot1_val_raw - pot1_val_raw ) > 50 )
{
    const float pot1_val = pot1_val_raw / 1024.0f;
    const float delay_time = min_delay_time + ( max_delay_time * pot1_val );
    delay1.delay( 0, delay_time );

    // update the old value
    old_pot1_val_raw = pot1_val_raw;
}

float pot2_val = analogRead( POT2_PIN ) / 1024.0f;
mixer1.gain( 0, pot2_val ); // feedback
mixer1.gain( 1, 1.0f ); // source audio

```

Bonus exercises

- **Add some LEDs**

- Adding a resistor in series connect an LED to a digital pin

- **Convert a different sample and add to code**

- Download and install wav2sketch
- Convert a wav file to a header file
- Import into your Arduino sketch

- **Make a drum machine!**

- In Arduino select File->Examples->Audio->SamplePlayer
- Hook up 6 buttons to pins 0-5