The M.I.D.I. [workshop]

KABK HackLab IST

https://github.com/RobBothof/midiworkshop

the **M.I.D.I.** [workshop]

PLANNING

Oktober 11

- Midi fundamentals
- Some of my custom (hacked) midi controllers, music instruments, installations
- Building a basic midi controller: (1 button - 1 fader/knob)
  - Controller overview
  - Building hardware / Soldering
  - Programming the midi device
  - Connecting to the ‘puredata’ software and making some noise

Oktober 12 - 24

- Designing your own controller (focus on hardware / software / form / usability ?)

Oktober 25 2023 (the M.I.D.I. workshop continues)

- Production / building / troubleshooting
- Concert / Jam-session / Playing with our awesome instruments!

MIDI Fundamentals

(Musical Instrument Digital Interface)

A Digital Protocol and Technical Standard to communicate between ‘audio related’ devices and/or software programs. Developed around 1981/1982 through a collaborative effort of synthesizer manufacturers.

MIDI can not be used to send audio!

It consists only of control instructions.

Fundamentals

the **M.I.D.I.** [workshop]

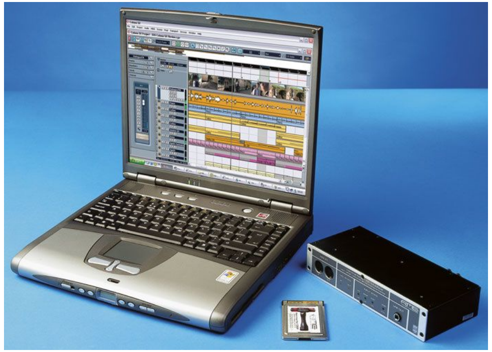
Fundamentals

MIDI Controller

MIDI
(instructions)

Sends what notes to play.

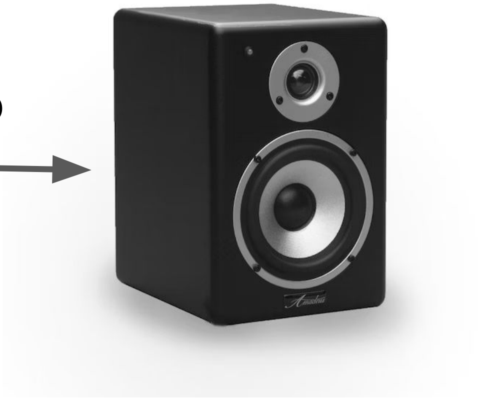
Computer



Generates audio with software based on received Midi notes from controller.

AUDIO

Speaker / Amplifier



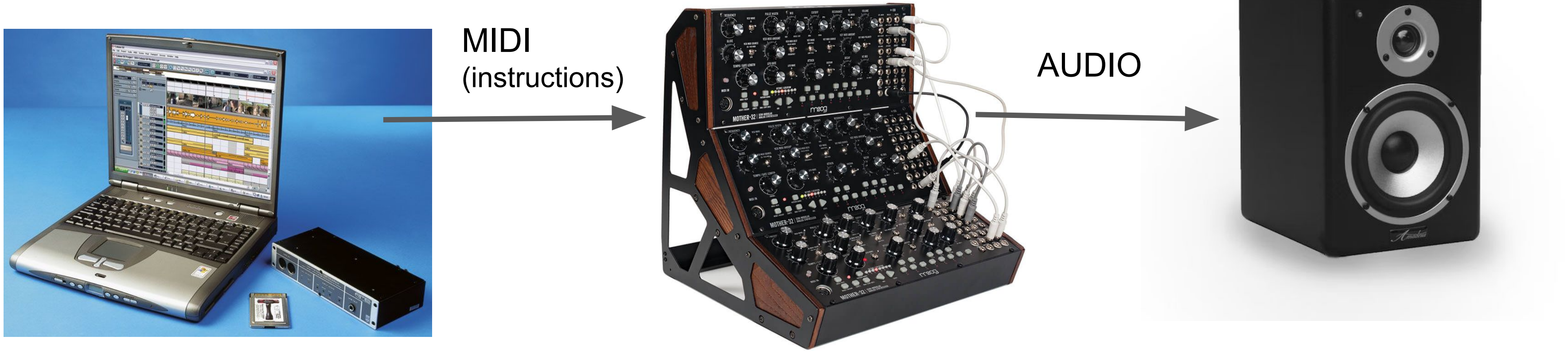
Creates audible sound from audio signal

Fundamentals

Computer
(functioning as midi controller)

Music Instrument
(e.g. synthesizer)

Speaker / Amplifier



MIDI
(instructions)

AUDIO

Sends what notes to play and what tonal setting to change.

Generates audio with hardware based on received Midi notes from computer.

Creates audible sound from audio signal

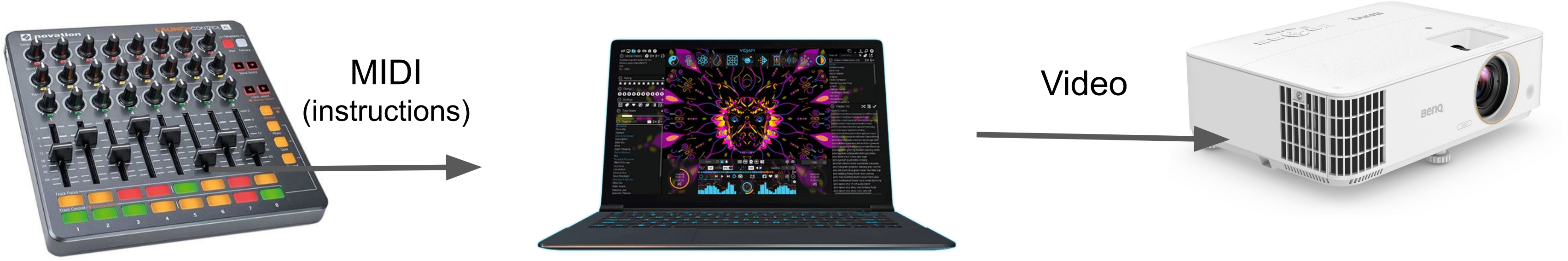
the **M.I.D.I.** [workshop]

Fundamentals

MIDI Controller

Computer
(running VJ software)

Projector



MIDI
(instructions)

Video

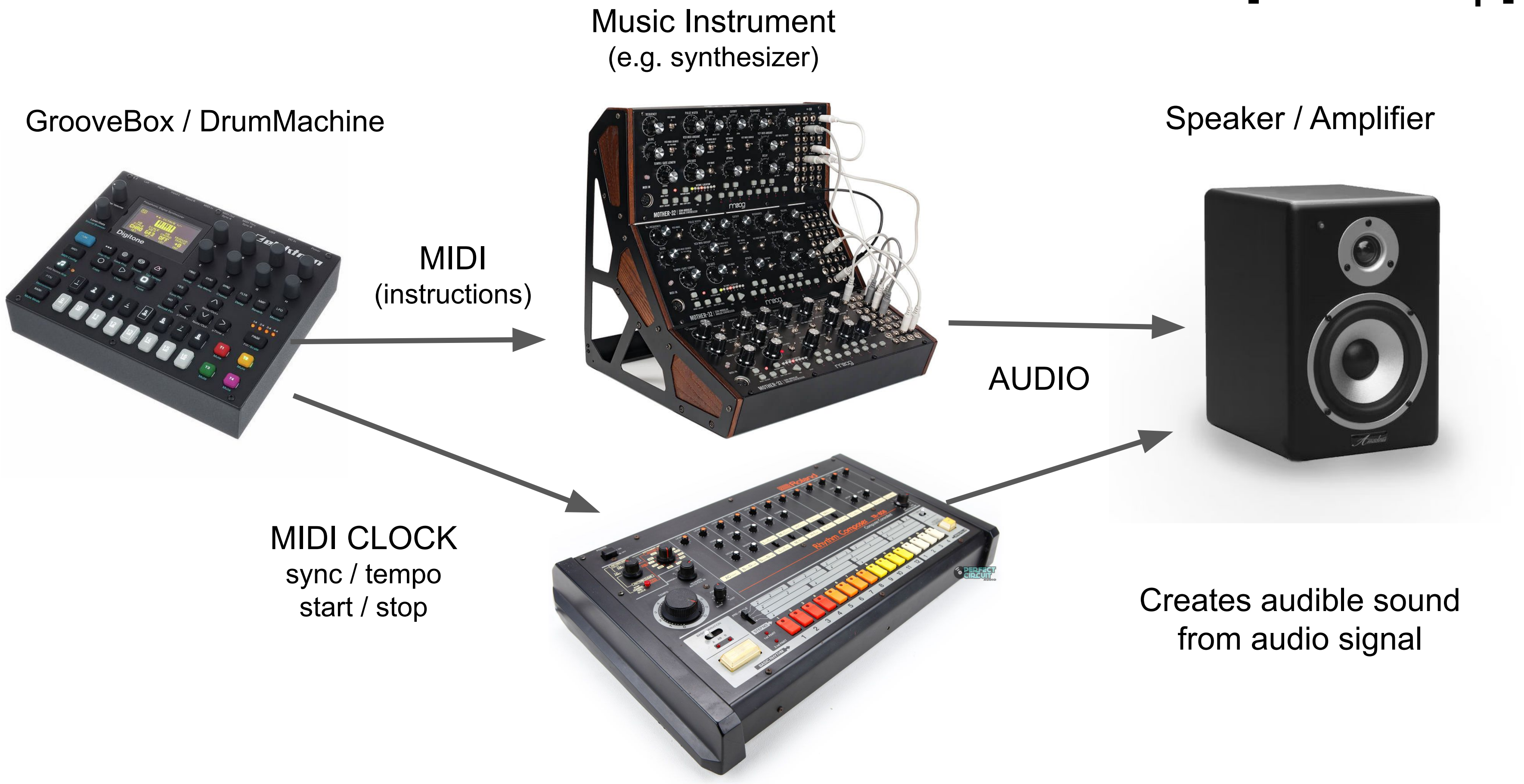
Tells the computer what
clips to play / effects to use.

Generates video with software
based on received Midi
instructions from controller

Creates Image / Video
Installation

the **M.I.D.I.** [workshop]

Fundamentals



GrooveBox / DrumMachine

MIDI
(instructions)

Music Instrument
(e.g. synthesizer)

MIDI CLOCK
sync / tempo
start / stop

AUDIO

Speaker / Amplifier

Creates audible sound
from audio signal

the **M.I.D.I.** [workshop]

Fundamentals

Our own custom
MIDI Controller

MIDI
(instructions)

Via USB

Computer

AUDIO

Video

WWW

MIDI Advantages

- Wide support in Music instruments, controllers, software programs (even web browsers)
- Simple serial protocol (doesn’t use much data)
- Connect via usb, or 5pin cable between other devices
- Don’t need to install drivers, support is build in OS.
- Can also be stored as a midi-file. Which can contain an entire orchestral score digitally.

One of the easiest ways to interface with other devices. Allows us to control computers in a different (more playful) way and approach devices as real instruments.

MIDI Disadvantages

- Low resolution (7 bits) means values go from 0 - 127 steps
- Could be faster (especially using usb)
- Predefined controls signals can be too limited to fully express the characteristics of a sound.

Examples

Some examples of how I have used MIDI

[Hacked controllers, custom instruments and installations]

Essential Instructions (MIDI Messages)

- **Note On, Note Off** messages

[Note Pitch]	(which key on the piano keyboard - pitch 60 = C4)	0 - 127	
[Note velocity]	(strength or force for that note)	0 - 127	(Note off = 0)
[Midi Channel]	(Channel is used to talk to multiple instruments	0 - 16	

- **Control Change** messages

[Control Function]	(e.g. 7 = change volume, 10 panning left/right)	0 - 127
[Control Value]	(amount for this function e.g. set volume to 80)	0 - 127
[Midi Channel]	(Channel is used to talk to multiple instruments	0 - 16

Others MIDI messages

- Program Change, pitch bend, aftertouch (key pressure)
- Clock, Transport (play, start, pause etc)
- Sysex - raw data (e.g. used to transfer user presets etc)

MIDI Messages

Unless we want to control a specific instrument (e.g. a Moog synthesizer, or Drum machine in Software program like Ableton) We can freely use these Note and Control Messages ourselves as we see fit within our own designs, and most software allows us to Map Note and Control Messages to various functions.

Although it is originally designed to control the synthesizer instrument, the protocol is often [hacked] to serve other purposes because of its wide support in hardware and software.

For example

- A Note-On is used to trigger a videoclip in a VJ program.
- Control Messages can be assigned to mix two tracks in a DJ program.
- A Note-On / Note-Off is often accepted in return by midi controllers to turn lights of buttons on and off.

Official Manufacturers of Instruments and devices (should) publish their implementation as their ‘Midi Specification’ which you can you look up if you want to control a specific instrument (but they do not always give you all of the specs).

[DEMO TIME]

Building the super basic midi controller

1 button and 1 fader (or rotary potmeter)

As a starting point for your own designs.

Building basic midi controller

[Set Up Software]

Do the blink test, see if environment is set up and we can upload code to the teensy

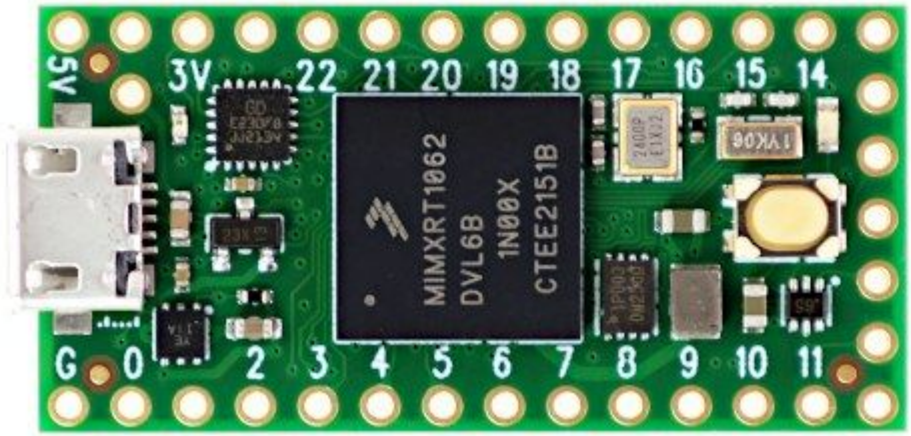
Teensy Library installation instructions:

https://www.pjrc.com/teensy/td_download.html

Teensy4.0 functions as the brain of our midi controller controller.

- Build-in MIDI firmware
- Lots of analog inputs for faders and pots
- Arduino Compatible
- Additional software libraries
- Blazing Fast (600Mhz)

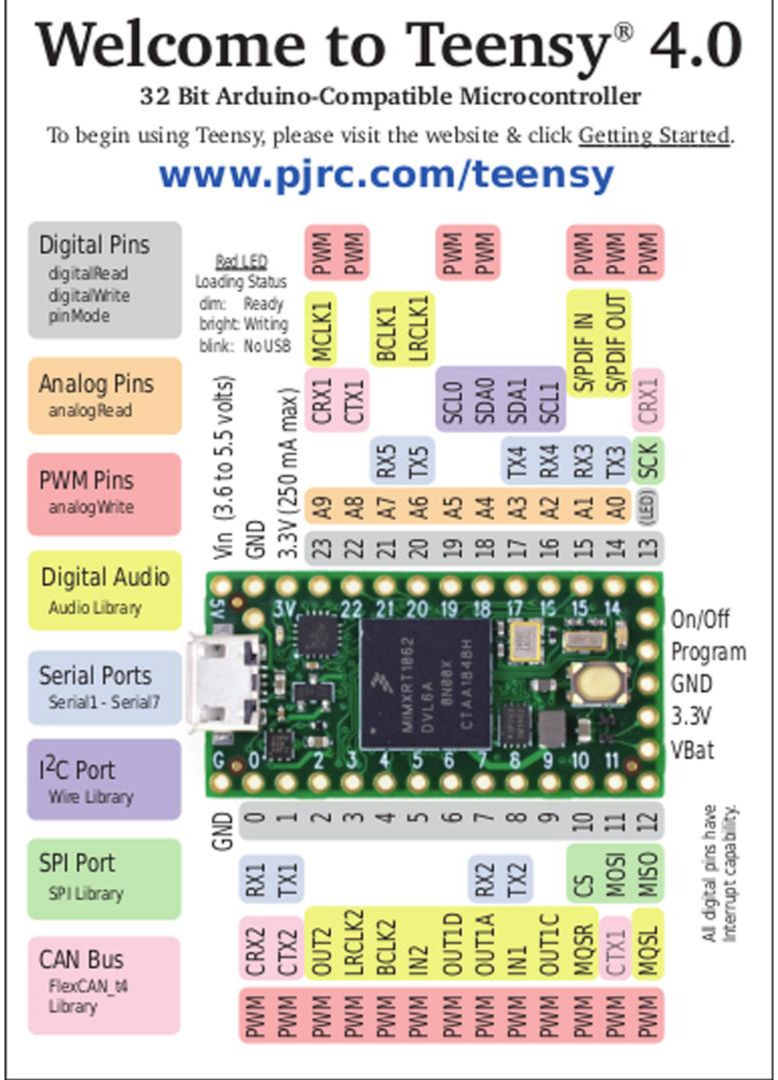
- Not cheap
  (will link to alternatives)



Building basic midi controller

the **M.I.D.I.** [workshop]

Building basic midi controller



Welcome to Teensy® 4.0

32 Bit Arduino-Compatible Microcontroller

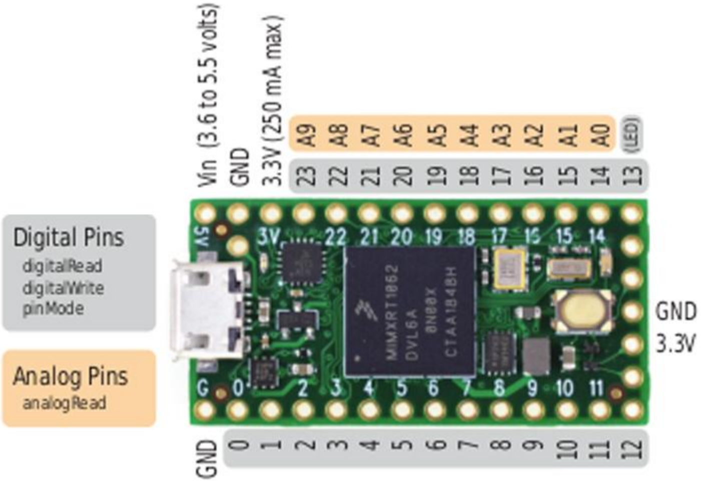
To begin using Teensy, please visit the website & click Getting Started.

www.pjrc.com/teensy

See the included Teensy 4.0 pinout card)

https://www.pjrc.com/store/teensy40.html

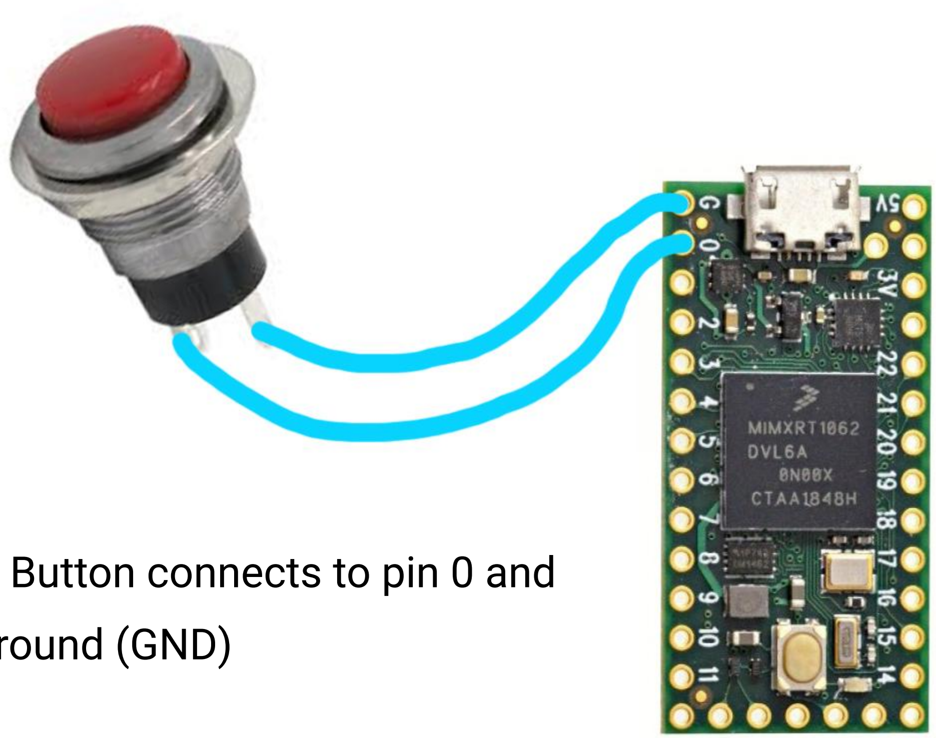
Building basic midi controller



See the included Teensy 4.0 pinout card)

https://www.pjrc.com/store/teensy40.html

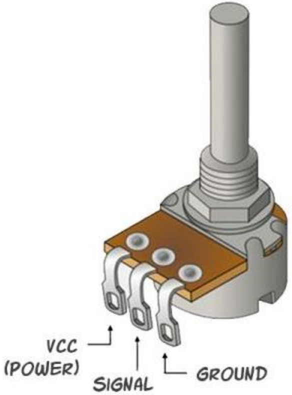
Building basic midi controller

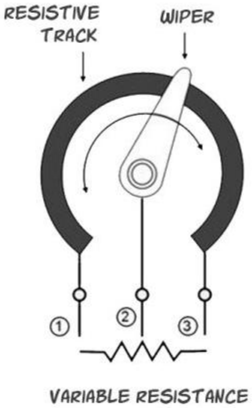


-> Button connects to pin 0 and Ground (GND)

the **M.I.D.I.** [workshop]

Building basic midi controller





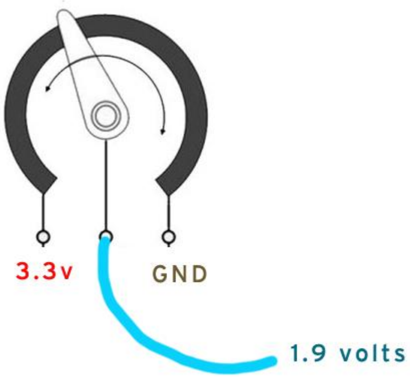
Potmeter overview

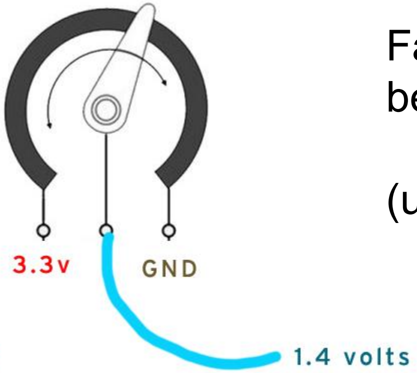
The Fader works exactly the same, different style / housing.

Potmeter: output is the middle pin (pin 2)

Fader: output position can vary between designs.

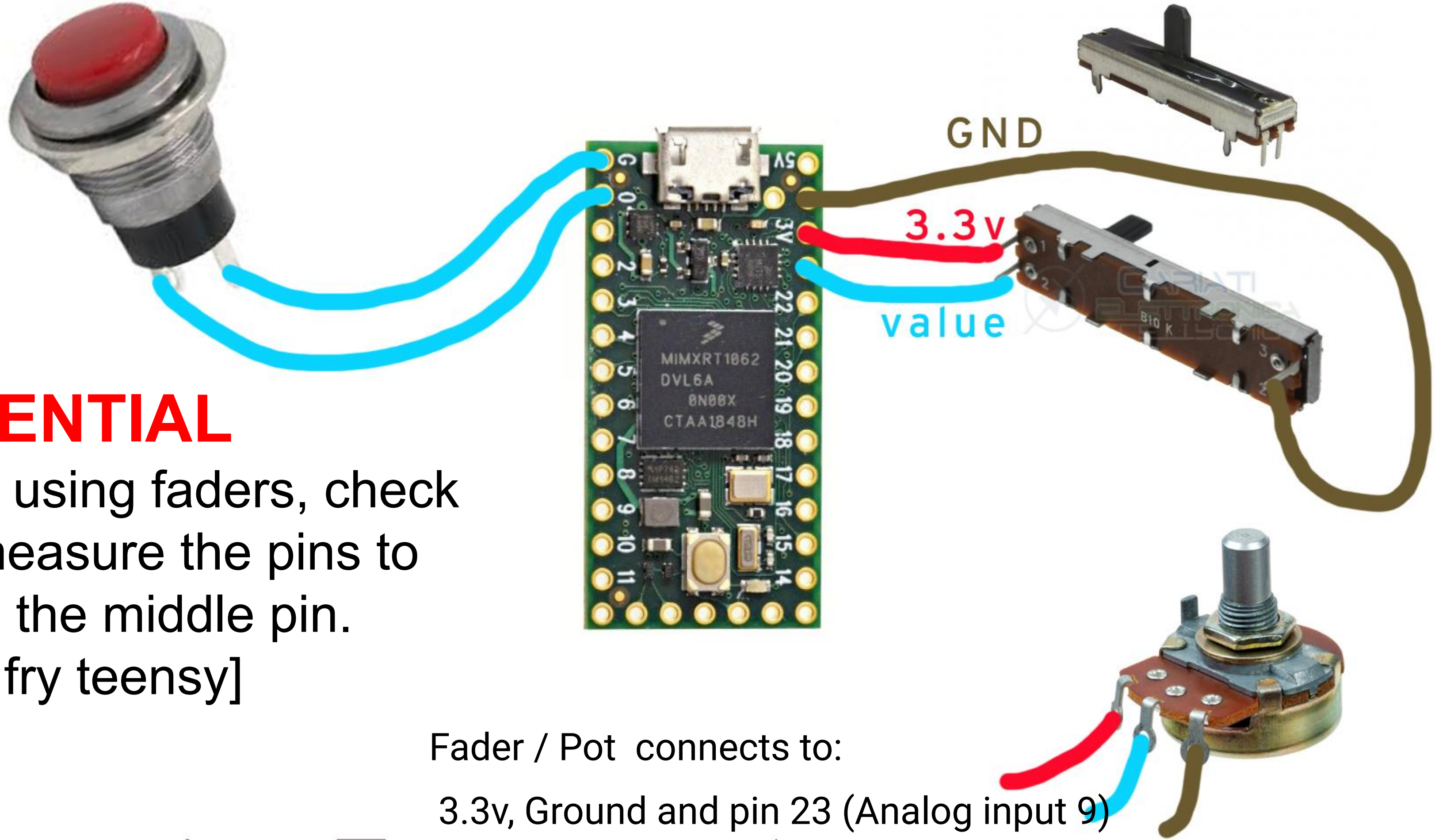
(usually labeled: pin 2)





the **M.I.D.I.** [workshop]

Building basic midi controller



ESSENTIAL

When using faders, check
and measure the pins to
ocate the middle pin.
[don't fry teensy]

Fader / Pot connects to:

3.3v, Ground and pin 23 (Analog input 9)

[Coding Time]

Testing the hardware.

Teensy Library installation instructions:

https://www.pjrc.com/teensy/td_download.html

Building basic midi controller