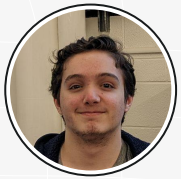
ECE Robotics and Microcontroller Workshop

Introductions

Prof. Hamid Timorabadi
h.timorabadi@utoronto.ca

Ryan Seto
ryan.seto@mail.utoronto.ca

Iris Wang
irisjiayi.wang@mail.utoronto.ca



Nick Mutlak
n.mutlak@mail.utoronto.ca

Shuntaro Wakamatsu
shuntaro.wakamatsu@mail.utoronto.ca

Simon Xu
xiaofengsimon.xu@mail.utoronto.ca

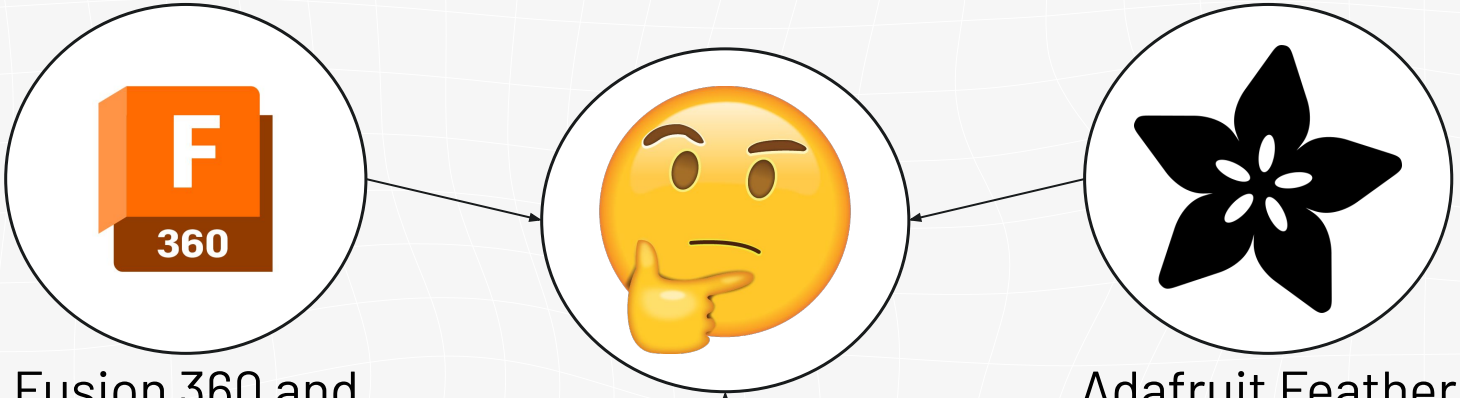
Preamble

Mechanical

Electrical

Software

Preamble



Fusion 360 and
3D Printing

Adafruit Feather
(ATmega32u4)

Arduino

Workshop Objective

Workshop Roadmap

Day 1

Learning how to CAD and why engineers use it

Day 2

Learning microcontrollers and general signal I/O

Day 3

Controlling motors and basic line tracking

Day 4

Discussing 3D printing and designing your own robot

Day 5

Programming bluetooth control and maze challenge

Day 6-8

Programming your design for the final competition

Challenges and Prizes

Challenge 1: Basic Line Tracking (Day 3)

- Your robot will have to autonomously traverse a simple line tracking course consisting of straight lines and shallow curves
- Fastest team to complete the course with the most precise line tracking wins a prize (TBD)

Challenge 2: Piloted Maze (Day 6)

- You will have to drive your robot through a maze of pins as the walls quickly
- Knocking over any pins results in a time penalty
- Fastest team wins a prize (TBD)

Challenges and Prizes Cont'd

Challenge 3: Advanced Line Tracking (Day 8)

- Your robot will have to autonomously traverse a complicated line tracking course with sharp turns, dotted lines, and dead ends
- Fastest team to complete the course with the most precise line tracking gets to keep their robot (**$150+ kit**)

Activity and Challenge Sheets

Phase 1: Guided Learning (Day 1-3)

- Activity 1-2: Learning how to CAD
- Activity 3-4: Programming a microcontroller for signal input and output
- Activity 5: Getting the robot in motion
- Challenge 1: Basic line tracking challenge

Phase 2: Open Designing (Day 4-8)

- Activity 6: Designing your own robot CAD
- Activity 7: Bluetooth manual control of robot
- Challenge 2: Piloted maze challenge
- Challenge 3: Advanced line tracking challenge

Final Notes Before We Begin

This workshop will be cover a lot of content at a high level to expose you to many facets of engineering, so if you ever feel lost ask questions earlier rather than later, as the subsequent lessons will feel overwhelming. Throughout the workshop, we will be covering bits and pieces from multiple courses in your second year:

- Visualizing 3D shapes in CAD
  → Helps with **MAT291 (Calculus 3)** and 3D integration
- Working with digital signals
  → Helps with understanding the context behind **ECE241 (Digital Systems)**
  → Helps with understanding discrete signals in **ECE216 (Signals and Systems)**
- Programming a processor to execute some code and different types of communication protocols
  → Helps with **ECE243 (Computer Organization)** labs

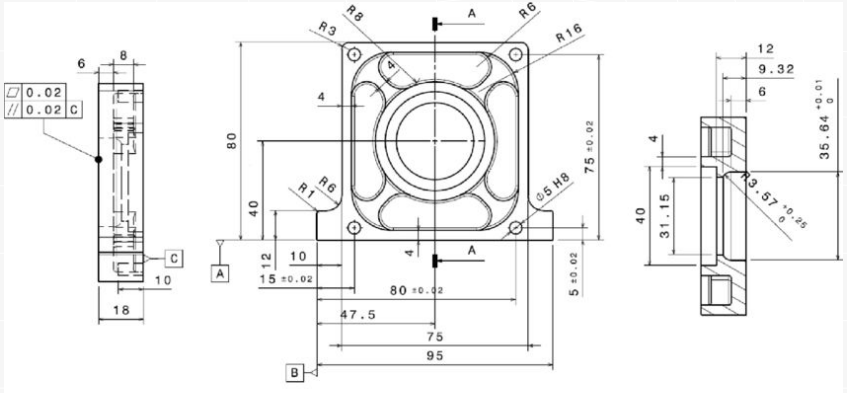
01

CAD Modelling and Fusion 360

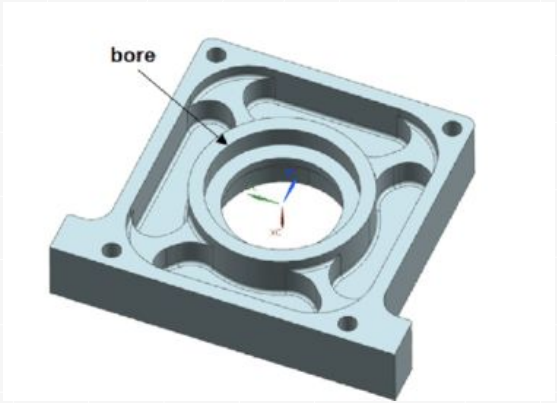
What is CAD Modelling?

CAD Modelling is a tool for engineers to **design** the shapes of their part(s) in order to **manufacture** it in a shop or fabrication center.

- Allows engineers to **simulate** a design before actually building their design
- Before CAD modelling, engineers would have to make 2D drawings of a part



2D Drawing



3D CAD Model

Design and Manufacturing Cycle

CAD Design

CAD Each Part/Component

Build a CAD Assembly

Manufacture Design

Manufacture Each Part

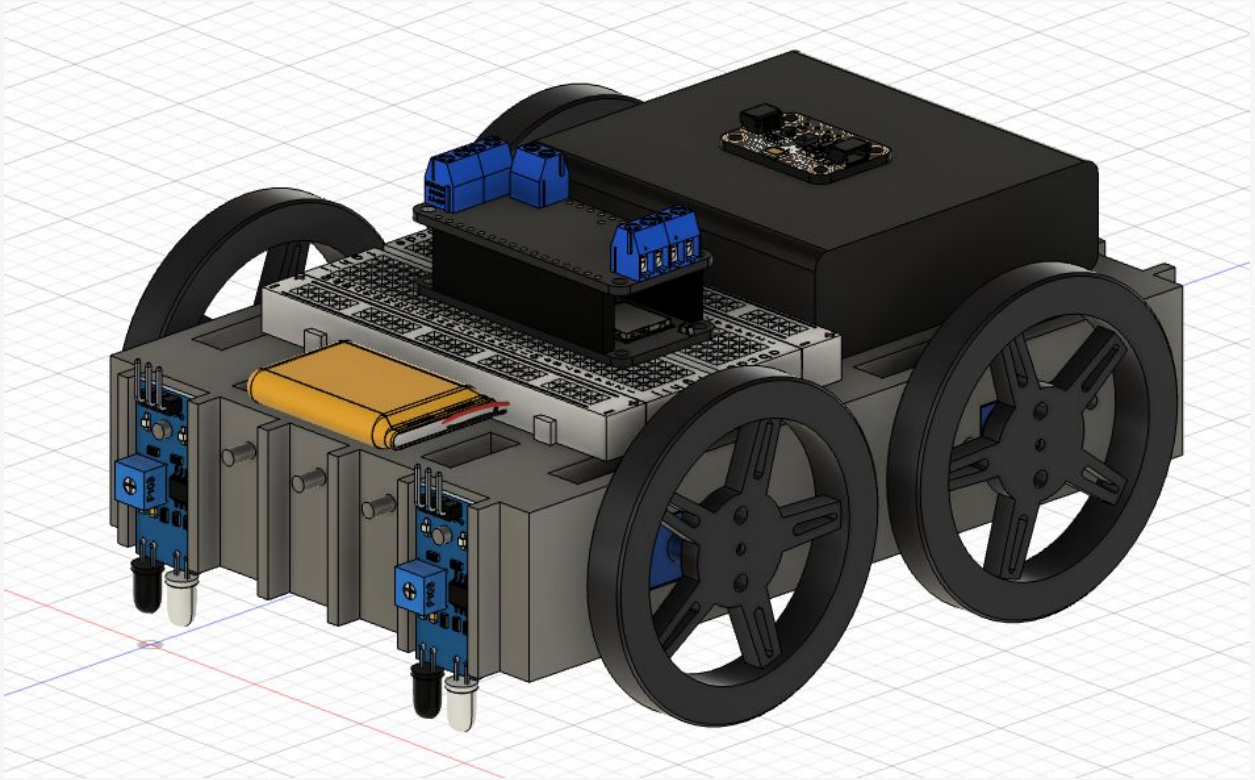
Physically Assemble

Benefits of CAD

CADing/simulating a design has three main benefits:

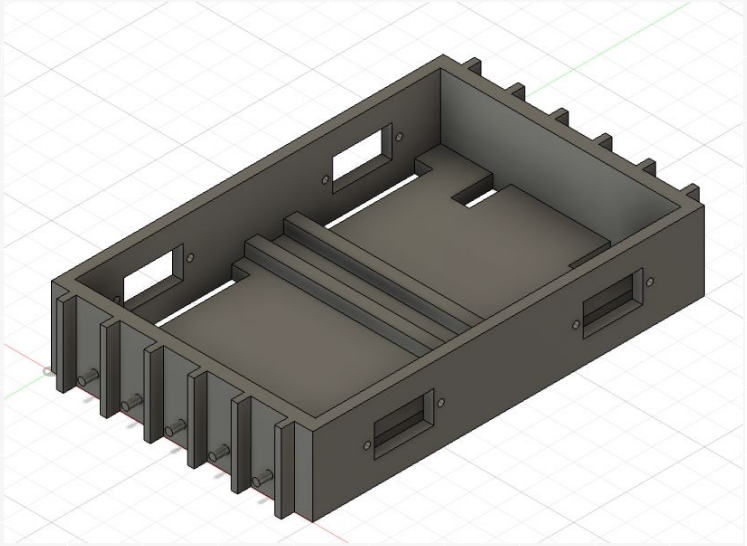
1. Allows engineers see if all the parts they design will fit together during assembly (**proper clearance and no collisions/conflicts**)
2. Allow engineers to assign densities to parts and see the total mass and the center of mass of the design alongside other physical properties (**validating weight requirements**)
3. Allows modifications to old models without compromising the original model (**version history to refer back to old designs**)
    a. No longer need to redraw the model from scratch in order to modify it

What Are You Designing?



Activity 1: Learning How to CAD Components

To learn how to build individual parts/components, you'll be following a tutorial to CAD this drivebase. Handout is on your workbench or in the directory.

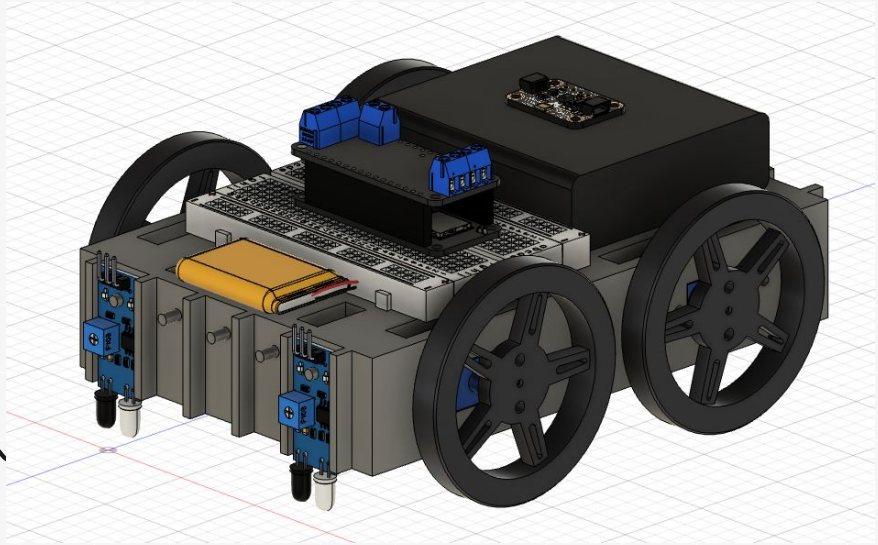


BREAK

Be back in 15 minutes

Activity 2: Learning How to CAD Assemblies

Next, to learn how to make CAD assemblies you'll be following a second tutorial to complete the full drivetrain assembly. If you're missing all the files, please talk to an instructor or revisit the preparation handout for this workshop.



Design Process Behind Drivebase

Summary:

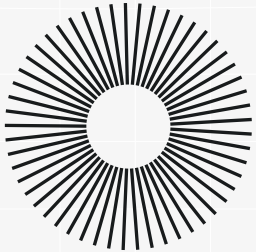
- Drivebase size is limited by the printing bed size
- Top of drivebase is flat to facilitate better bed adhesion for 3D printing
- Motor cutouts were made larger than necessary to ensure the motor can fit
- Ribs were attached for more structural support
- Multiple IR sensor mounts were created to allow you to experiment with where to place your IR sensors
    - The mount peg was made slightly smaller than the IR sensor hole to ensure the sensor could fit on top
- Wire cutouts double as a place to fasten the batteries with a belt

Day 1 Complete!

02

Microcontrollers

Processors, Pinouts, and Digital Signals



What is a Microcontroller

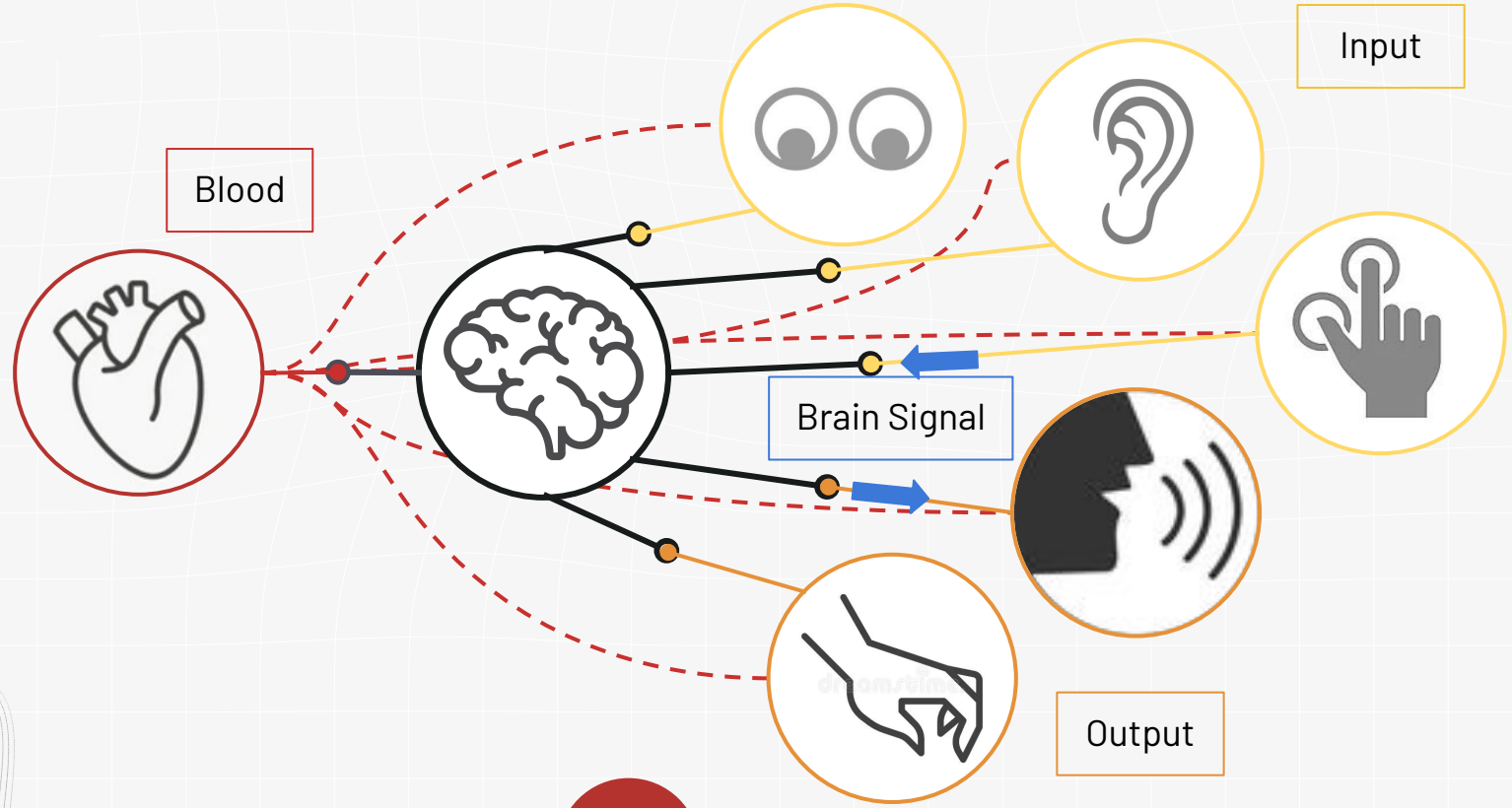
A microcontroller is a small, integrated circuit that you can program to complete specific tasks

- Ex: turn on an LED, spin a motor, monitor the ambient light in a room

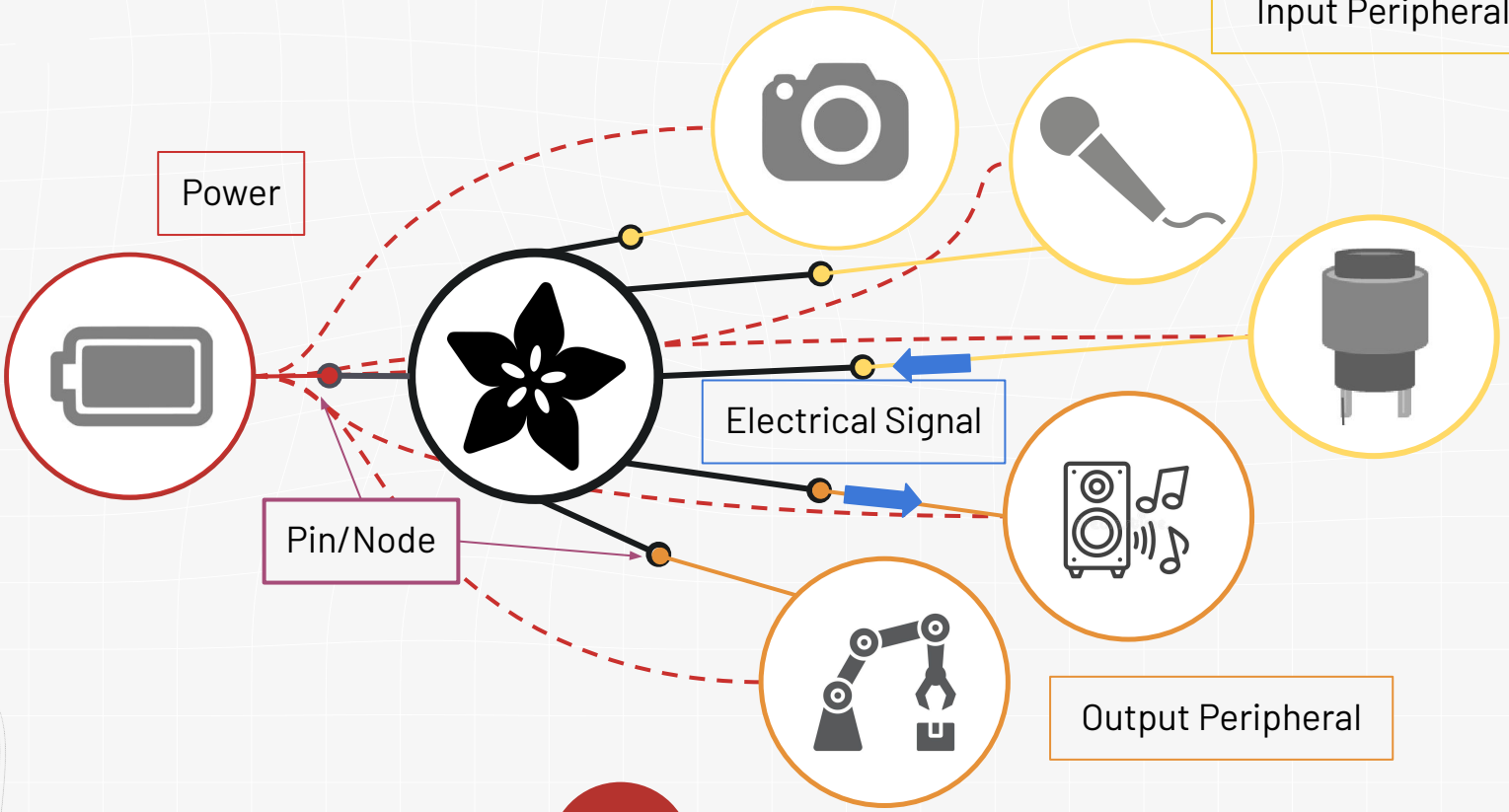
Manually adjust function/signal generator to control a circuit

Program a microcontroller to generate the signal and control a circuit

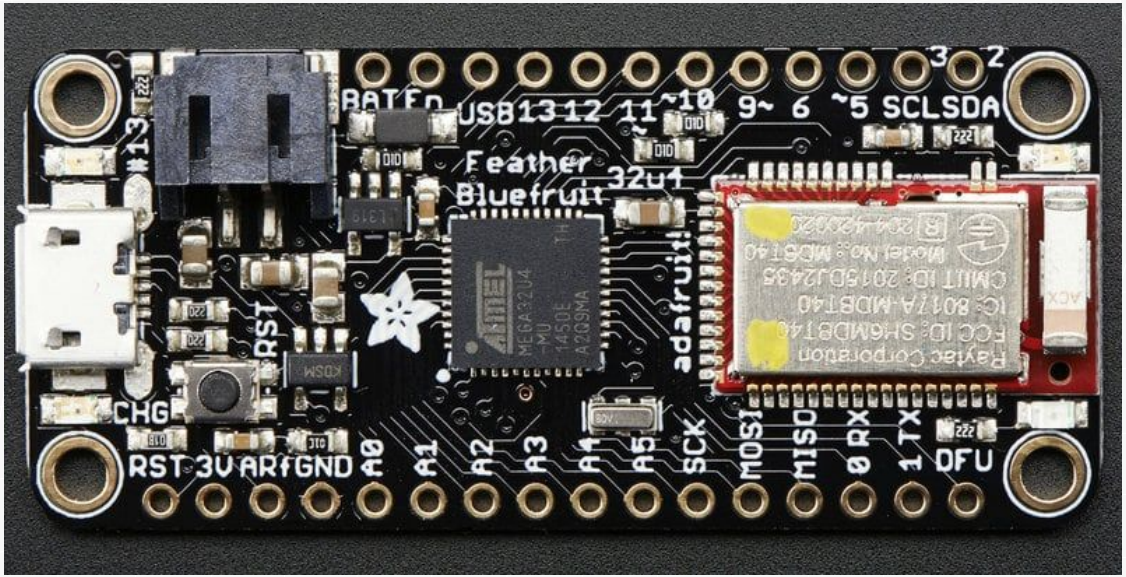
Microcontroller Analogy



Microcontroller Analogy

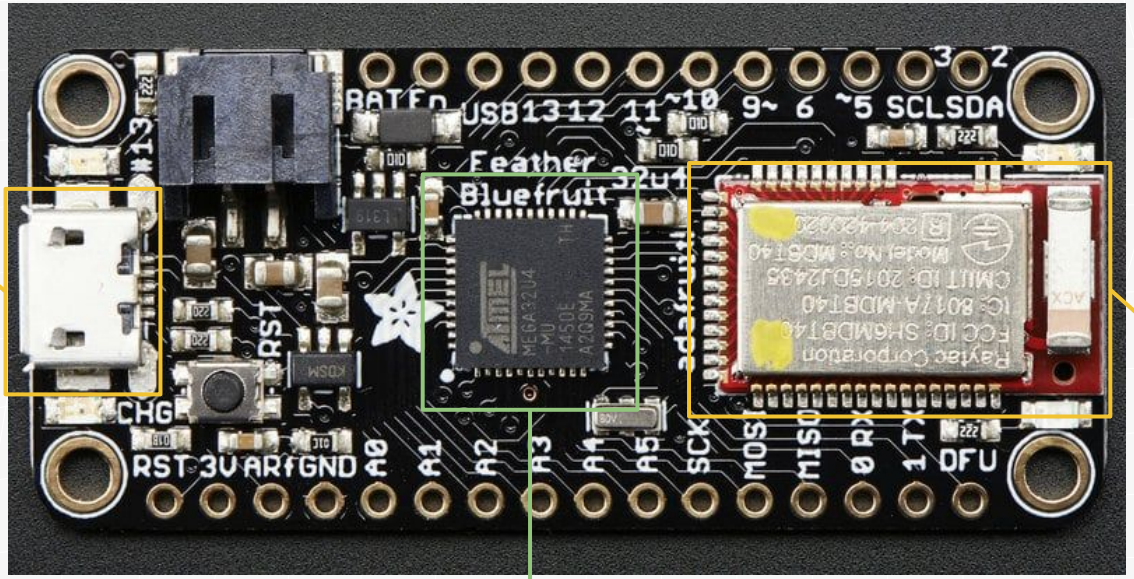


The Microcontroller You Will Be Using



Processor and Internal Peripherals

USB Peripheral

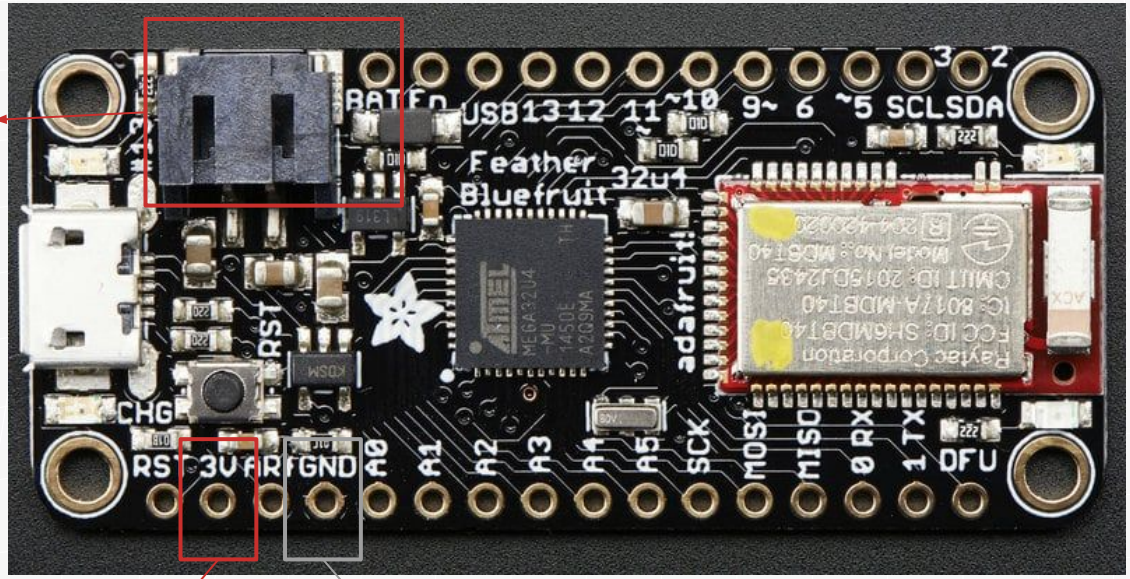


Bluetooth Peripheral

Processor Chip
(ATmega32u4)

Power Pins

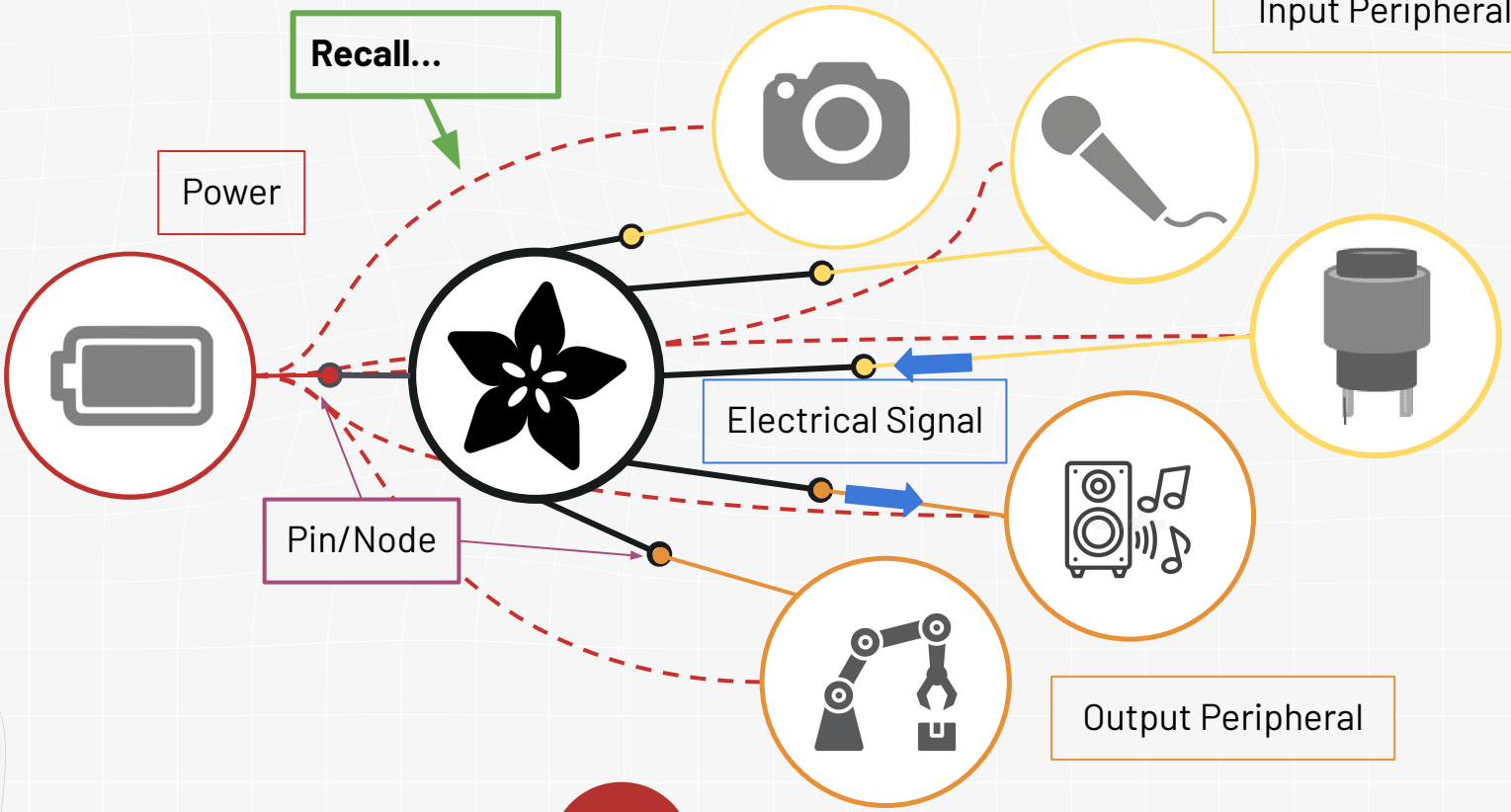
BAT pin and **JST Connector** for LiPo battery power



GND pin: voltage ground for all logic and power

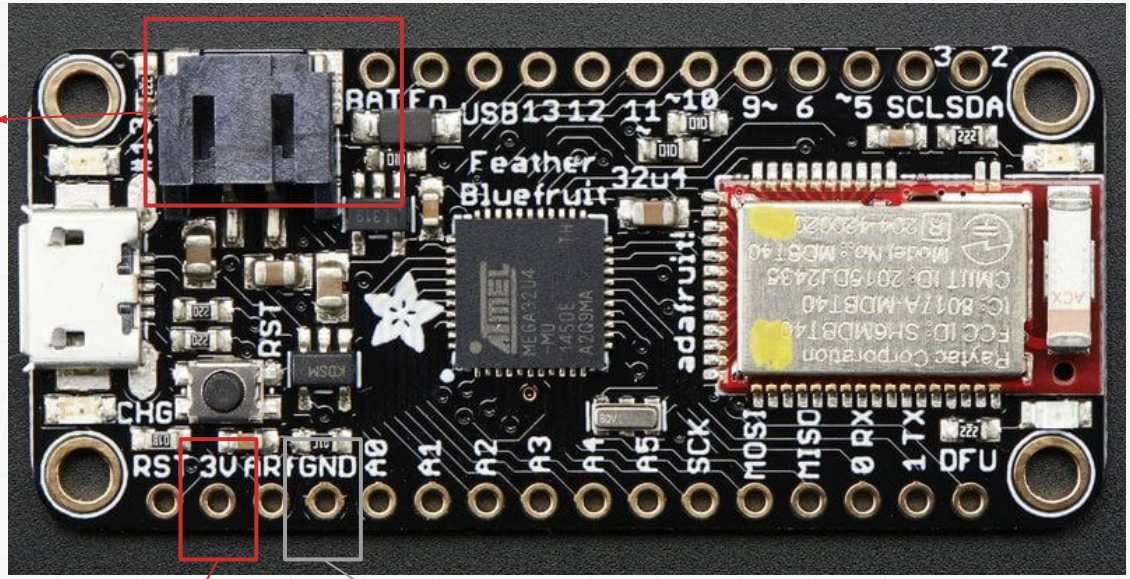
3V pin: 3V voltage source to power external peripherals

Microcontroller Analogy



Power Pins

BAT pin and **JST Connector** for LiPo battery power

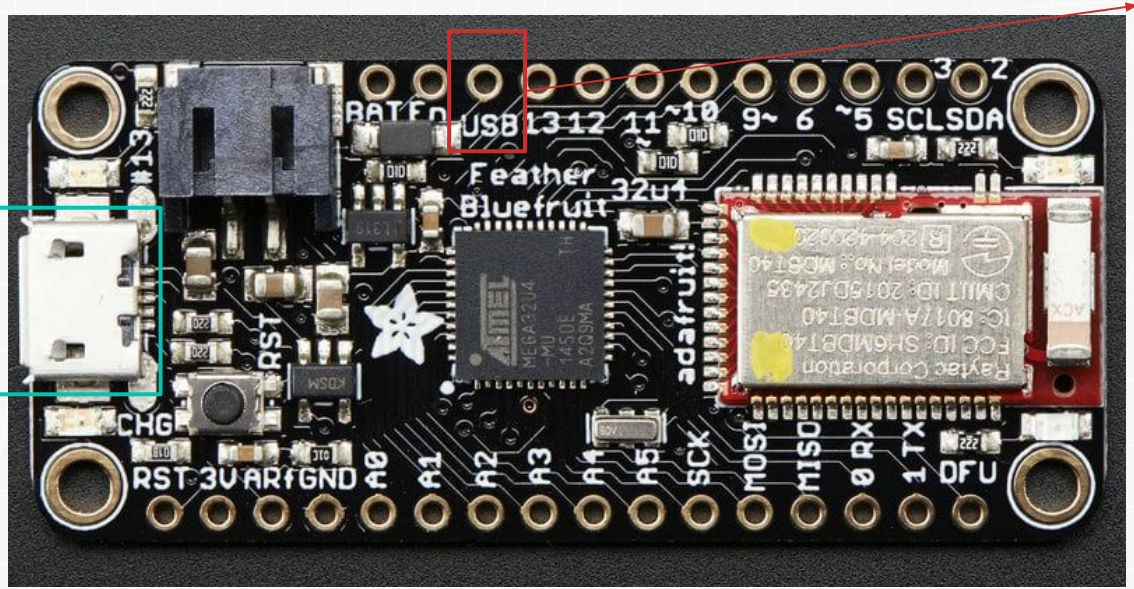


GND pin: voltage ground for all logic and power

3V pin: 3V fixed voltage source to power external peripherals

Programming Pins

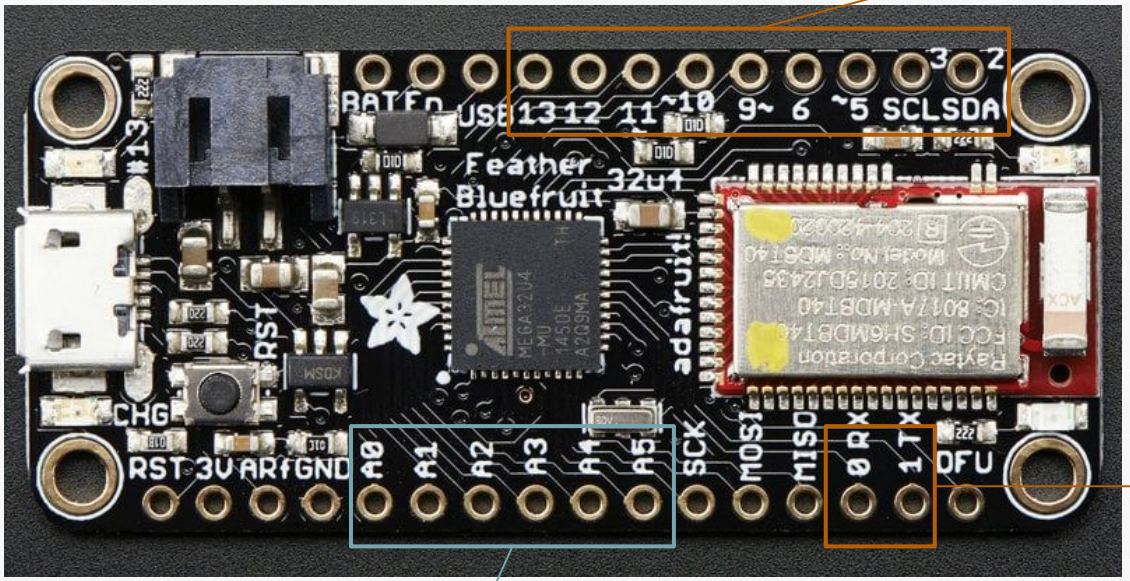
USB pin and **MicroUSB Connector** for connecting the board to your computer



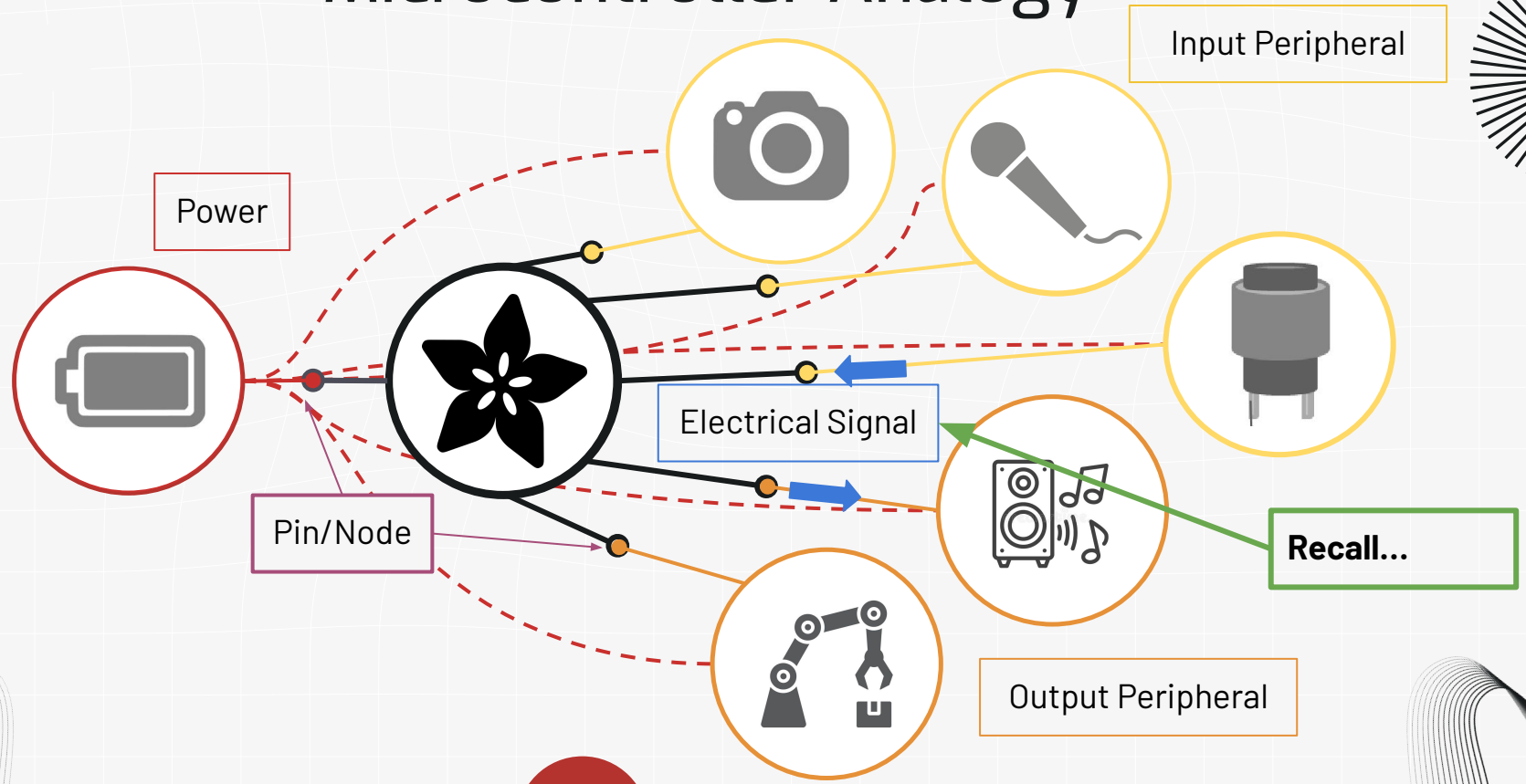
MicroUSB Power

Variable Signal Pins

Notice digital pins 4, 7 and 8 are missing. They are used by the Bluetooth Peripheral so they cannot be used.



Microcontroller Analogy



Digital vs. Analog Signals

Recall that the microcontroller receives and sends electrical signals through the input peripherals and output peripheral. These electrical signals are **voltages**

An analog signal is a continuous/real signal like the signals you've worked with up to this point in electrical courses

A digital signal is an analog signal broken down into discrete value (ie. a processed analog signal) and transmitted as a **pulse/serially** (ie. strings of bits)

Digital Signal Analogy

Speech

A B C D E F G H
I K L M N O P Q
R S T V X Y Z

Alphabet

HELLO
WORLD

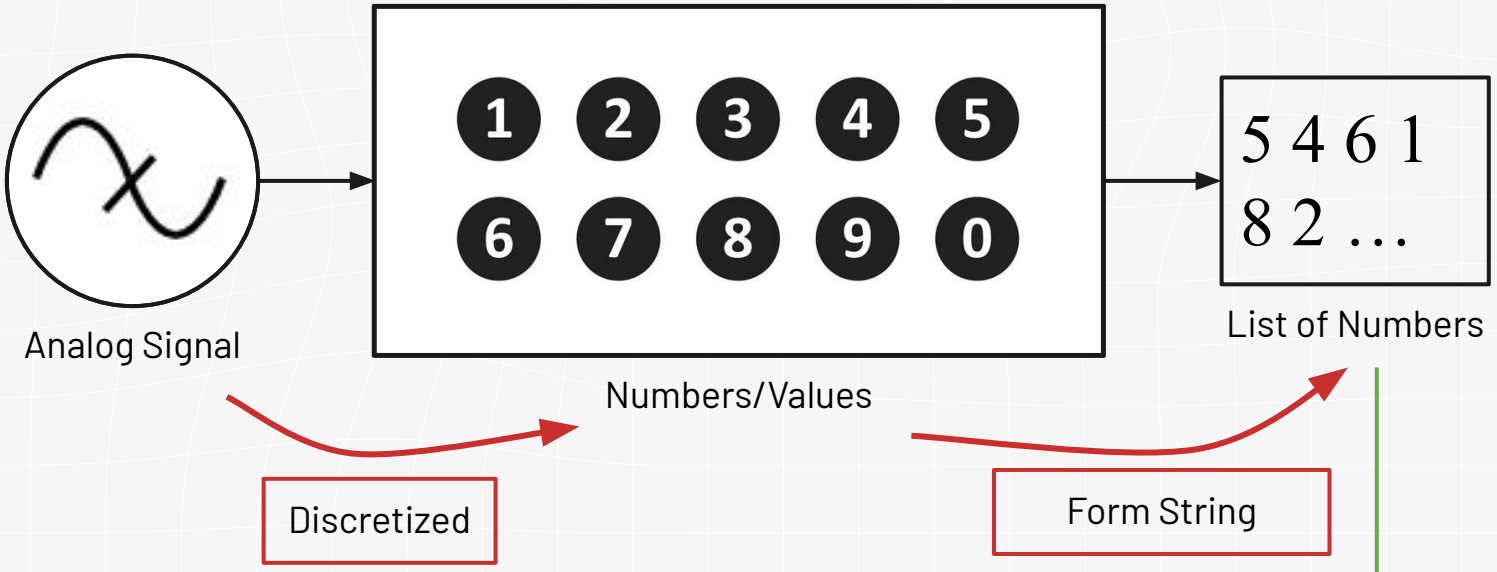
Words

Discretized

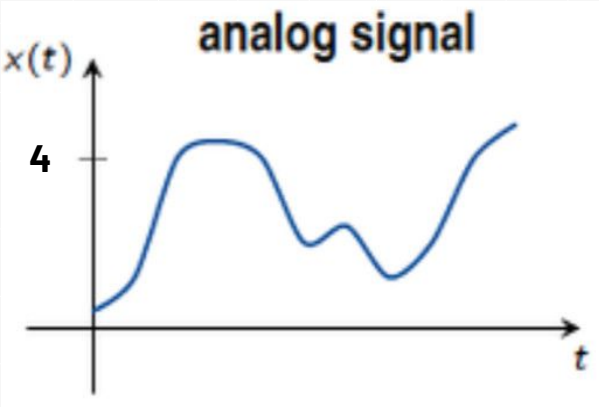
Form String

If you know how to interpret words, you can always remake the spoken word

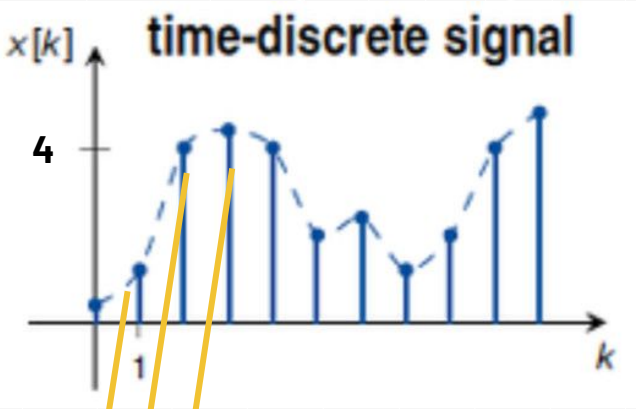
Digital Signal Analogy



Graphical Interpretation



Discretization



We're **sampling** the function

Form String

x[k] = [1, 4, 4.2, 4, 2, 2.3, 1, 2, 4, 4.6]

Successfully broken down an analog signal

Digital vs. Analog Signals

Recall that the microcontroller receives and sends electrical signals through the input peripherals and output peripheral. These electrical signals are **voltages**

An analog signal is a continuous/real signal like the signals you've worked with up to this point in electrical courses

A digital signal is an analog signal broken down into discrete value (ie. a processed analog signal) and transmitted as a **pulse/serially** (ie. strings of bits)

1

2

Binary Numbers and Bits

A bit (binary digit) is a digit that only has two values: 1 (on) or 0 (off). A binary number is a string of bits. We can represent standard decimal integers with combinations/strings of bits like the following:

Decimal Number	4 bit Binary Number ABCD
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1
10	1 0 1 0
11	1 0 1 1
12	1 1 0 0
13	1 1 0 1
14	1 1 1 0
15	1 1 1 1

Binary Numbers and Bits Cont'd

With 4 bits (2 values), we can get 16 unique integers..., how many integers can you represent with 3 bits?

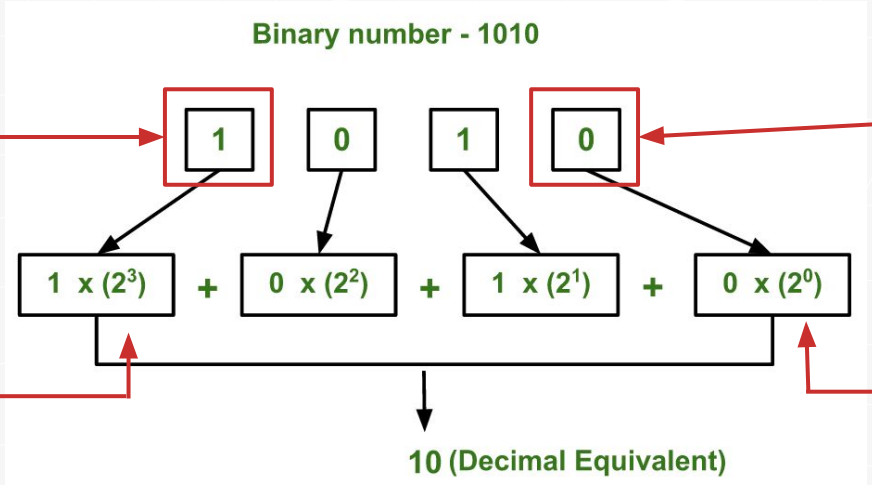
- Recall for decimal numbers, 3 digits ⇔ 0 to 999 ⇔ 1000 values ⇔ 10^3

Decimal	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

$8 \Leftrightarrow 2^3 \Leftrightarrow (\text{base})^{(\#\ \text{digits})}$

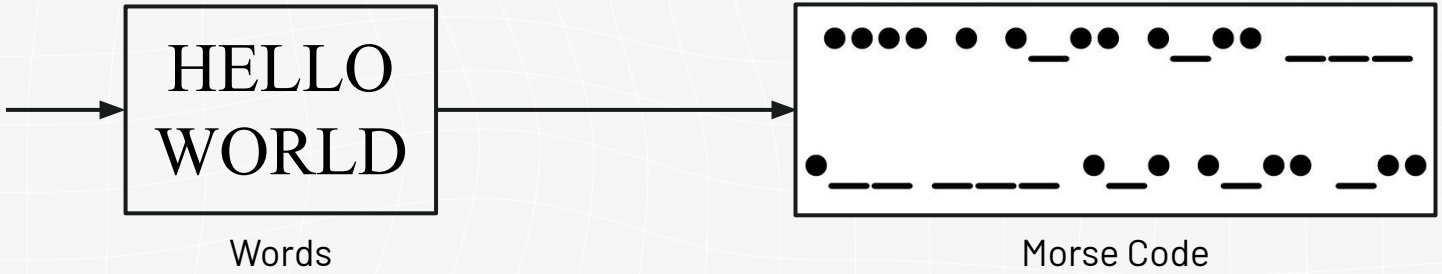
Binary-to-Decimal Conversion

Most Significant Bit
- Largest influence on magnitude

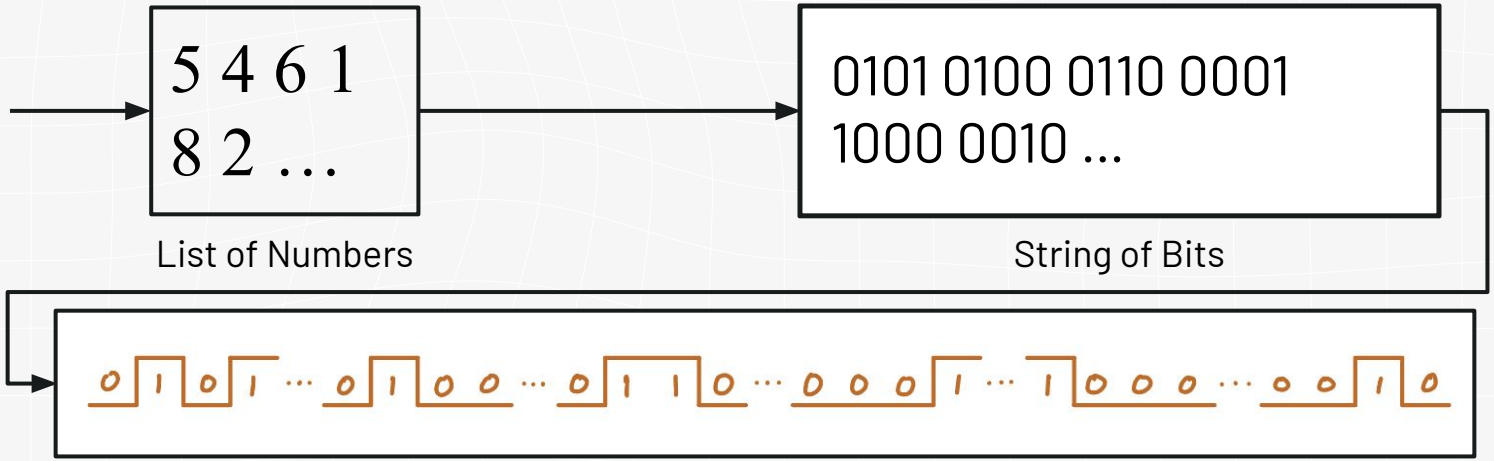


Least Significant Bit
- Smallest influence on magnitude

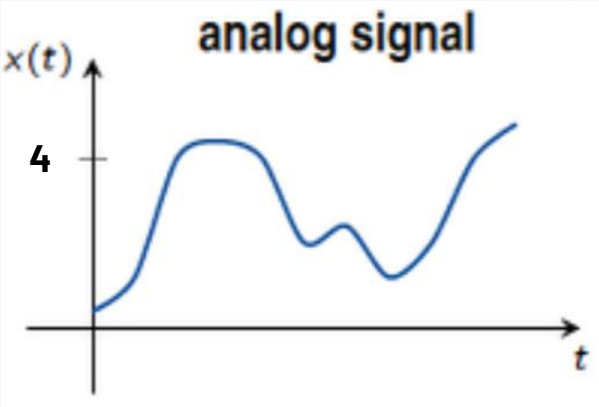
Digital Signal Analogy Cont'd



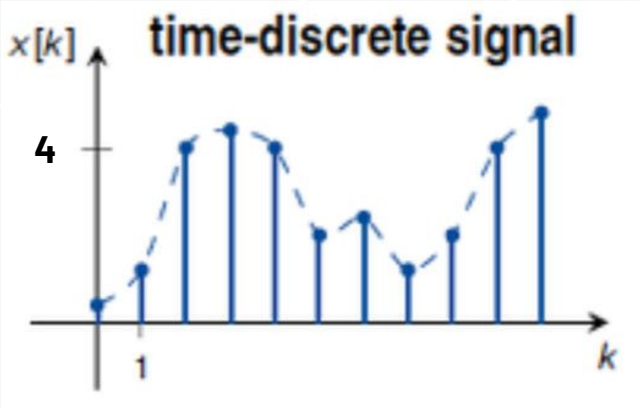
Digital Signal Analogy Cont'd



Graphical Interpretation



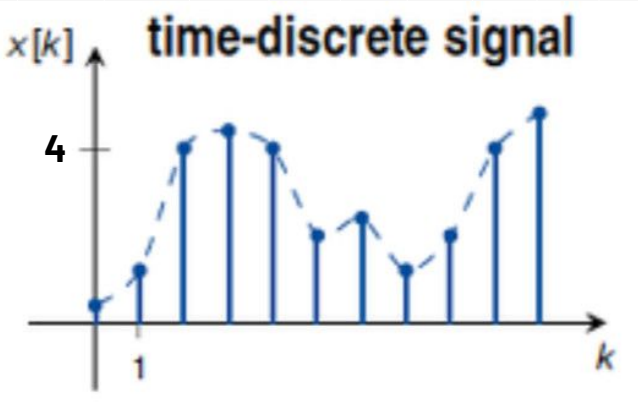
Discretization



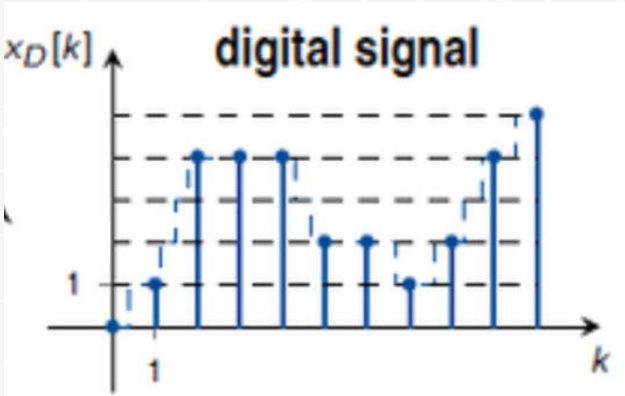
Form String

x[k] = [1, 4, 4.2, 4, 2, 2.3, 1, 2, 4, 4.6]

Graphical Interpretation



Round



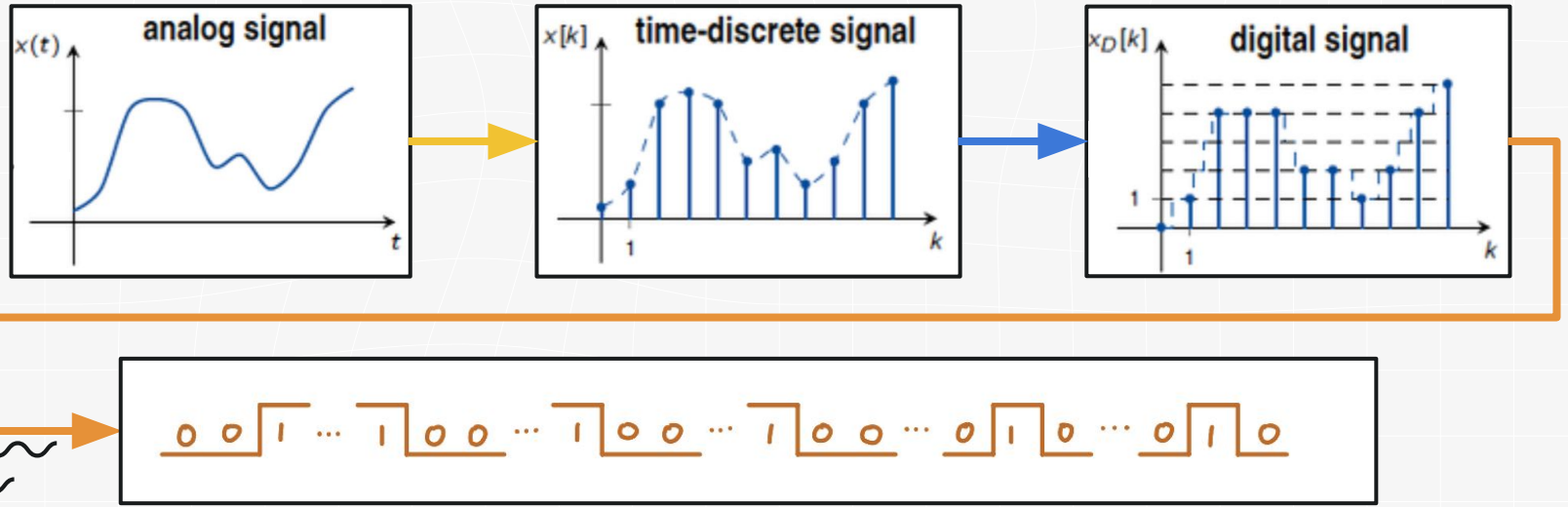
Form String

Rounding to **quantized** levels

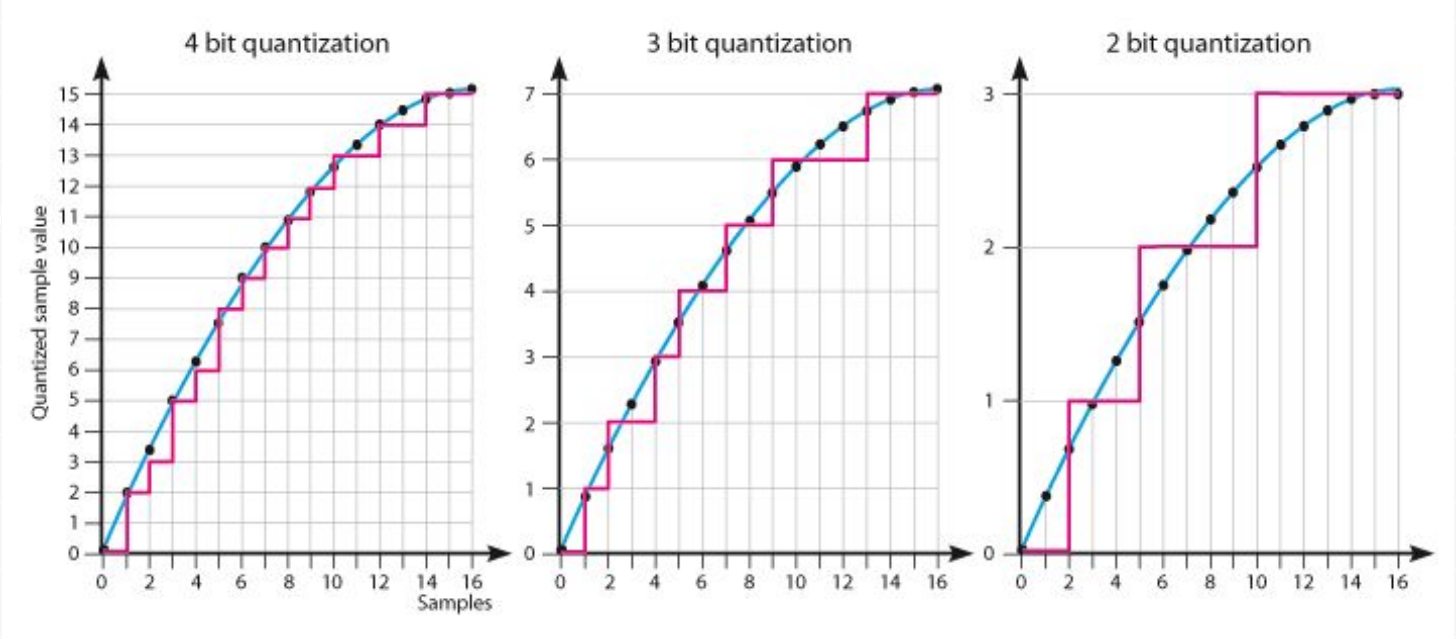
x[k] = [1, 4, 4, 4, 2, 2, 1, 2, 4, 5]

Summary of Digital Signals

Digital signals are analog signals that have been **sampled** and **quantized** into discrete, integer values that can be encoded into a **pulse wave** for **serial communication**



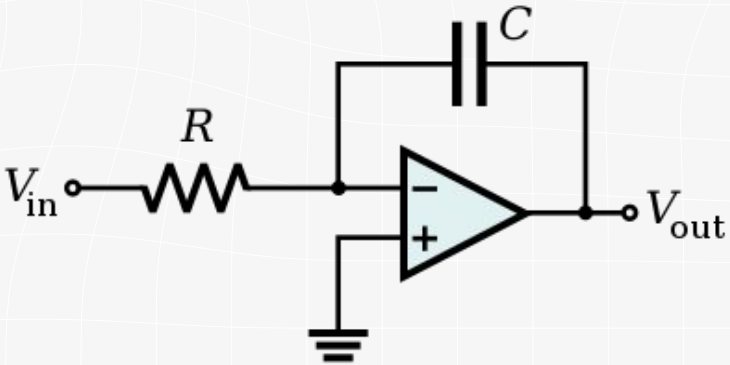
Analog Signal Chopping (Discretization)



Larger bit length (n) means more precise digitization

Why Not Use An Analog Computer?

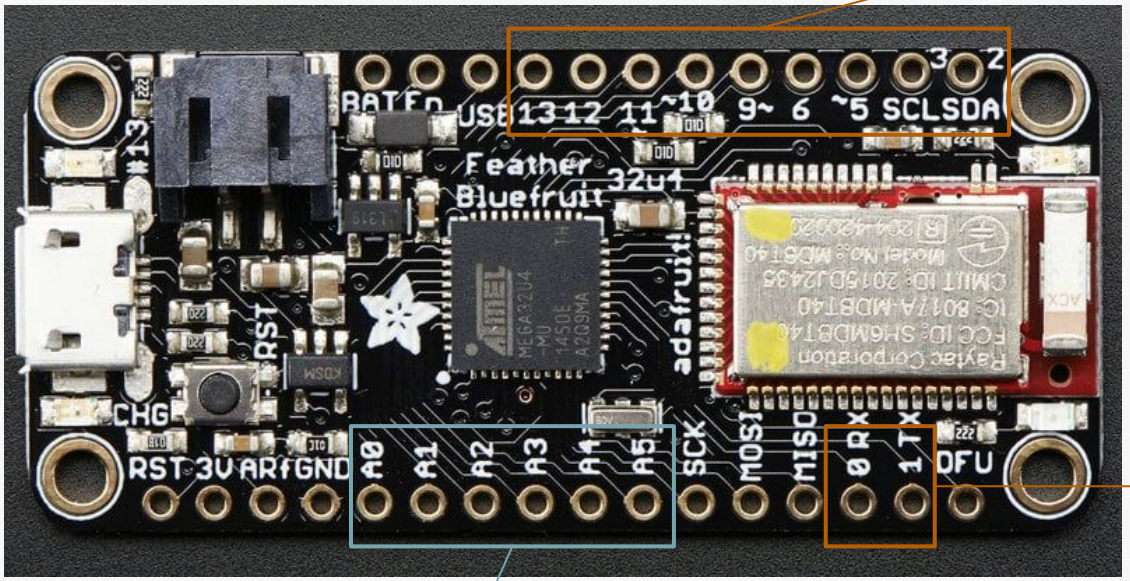
This seems like a lot of work to create convert analog signals into digital signals, so why do this? The reason is because analog computers are not easy to modify, meaning an analog computer needs to be built for a specific purpose whereas digital computers are very easy to modify and you can rebuild analog signals from digital ones. You'll learn more about the math behind the signal processing in **ECE216**.



Op Amp Integrator
(Cannot be used for anything else)

Signal Pins

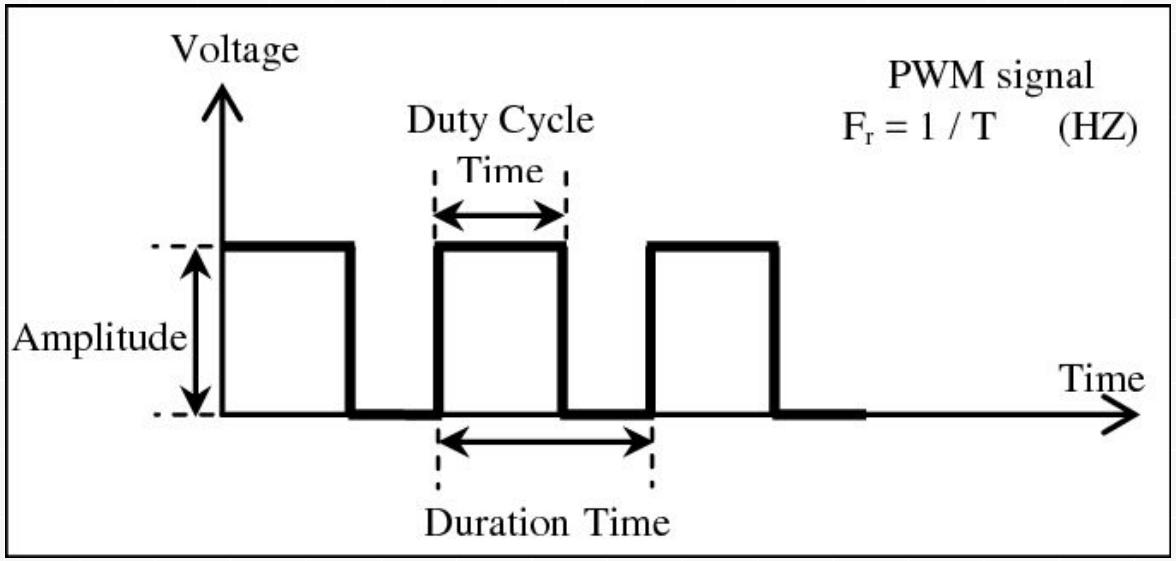
Notice digital pins 4, 7 and 8 are missing. They are used by the Bluetooth Peripheral so they cannot be used.



Pulse Width Modulation (PWM)

A pulse is a signal that switches between HIGH and LOW

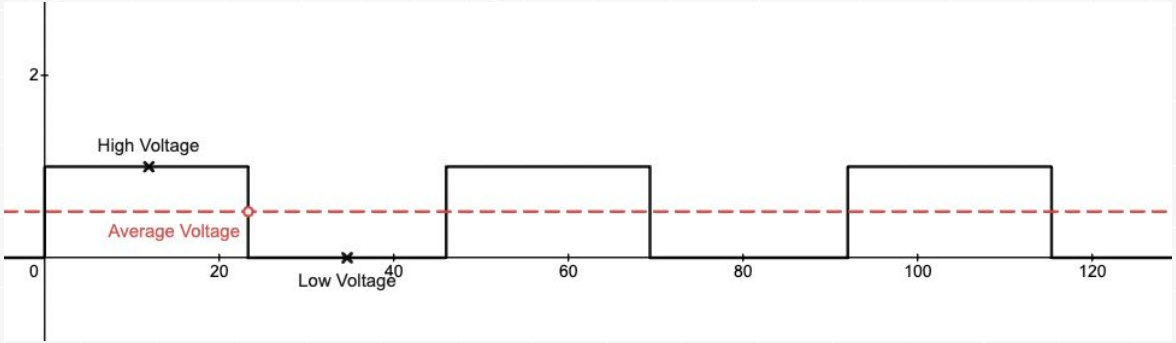
- Pulses with a frequency can undergo PWM



Average Value of a PWM

Like other AC signals, a repeating pulse has an average value; this average value can be thought of as the DC value of a PWM signal.

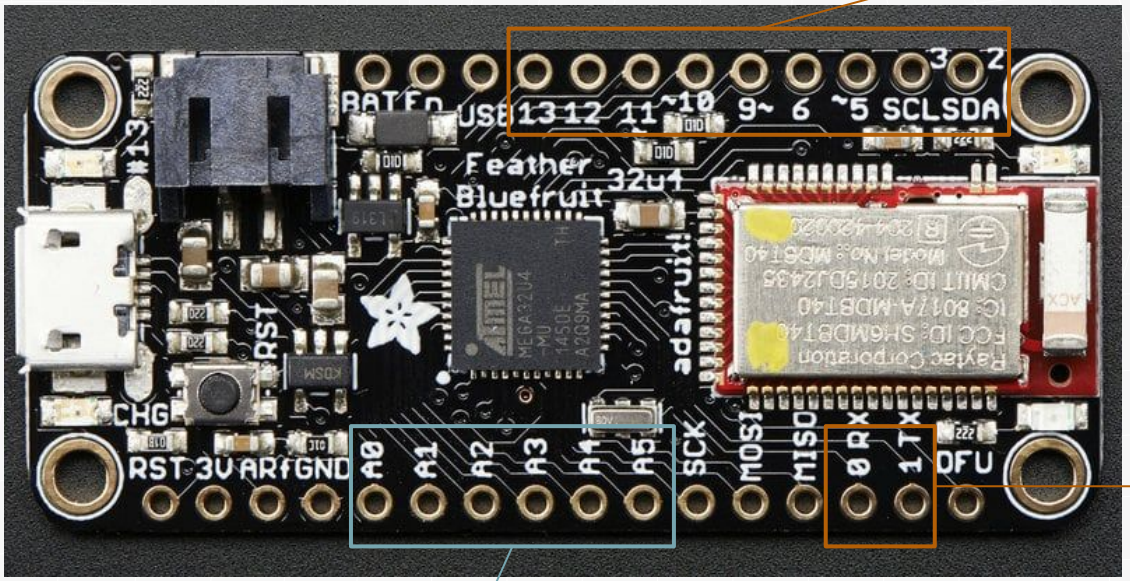
Link to Desmos Demo



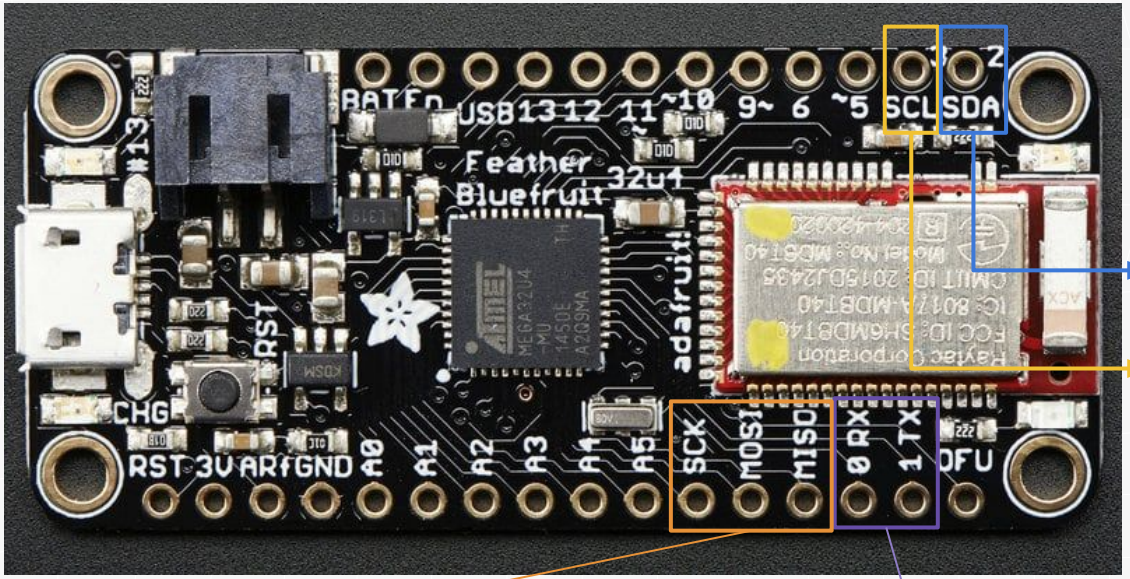
PWM and LED Brightness

Signal Pins

Notice digital pins 4, 7 and 8 are missing. They are used by the Bluetooth Peripheral so they cannot be used.



Signal Pins Cont'd



I2C Serial Data

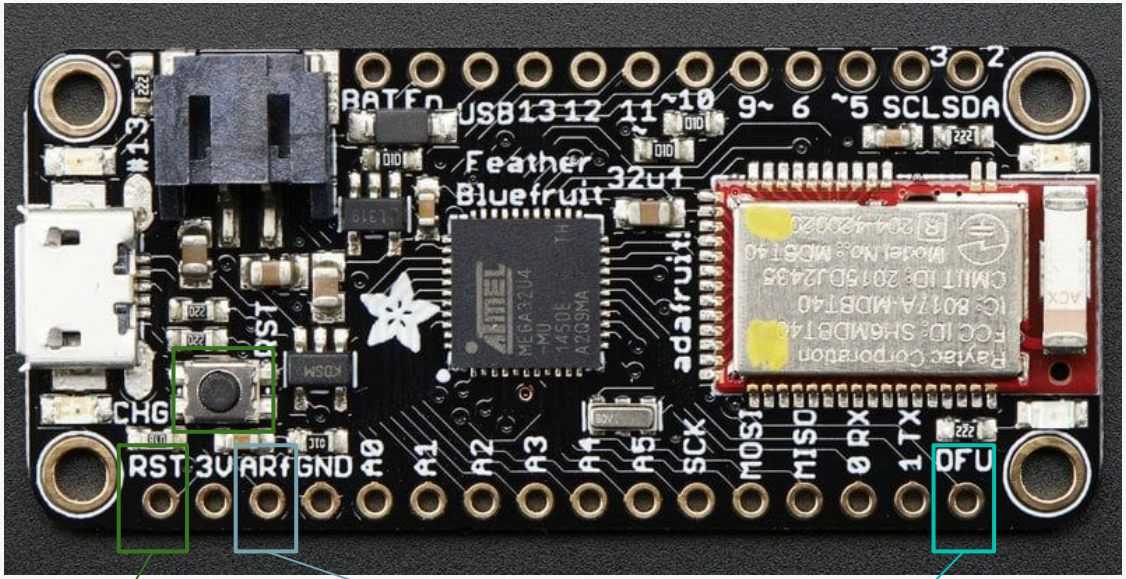
I2C Serial Clock

Optional content in activity appendices

SPI Serial Clock (SCK), Master-Out-Slave-In (MOSI) and Master-In-Slave-Out (MISO)

UART Receive (Rx) and Transmit (Tx)

Other Pins



Reset (**RST**) Pin and Button

Analog Reference (**ARef**) Pin

Device Firmware Upgrade (**DFU**)

Activity 3: Microcontroller Flashing LED

To introduce you to programming a microcontroller, we'll start by getting an LED to flash. Find the tutorial in the directory in the QR Code

Activity 4: Reading Input From Sensors

Now that you know how to write digital and analog signals, you'll learn how to read digital and analog signals using an IR sensor and light sensor.

Day 2 Complete!

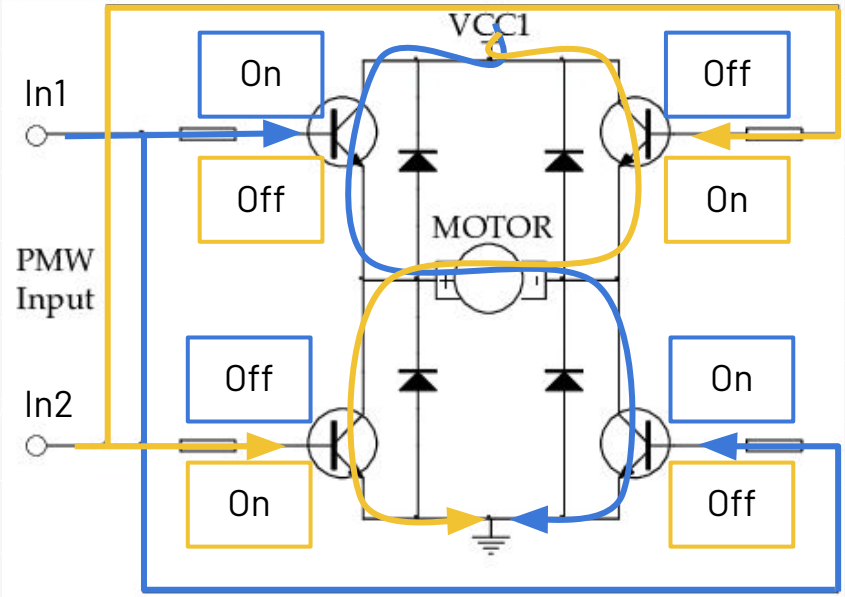
03

Robot In Motion

H-Bridges, Motorshield Library, Line Tracking

Review/Intro of H-Bridges

The H-Bridge is a circuit that allows you to safely control the speed of a motor using PWM and the direction of a motor through the transistors.

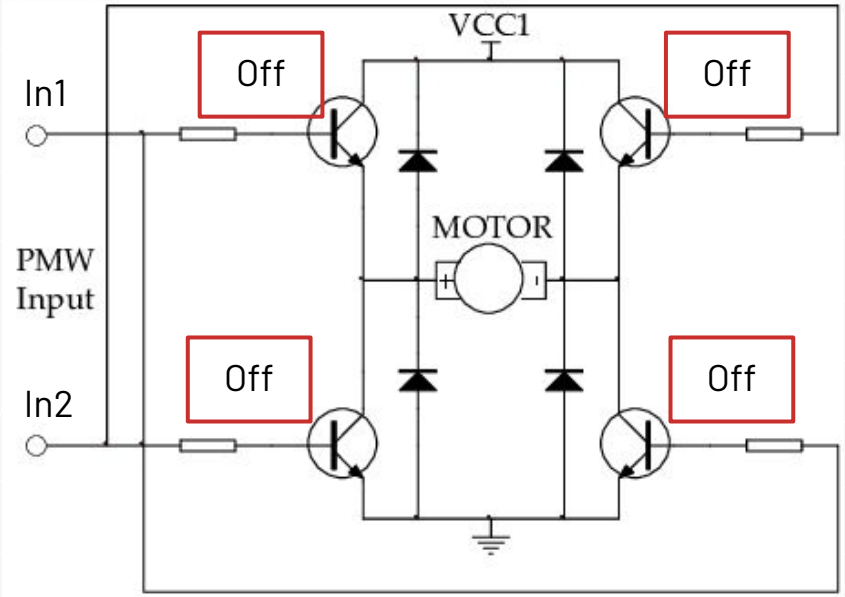


Forward

Backward

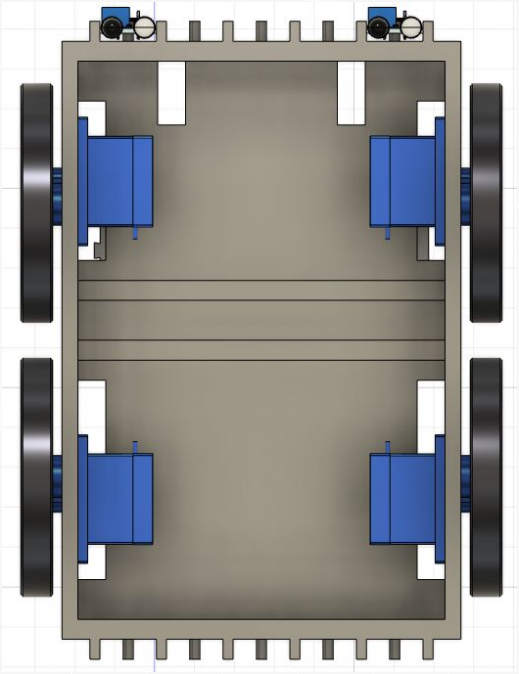
Review/Intro of H-Bridges

The H-Bridge is a circuit that allows you to safely control the speed of a motor using PWM and the direction of a motor through the transistors.



Release

PWM Pins on The Board

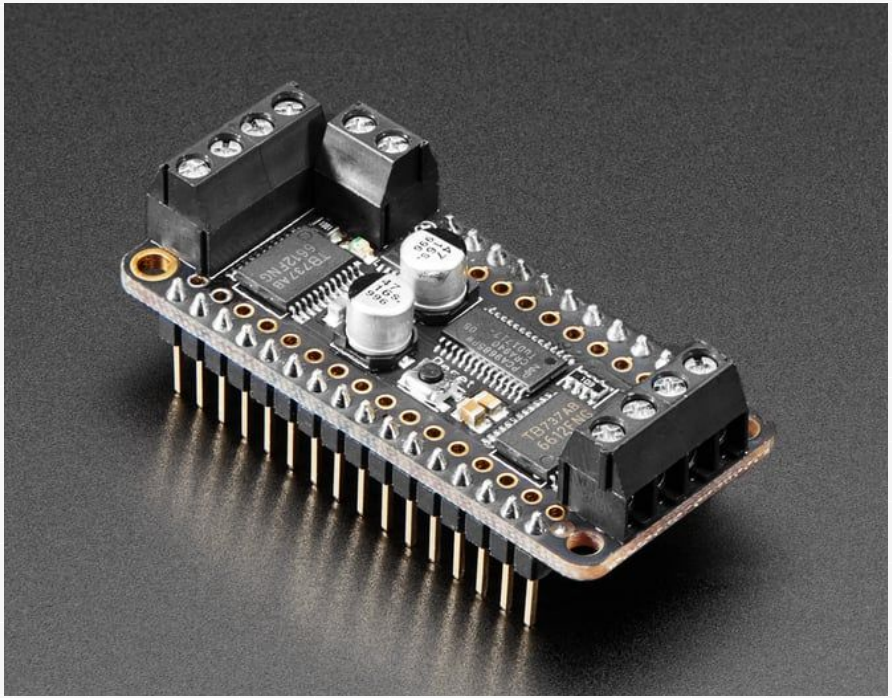


To control the speed and direction of 4 DC motors, we need **8 PWM pins**. What's the problem?

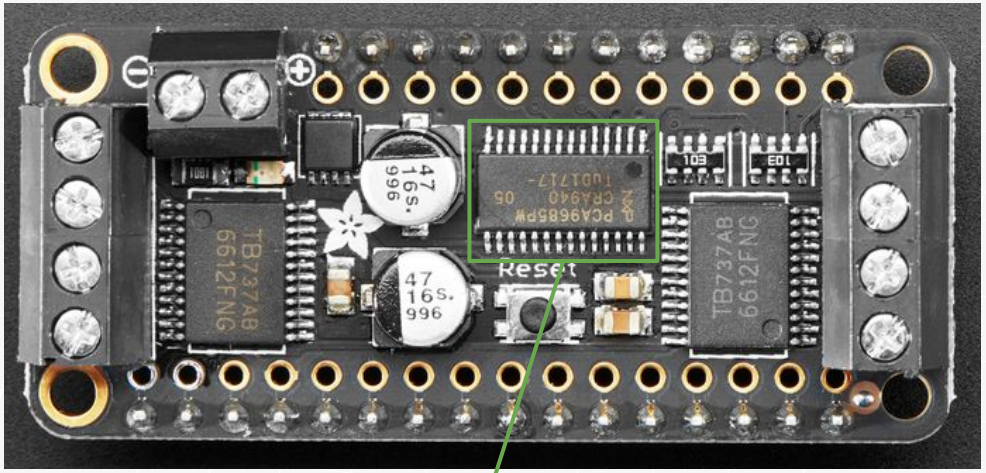
→ We only have 7 PWM pins on the board

How can we get more PWM pins?

Motorshield



PWM Driver



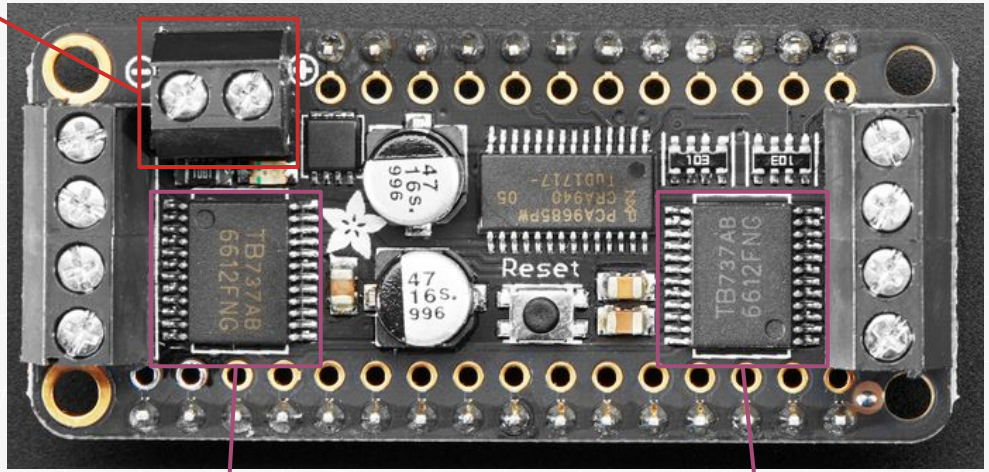
PWM Driver

- Takes 2 PWM pins and maps them to 8 PWM pins

H-Bridges and Power

Battery Port
(Note the polarity)

Motor Terminal

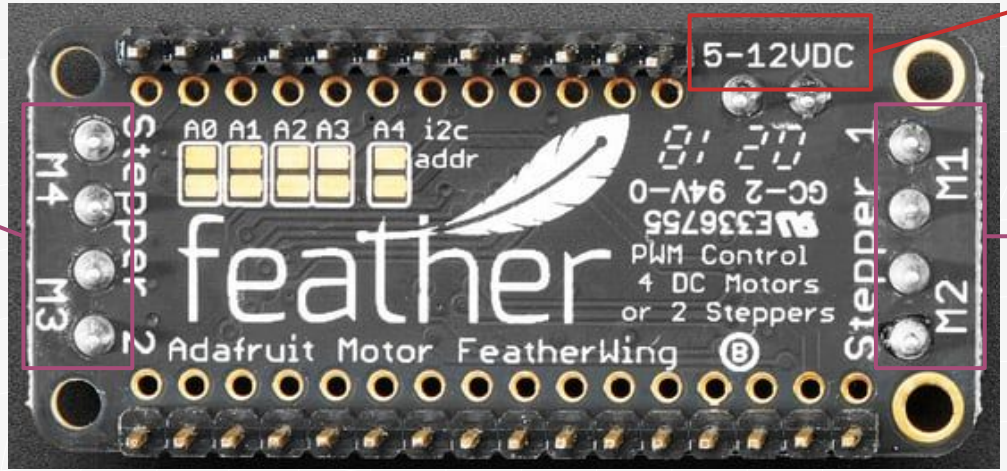


Motor Terminal

Two H-Bridges

Two H-Bridges

H-Bridges and Power Cont'd



Activity 5: Motor Control

In this activity you'll start working with the motorshield to get a motor spinning with a wheel on it. QR Code for the worksheet directory is provided.

Motor Control Code

```
#include <Adafruit_MotorShield.h>

#define MOTOR_TERMINAL 1

Adafruit_MotorShield AFMS = Adafruit_MotorShield();

Adafruit_DCMotor *MOTOR = AFMS.getMotor(MOTOR_TERMINAL);

void setup() {
  // Start up motorshield
  AFMS.begin();

  // Reset motor
  MOTOR->setSpeed(0);
  MOTOR->run(RELEASE);
}

void loop() {
  MOTOR->setSpeed(150);
  MOTOR->run(FORWARD);
  delay(3000);
  MOTOR->run(BACKWARD);
  delay(3000);
  MOTOR->run(RELEASE);
  delay(3000);
}
```

Motor Control

BREAK

Be back in 20 minutes

Challenge 1: Basic Line Tracking

Now you have all the tools and components to build your robot and program it to autonomously follow a black line to a destination:

- Program the light sensor to activate your robot with your phone flashlight
- Program the motors to move the robot
- Program the IR sensors to detect black lines and adjust accordingly

How to Line Track

Suggestion 1:

- Drive along the line as long as you detect white on the sides of the robot
- Turn left/right based on which side detects black

Suggestion 2:

- Drive along the line as long as you detect black on the sides of the robot
- Turn left/right on which side detects white

Day 3 Complete!

04

Designing Your Own Robot

Microcontroller Protection, Challenge Description, Open Designing

What is 3D Printing

3D printing is a type of **additive manufacturing** that builds a model layer by layer to form the final shape.

There are two popular types of 3D printing:

- Fused Deposition Modelling (FDM) or Fused Filament Fabrication (FFF)
- Stereolithography (SLA)

FDM Printing

SLA Printing

Slicers

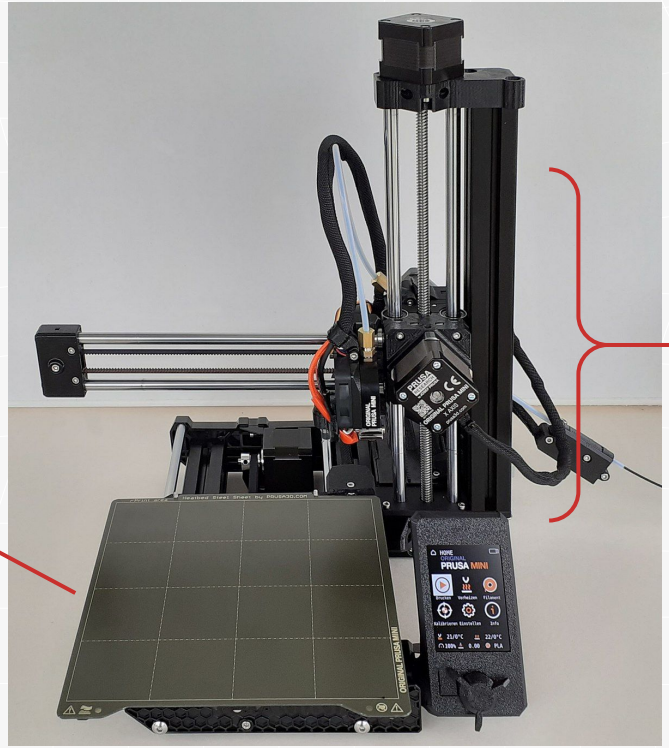
A **slicer** is a piece of software that breaks down your 3D models into the layers of your print where you can preview the entire printing process. If you'd like, you can download the slicer yourself and preview your prints.

Considerations of FDM Printing

For this workshop, you will be designing a drivebase for FMD printing. This comes with three major considerations that you'll need to design around:

1. Bed Size
2. Bed Adhesion
3. Support Placement

Bed Size

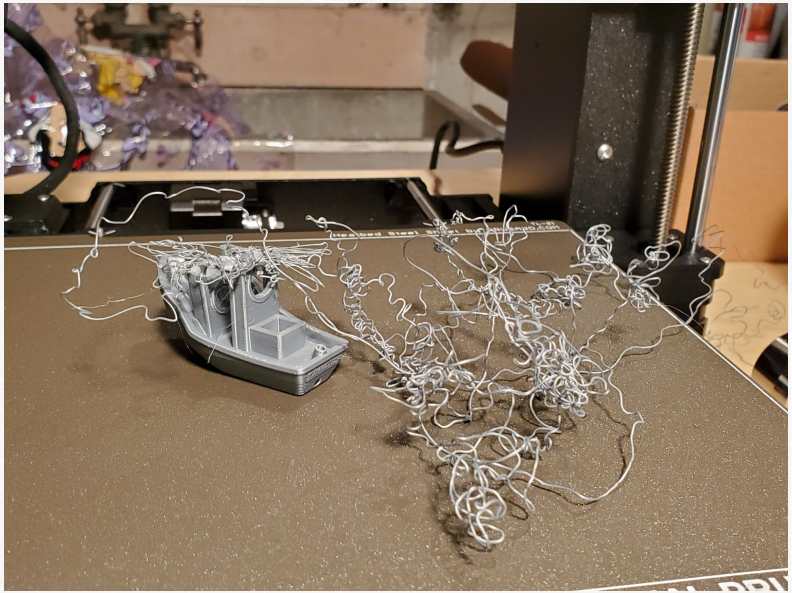


17cm printing height

17x17cm bed

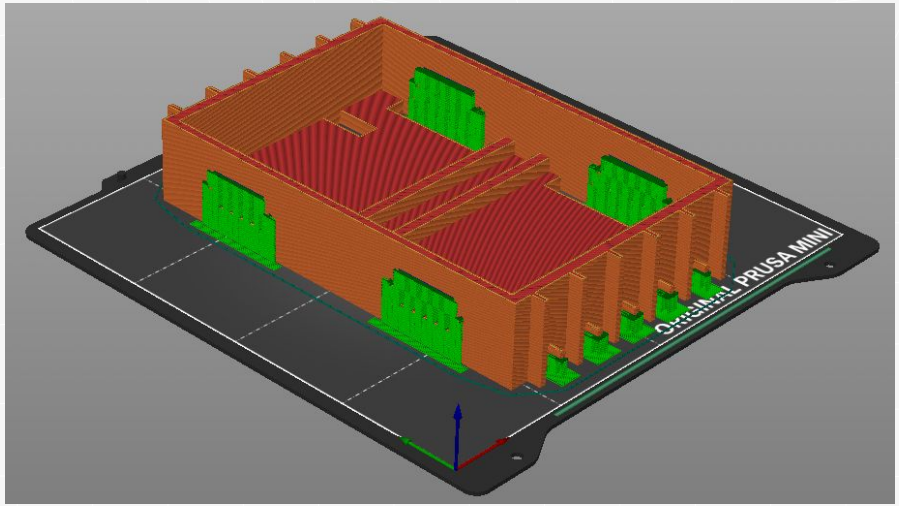
Bed Adhesion

If your part does not stick to the bed of the printer, it may fly off mid-print and become a spaghetti-monster



Support Placement

Supports are extra 3D printed extrudes that are necessary for elevated features to be printed properly. Recall that FDM works layer by layer, so an elevated feature needs something below it to be printed onto.



DESIGN TIME