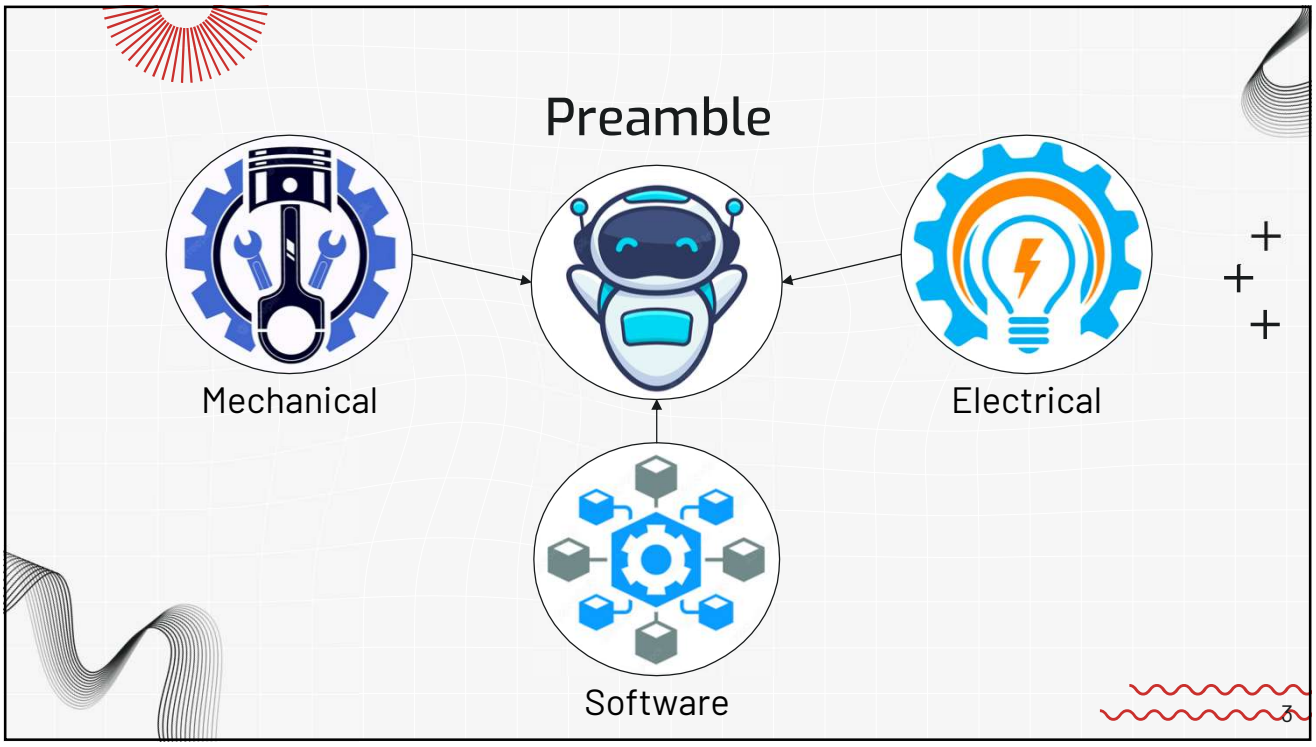
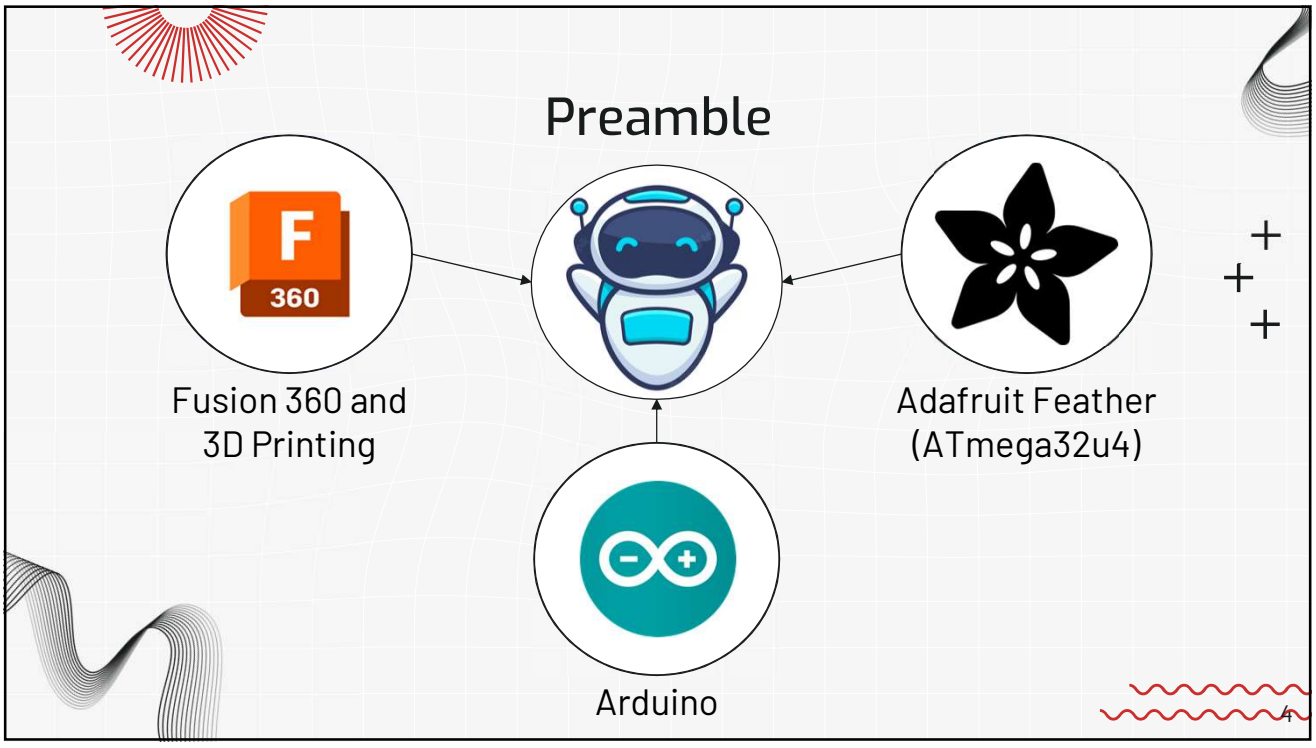
ECE Robotics and Microcontroller Workshop

1

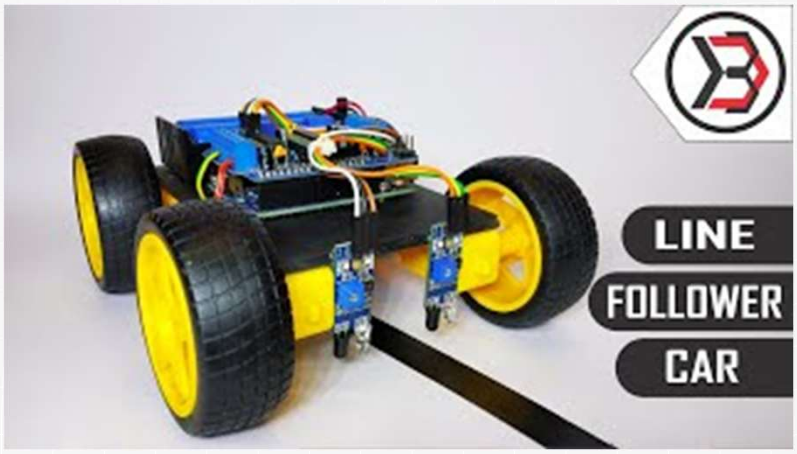
The Google Drive

2


Preamble
Mechanical
Electrical
Software
3


Preamble
F
360
Fusion 360 and
3D Printing
Adafruit Feather
(ATmega32u4)
Arduino
4

Workshop Objective



5

Workshop Roadmap

Day 1
Learning CAD, assembling the motors and wheels

Day 2
Learning microcontrollers and general signal I/O

Day 3/maybe 4
Basic Line tracking and designing your own robot

Day 5-7
Preparing for the Final Race

6

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 more complex maze
- Fastest team wins a prize (TBD)

7

Activity and Challenge Sheets

Phase 1: Guided Learning (Day 1-3)

- Activity 1: Learning how to CAD
- Activity 2-4: Programming the microcontroller
- Challenge 1: Basic line tracking challenge

Phase 2: Open Designing (Day 4-8)

- Activity 5: Designing your own robot CAD
- Activity 6: Bluetooth manual control of robot
- Challenge 2: Advanced line tracking challenge

8

Final Notes Before We Begin

- Lots of content at a high level
- If you ever feel lost ask questions earlier rather than later
- Workshop will support concepts for many 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

9

01

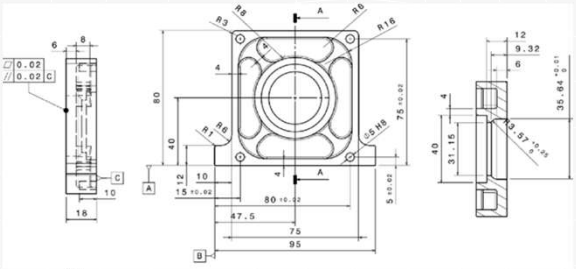
CAD Modelling and Fusion 360

10

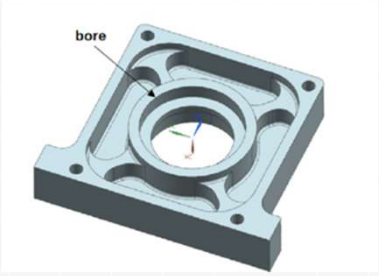
What is CAD Modelling?

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

11

Design and Manufacturing Cycle

CAD Design

CAD Each Part/Component

Build a CAD Assembly

Manufacture Design

Manufacture Each Part

Physically Assemble

12

Fusion 360

What is Fusion 360 & its functions
- Software that allows you to create detailed 3D models, it can be used anything from mechanical parts to art
- Helps bring 2D designs to life
- Simulate how things will work and prepare them for manufacturing

13

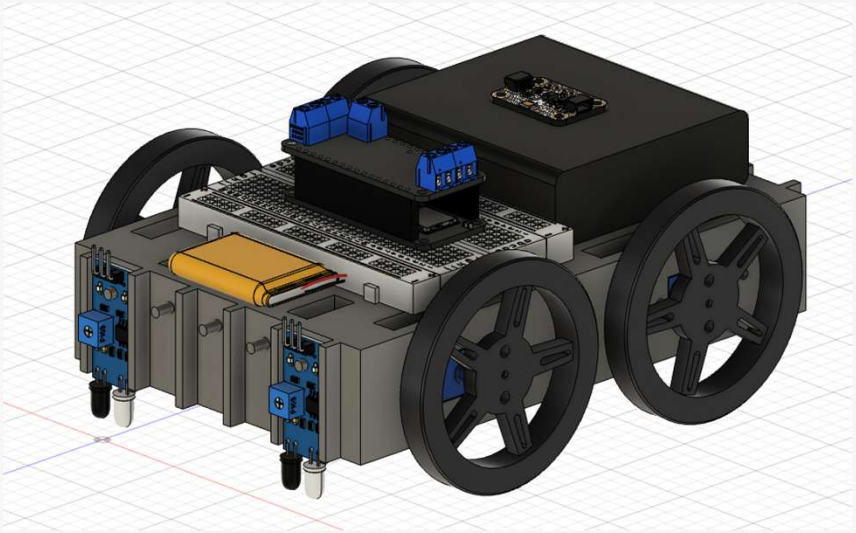
Benefits of CAD

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

14

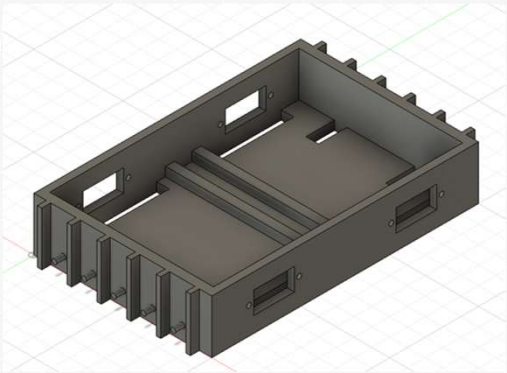
What Are You Designing?



15

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.



16

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

17

Day 1 Complete!

18

02

Microcontrollers

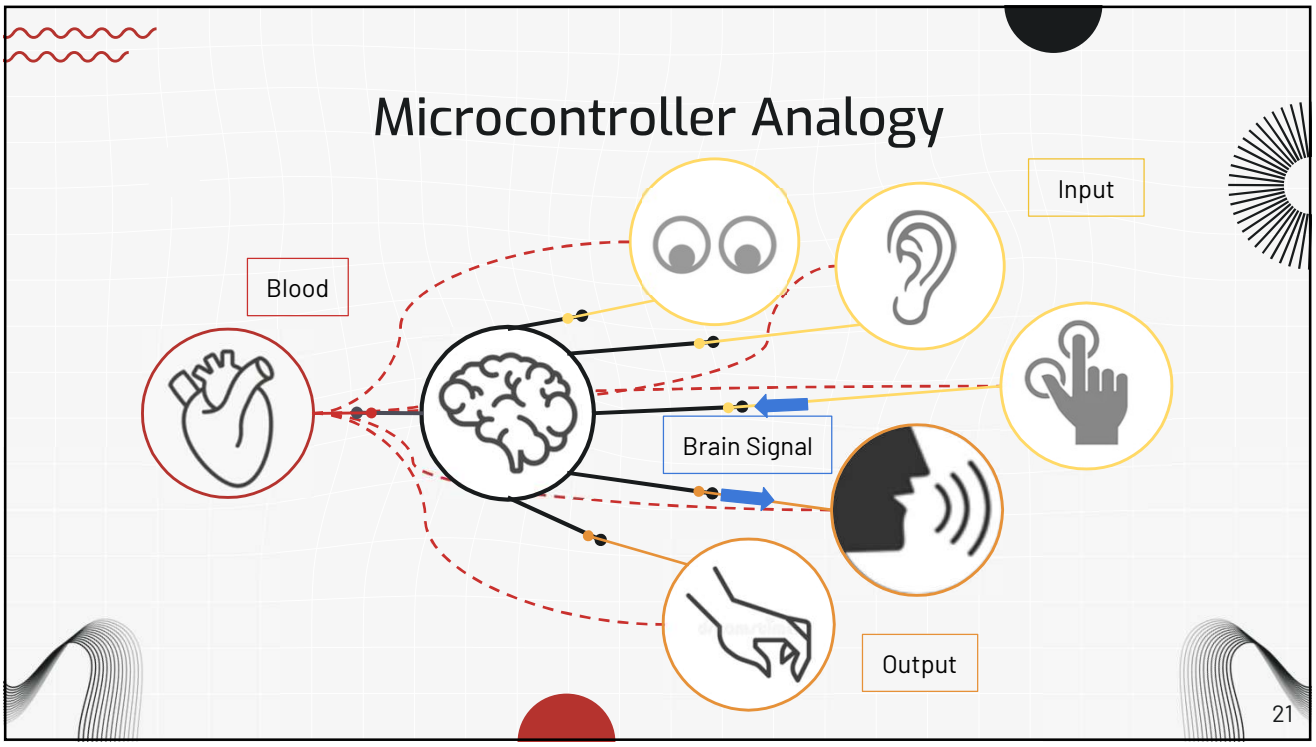
Processors, Pinouts, and Digital Signals

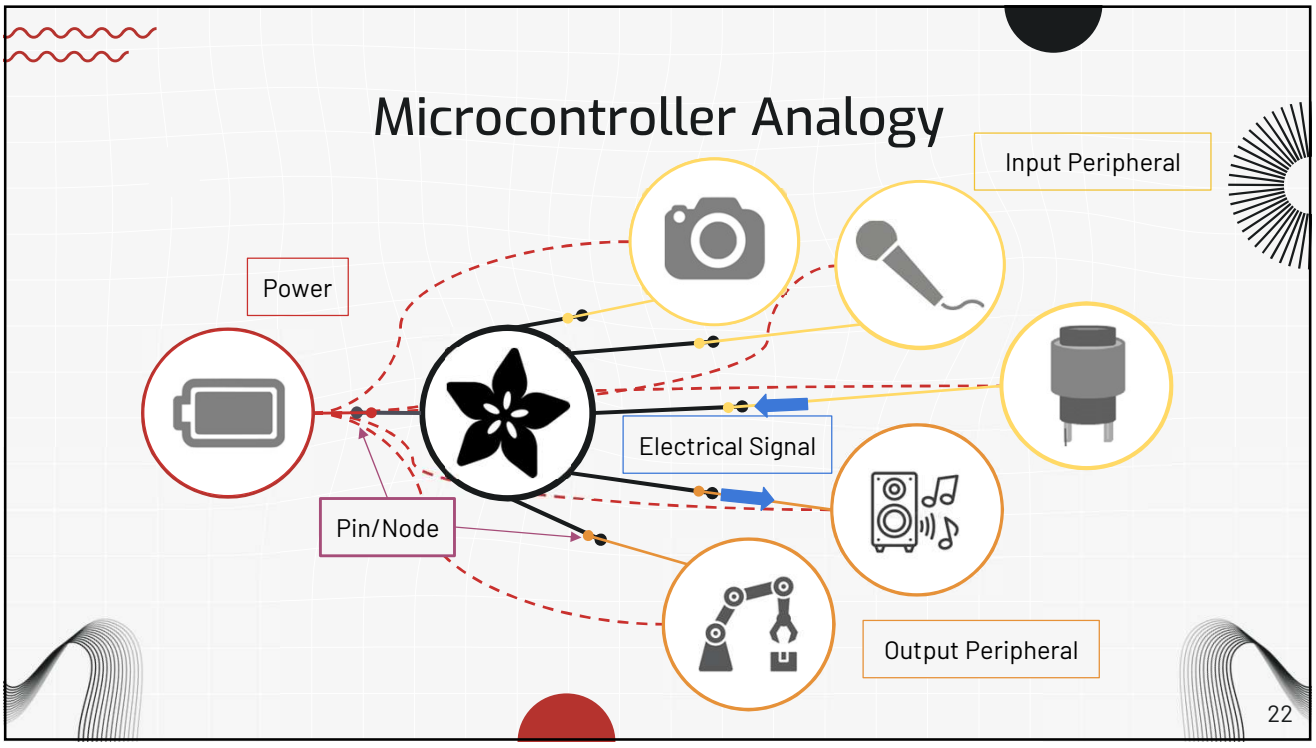
What is a Microcontroller

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

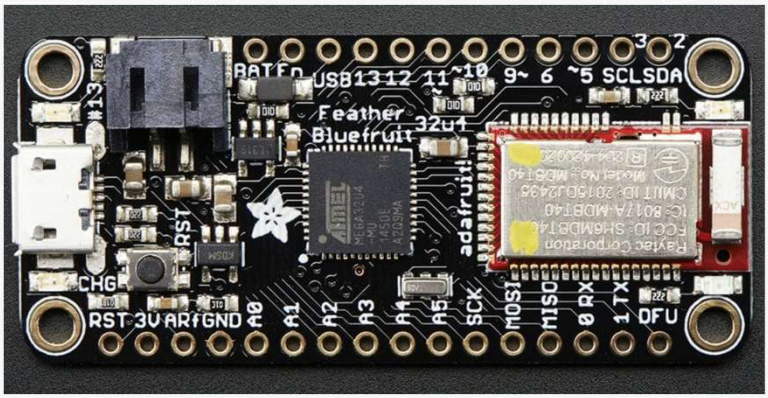
- Ex: turn on an LED, spin a motor, monitor the ambient light in a room

Manually adjust function/signal generator to control a circuit → Program a microcontroller to generate the signal and control a circuit

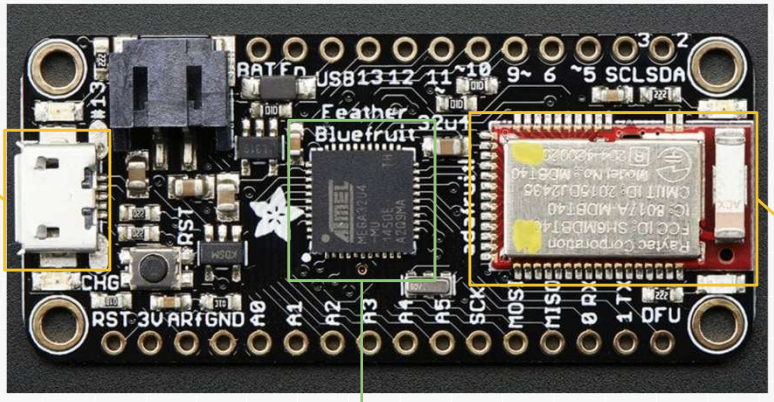

Microcontroller Analogy
Input
Blood
Brain Signal
Output
21


Microcontroller Analogy
Input Peripheral
Power
Electrical Signal
Pin/Node
Output Peripheral
22

The Microcontroller You Will Be Using



Processor and Internal Peripherals



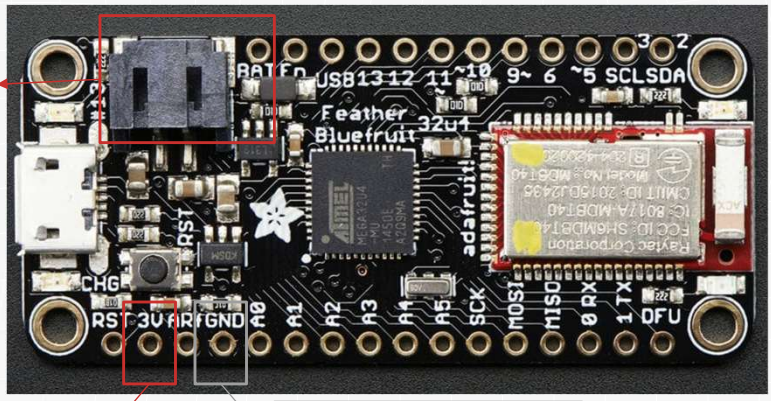
USB Peripheral

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

25

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**

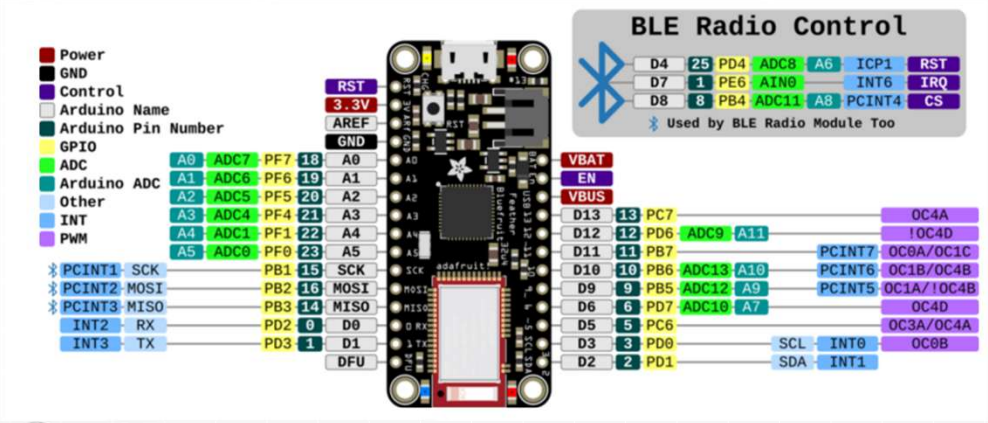
An analog signal is a continuous/real signal like the signals you've worked with up to this point in electrical courses

A digital signal is a zero or a one (high/low).

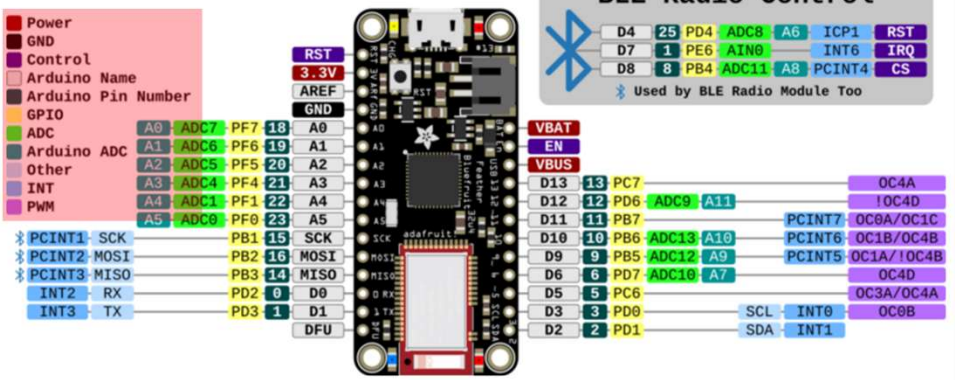
For the processor to understand analog signals, they are broken down into discrete values and transmitted as a **pulse/serially** (ie. strings of bits)

26

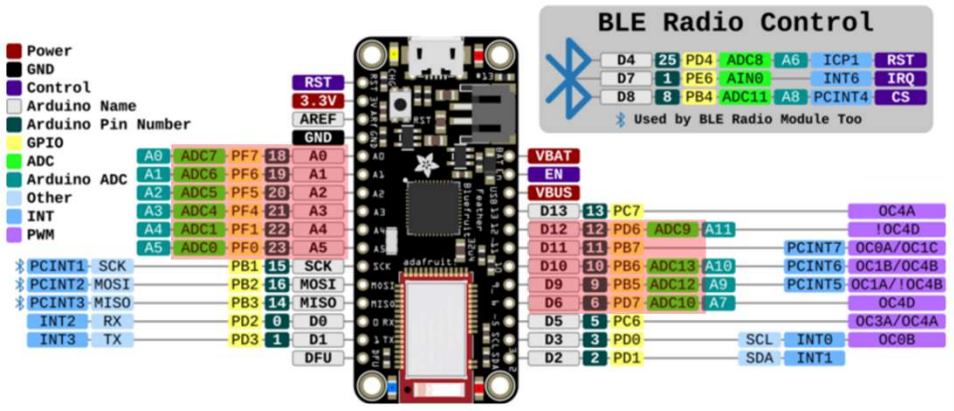
Variable Signal Pins



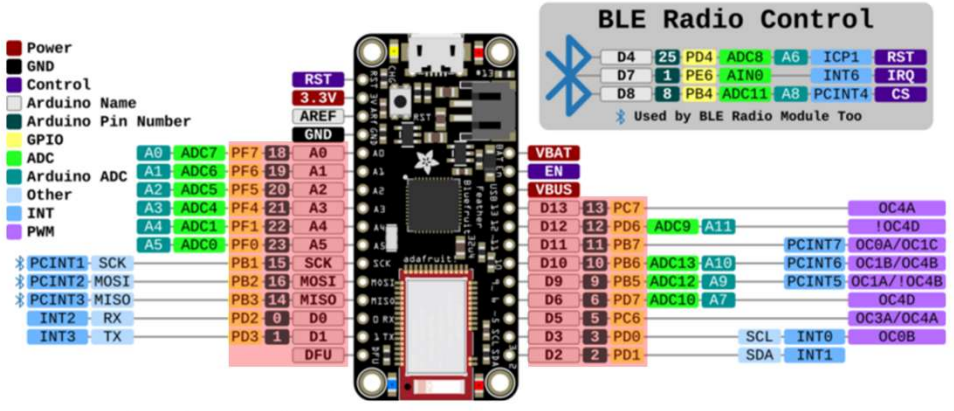
Variable Signal Pins



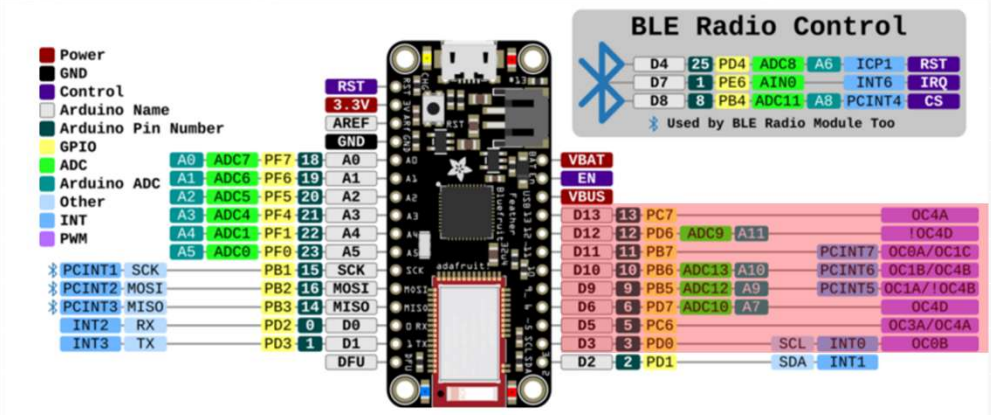
Variable Signal Pins



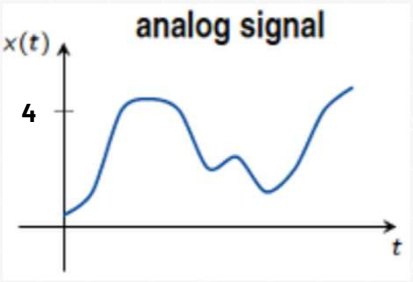
Variable Signal Pins



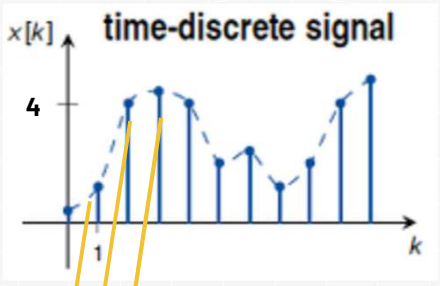
Variable Signal Pins



Graphical Interpretation



Discretization



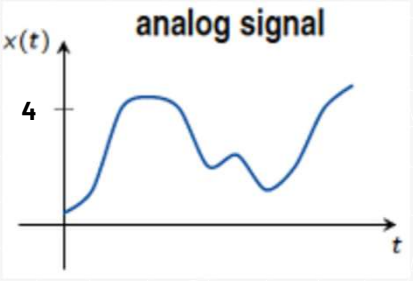
We're **sampling** the function

Form String

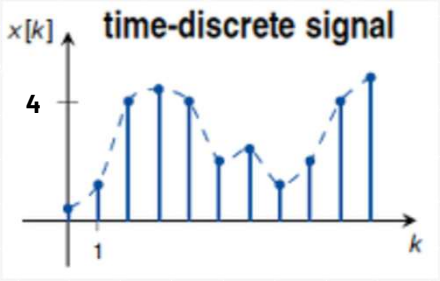
Successfully broken down an analog signal

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

Graphical Interpretation



Discretization

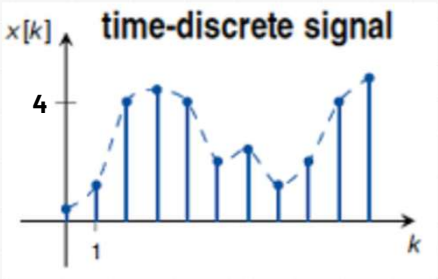


Form String

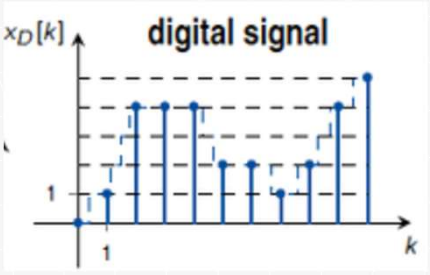
x[k] = [1, 4, 4.2, 4, 2, 2.3, 1, 2, 4, 4.6]

33

Graphical Interpretation



Round



Rounding to **quantized** levels

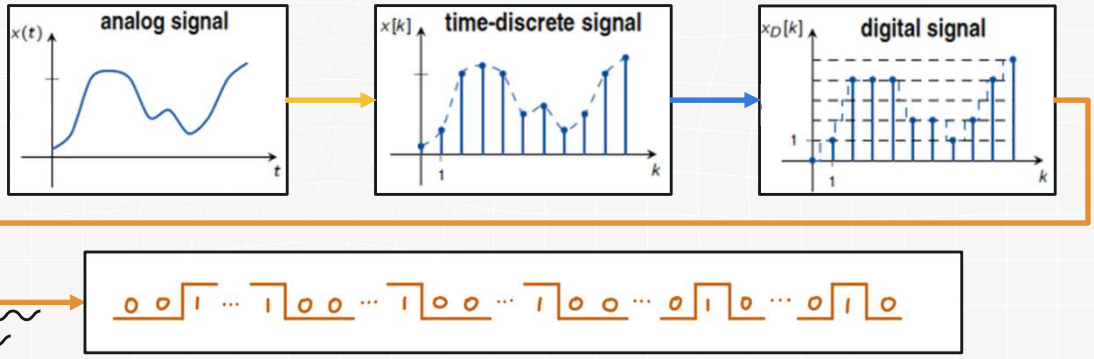
Form String

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

34

Summary of Digital Signals

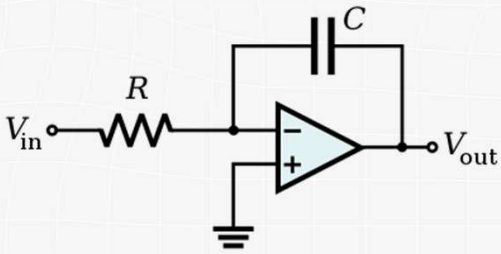
Digital signals are analog signals that have been **sampled** and **quantized** into discrete, integer values that can be encoded into a **pulse wave** for **serial communication**



35

Why Are Digital Circuits Useful?

The reason is because analog circuits are not easy to modify, they perform only one function. Digital computers are very easy to modify and integrate with code. You'll learn more about the math behind the signal processing in **ECE216**.



Op Amp Integrator
(Cannot be used for anything else)

36

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

37

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

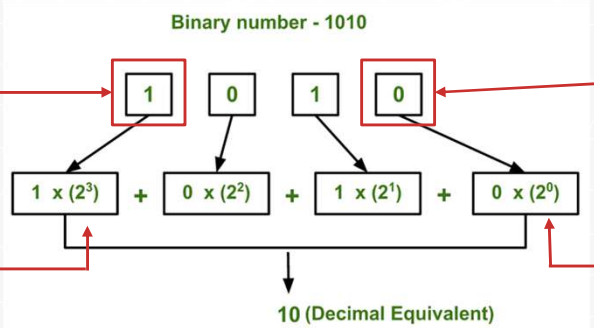
Decimal	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

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

38

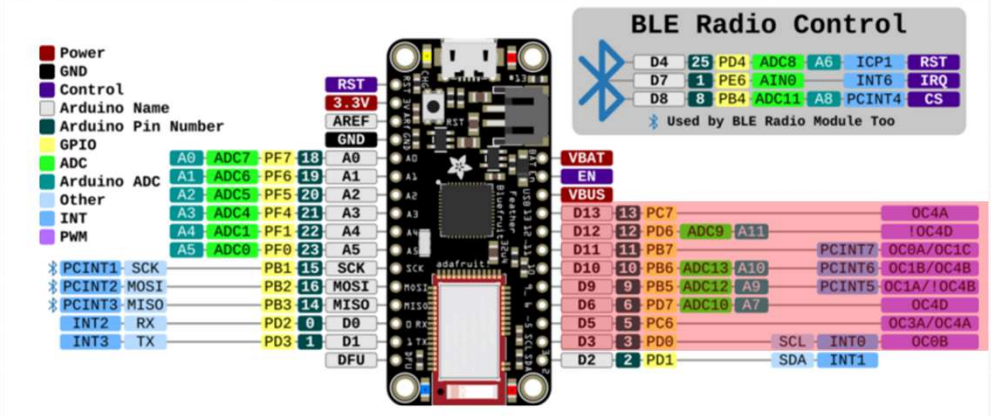
Binary-to-Decimal Conversion

Most Significant Bit
- Largest influence on magnitude



Least Significant Bit
- Smallest influence on magnitude

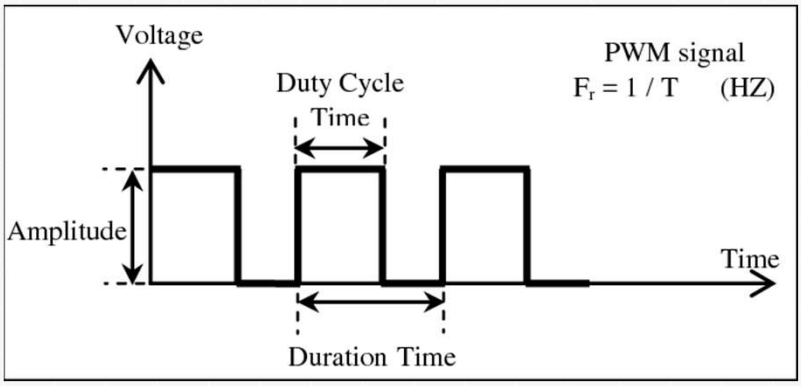
Variable Signal Pins



Pulse Width Modulation (PWM)

A pulse is a signal that switches between HIGH and LOW

- Pulses with a frequency can undergo PWM

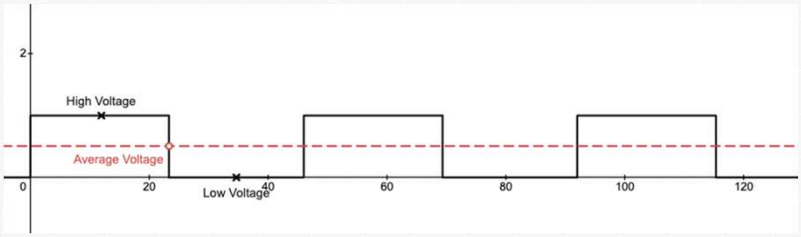


41

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



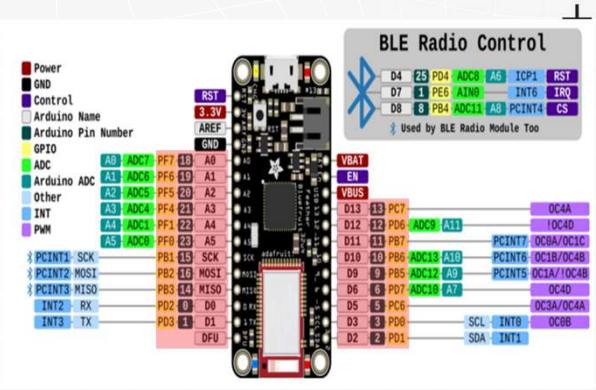
42

GPIO Pins

GPIO: General Purpose Input/Output

Can be used to Input and Output digital signals

- digitalWrite(pinName, HIGH/LOW)
- digitalRead(pinName)



43

Analog Pins

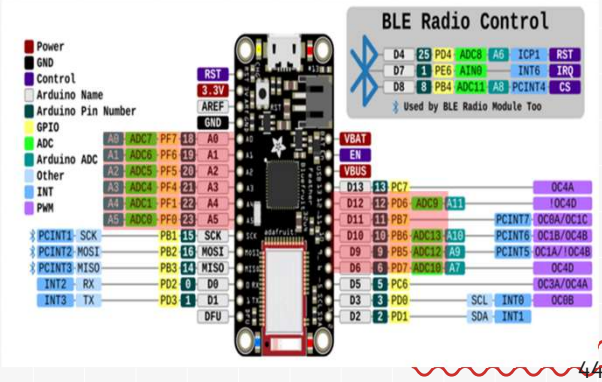
Used to read analog signals. They have a built-in ADC to convert to digital signals. The analog signal read is converted to a 10-bit value (between 0 and 1023) depending on the voltage level.

Conversion System:

- 0V -> 0
- >= 3.3V -> 1023

Programming Syntax:

- analogRead(pinName)



44

PWM pins

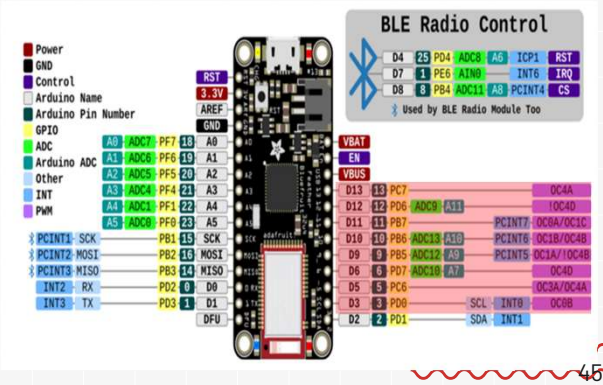
Used to produce an analog output.

The value entered into this function is between 0 and 255 (8 bits).
This value controls the duty cycle.

Duty cycle = value/255

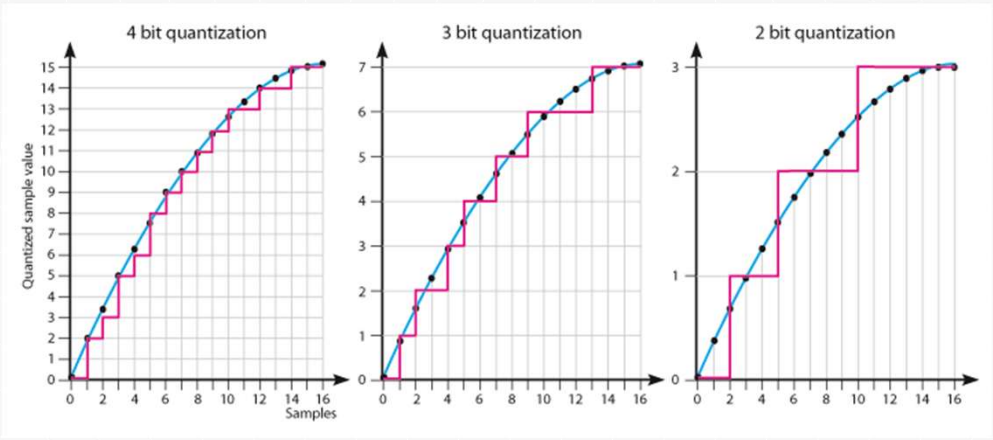
Programming Syntax:

- analogWrite(pinName, value) →



45

Analog Signal Chopping (Discretization)

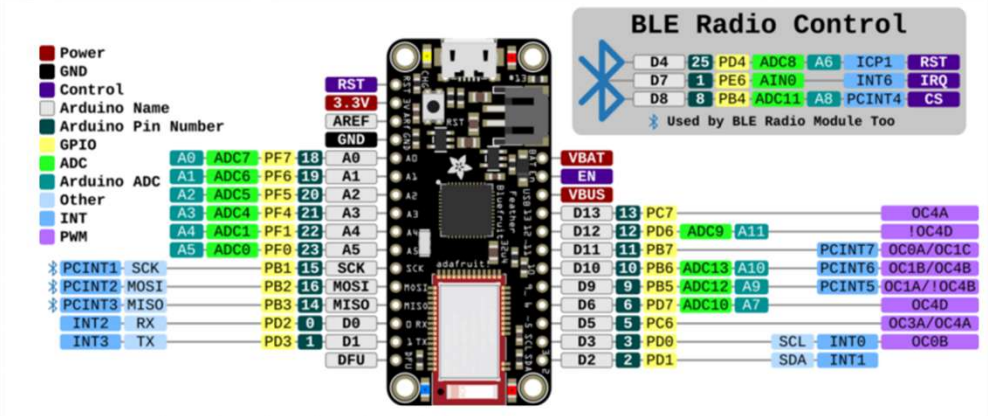


Larger bit length (n) means more precise digitization

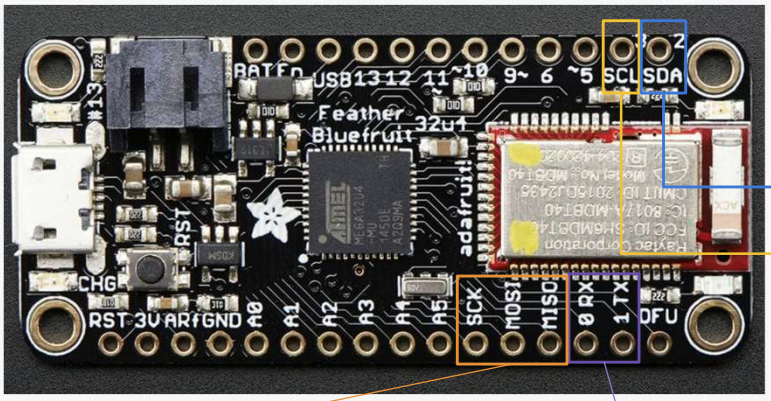
46

PWM and LED Brightness

Variable Signal Pins



Signal Pins Cont'd



I2C Serial Data

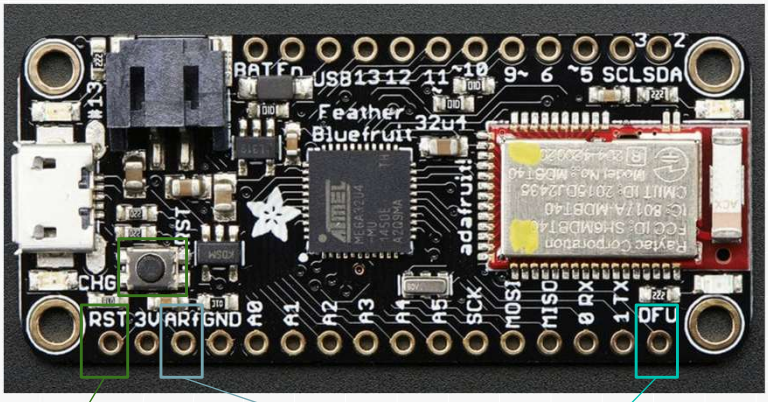
I2C Serial Clock

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 2: Microcontroller Coding

. Follow the handout, it will give you a good introduction to programming microcontrollers using the Arduino IDE

51

Activity 3: 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.

52

03

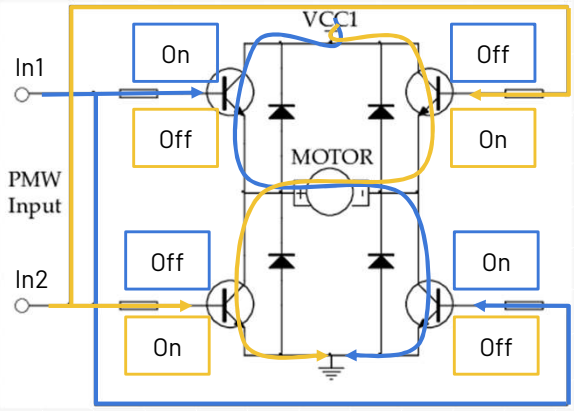
Robot In Motion

H-Bridges, Motorshield Library, Line Tracking

53

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.



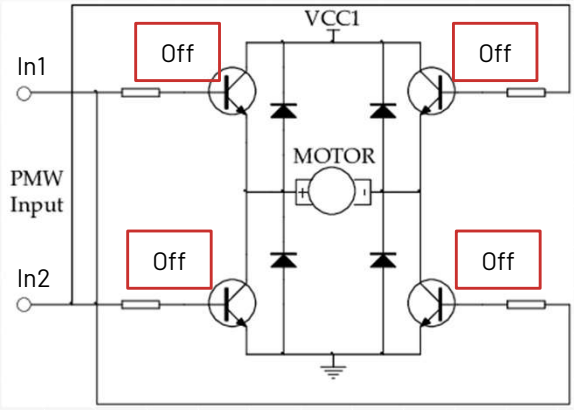
Forward

Backward

54

Review/Intro of H-Bridges

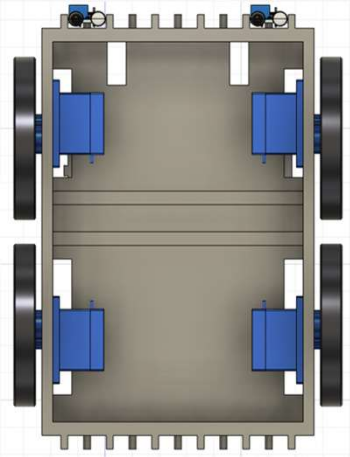
The H-Bridge is a circuit that allows you to safely control the speed of a motor using PWM and the direction of a motor through the transistors.



Release

55

PWM Pins on The Board



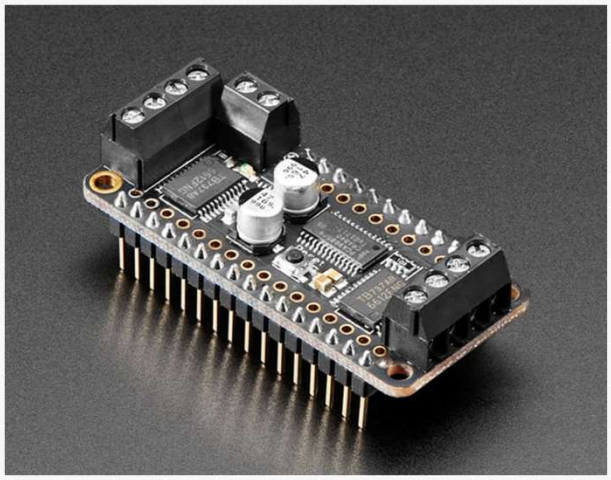
To control the speed and direction of 4 DC motors, we need **8 PWM pins**. What's the problem?

→ We only have 7 PWM pins on the board

How can we get more PWM pins?

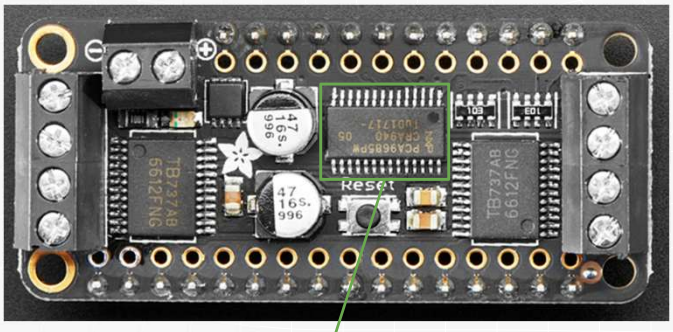
56

Motorshield



57

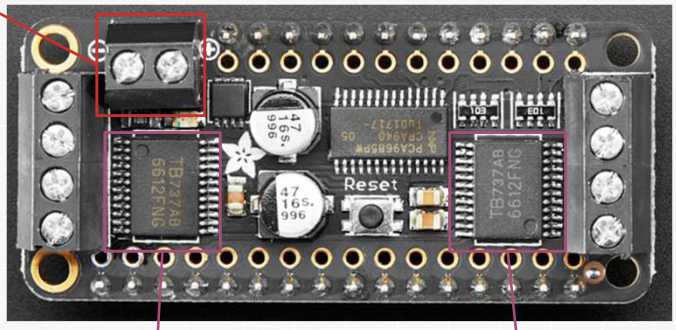
PWM Driver



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

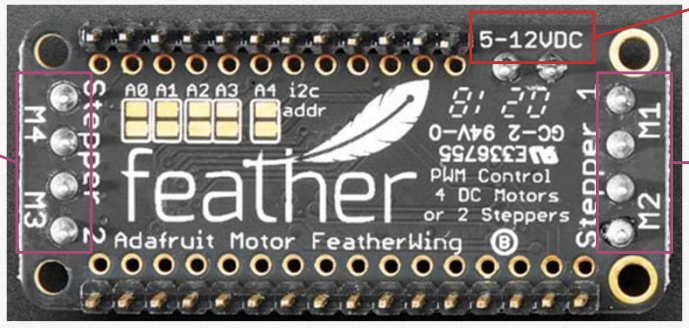
58

H-Bridges and Power



59

H-Bridges and Power Cont'd



60

Activity 4: Motor Control

In this activity you'll start working with the motorshield to get a motor spinning with a wheel on it.

61

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);
}
```

62

Motor Control

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
  - When the sensor detects black → the output is 1 (HIGH)
  - When the sensor detects white → the output is 0 (LOW)

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

65

04

Designing Your Own Robot

Microcontroller Protection, Challenge Description, Open Designing

66

What is 3D Printing

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

We will be using a method called **Fused Deposition Modelling (FDM)** or **Fused Filament Fabrication (FFF):**

- It uses thermoplastic filaments, which are melted and extruded layer by layer to build an object

67

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.

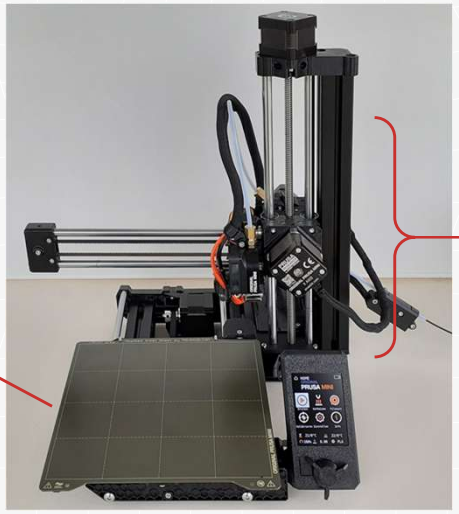
68

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

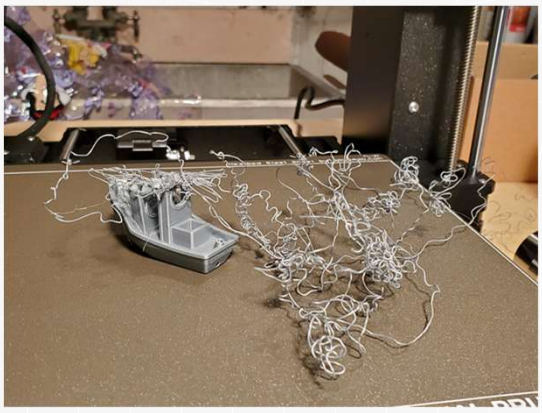


17x17cm bed

17cm printing height

Bed Adhesion

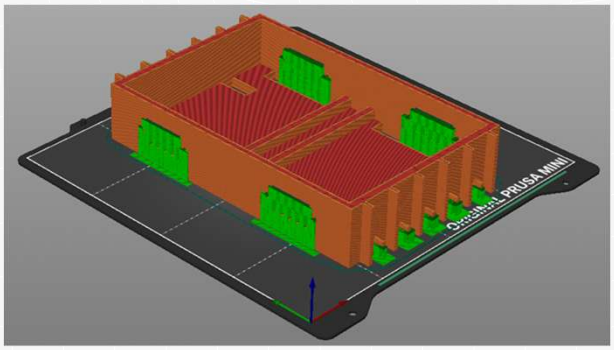
If your part does not stick to the bed of the printer, it may fly off mid-print and become a spaghetti-monster



71

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.



72

DESIGN TIME

Bluetooth

Communication Protocols

- SPI is used between the MCU and the bluetooth module (on the board)
  - SPI: Serial Peripheral Interface
- The phone communicates with the bluetooth module using UART
  - UART: Universal Asynchronous Receiver-Transmitter

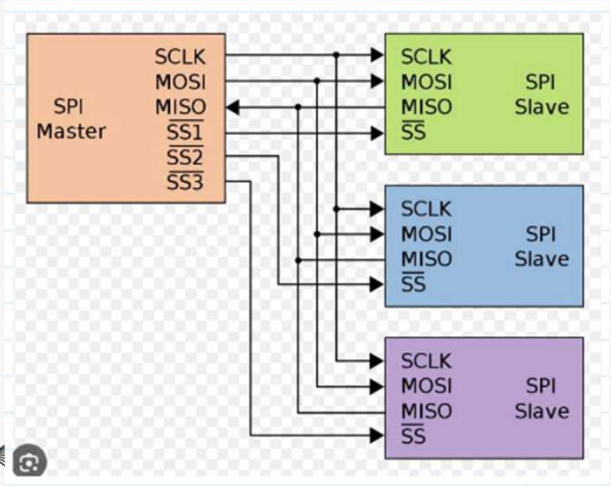
75

SPI Protocol

- Used for communication between microcontrollers and peripherals (sensors, memory devices, displays)
- Master and Slave set-up
  - Master initiates communication, slave responds
- Full-Duplex Communication: Data can be transmitted and received simultaneously

76

SPI Protocol



- MOSI: Master Out Slave In
- MISO: Master In Slave Out
- SCK: Serial Clock
- SS: Slave select

77

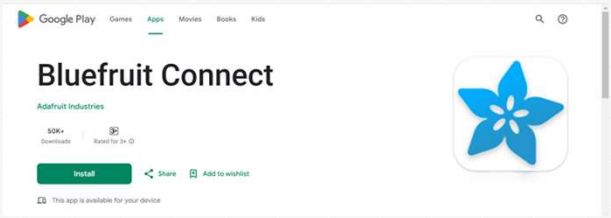
UART Protocol

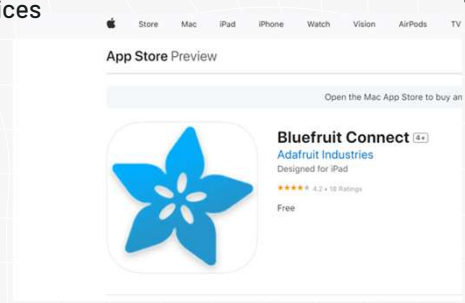
- The Bluefruit Connect app only uses UART as the transport for sending and receiving data to/from your BLE device
- UART communication involves sending data in packets, which consist of a start bit, data bits, optional parity bit (error checking), and stop bits
- UART operates without a clock signal, meaning the sender and receiver must agree on the baud rate (speed of data transmission) beforehand.
- Baud rate examples include 9600, 19200, 38400, etc.

78

App

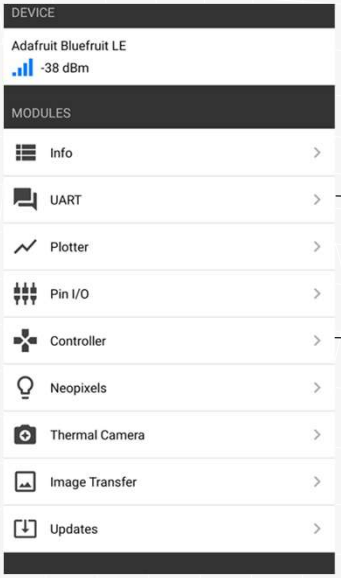
- Facilitates communication with Bluefruit LE devices
- Provides tools for:
  - Basic communication and info reporting
  - Arduino Pin Control





79

App

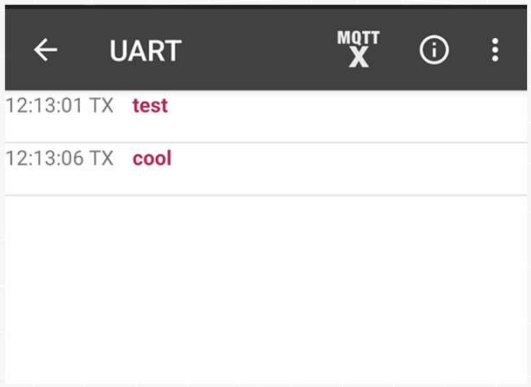


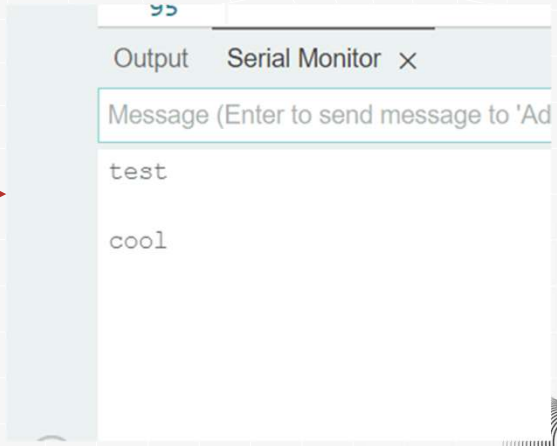
'serial terminal' interface for sending and receiving strings

Control Pad layout

80

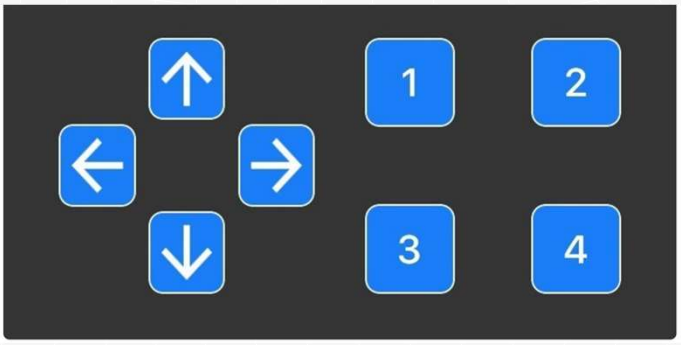
UART





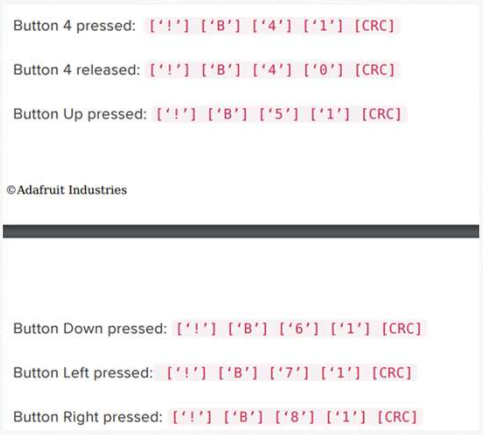
81

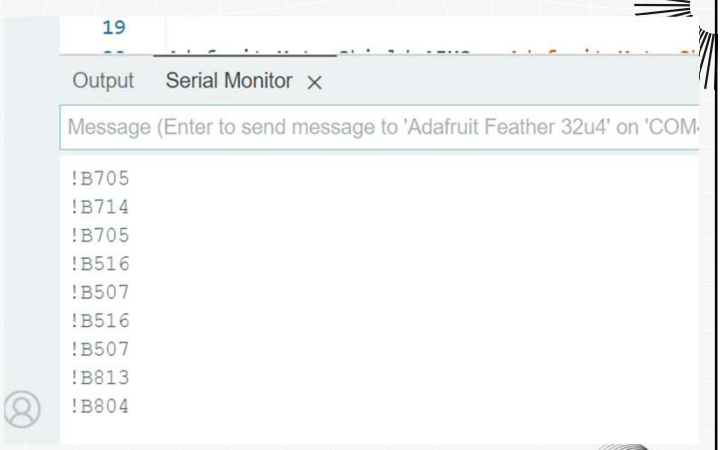
Controller



82

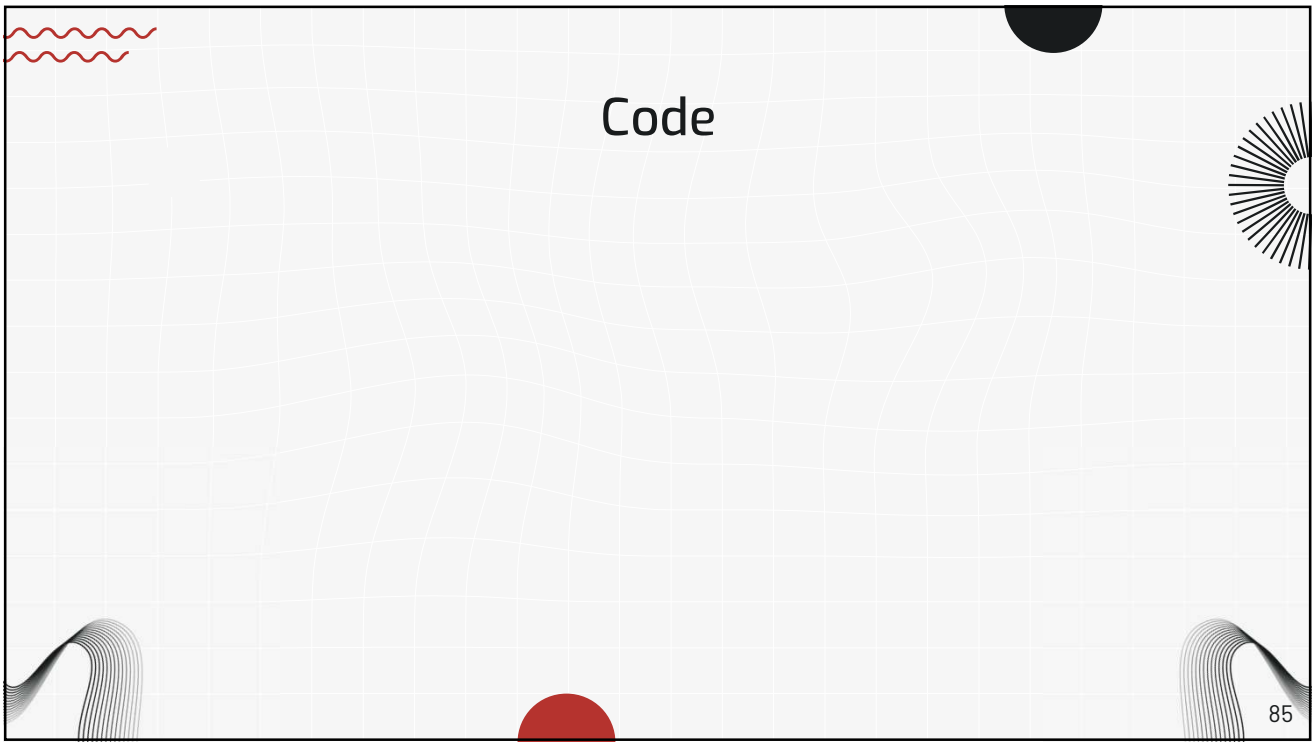
Controller





Libraries

```
#include "Adafruit_BluefruitLE_SPI.h"
```


Code
85