Schedule

Tuesday (7/18)

8:30 AM – 10:00 AM: Challenge Introduction
10:00 AM – 11:30 AM: Rover Electrical Layout
11:30 AM – 1:00 PM: Tour and Lunch
1:00 PM – 2:00 PM: Mechanical Design Introduction

Wednesday (7/19)

8:30 AM – 12:30 PM: Mechanical Design using OnShape
12:30 PM – 1:15 PM: Lunch
1:15 PM – 2:00 PM: Software Introduction

Thursday (7/20)

8:30 AM – 11:30 AM: Electromechanical Integration
9:00 AM: Descent Cave Spectral Analysis Electrical Session*
9:30 AM: Probe Button Jumpstart Mechanical Session*
10:15 AM: Sample Collection Mechanical Session*
11:30 AM – 12:30 PM: Lunch
12:30 PM – 2:00 PM: Rover & Controller Programming
12:45 PM: Probe Button Jumpstart Software Session*
1:00 PM: Uplink with Satellite Software Session*

Friday (7/21)

8:30 AM – 11:00 AM: Finish Cars
11:00 AM – 12:00 PM: Lunch and Garmin Opportunities
12:00 PM – 3:00 PM: Finish Cars
3:00 PM – 5:00 PM: Challenge Test!

* These sessions are challenge specific. Only those planning on completing the challenge need to attend.

Project G-Rover

The year is 2064, and Garmin's great success has led them to branch out into planet exploration! We've partnered with NASA to investigate Pluto and need your help. We need to design a set of rovers to explore the surface and return as much data as possible. Due to the cold surface of the distant (dwarf) planet, we don't expect our equipment to survive very long, so making the most of our time is of utmost importance!

Rules

- Teams of 4 – assigned to you on the first day
- Every teammate will design, build, and control their own rover
- You will have 3 minutes for a final mission run
- Each team is allowed 2 mission runs

Challenges

There are 5 major targets for exploration. Each will require special design and skills from different rovers and involve different terrain and technology. These will each result in samples released around the area of investigation that we will want to collect for further research. In total, there are 6 missions we can attempt to complete.

Each challenge will have two levels of difficulty. Completing the **easy level will release 1 sample**, and completing the **hard level will release 1 bonus sample**.

1. Uplink with Satellite
    - Once we land, we'll need to establish a connection to our satellite above Pluto. If we can connect, we'll be able to communicate more effectively and identify another set of sample targets. However, some of the data may be corrupted and will need to be de-scrambled.
        - Easy: Establish connection (transmit data packet)
        - Hard: Record response signal and de-corrupt (read and respond)
        - Engineering skills: IR (Infrared) communication, processing data in software
2. Team Location Synchronization
    - Once the rovers land, we'll need to triangulate the area of research. This will be done by having a set of rovers move to the bounds of the research area and synchronize with the Pluto Positioning System (PPS, instead of GPS). Once we've accomplished this, our rovers will be able to precisely locate some extra samples that we've identified via telescope.
        - Easy: Two rover alignment
        - Hard: Three rover alignment
            - *One three rover alignment will net both the sample and bonus sample
        - Engineering skills: Teamwork & coordination, rover design
3. Climb Mount Fenix
    - We would like to collect samples from a nearby mountain we've nicknamed "Mount Fenix". It will most likely require a specialized vehicle to scale the slope and retrieve these samples.
        - Easy: Lower tier hill
        - Hard: Upper tier hill
        - Engineering Skills: Learning about gear ratios & load balancing your rover

4. Probe Button Jumpstart
   - There is a probe that we have landed in our area of research, but to retrieve the data it collected, it will need a jump start. There is a panel on the front that can be pushed in to give a little juice, and then a panel on the top that will fully enable it.
     - Easy: Repair (push in) front panel
     - Hard: Kick-start probe (push in top panel)
     - Engineering Skills: Lever design, attaching & programming a servo motor

5. Descent Cave Spectral Analysis
   - We would also like to perform a spectrograph of a nearby cave. We'll need to produce specific wavelengths of light to identify what minerals are present.
     - Easy: One wavelength
     - Hard: Multi-wavelength
     - Engineering Skills: Choosing & soldering resistors to light up LEDs, rover design
6. Sample Collection
   - Once samples have been identified, we would like to bring them to our science pod for further analysis. Any sample that can be brought to the science pod will provide us with even more information. **To maximize data capture, we recommend having a G-Rover whose primary task will be bringing samples back to the science pod.**
     - All Samples Retrieved: 10 Points
     - 1-2 Samples Left on Field: 8 Points
     - 3+ Samples Left on Field: 6 Points
     - No Samples Retrieved: 0 Points
     - Engineering Skills: Sample scoop design, remote control handling practice & design

Scoring:

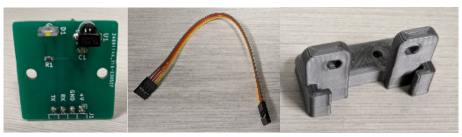
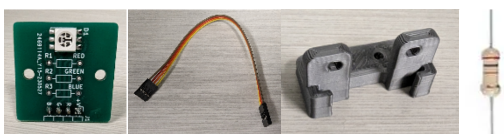
Our performance will be rated by NASA based on how much data we get back to Earth before our equipment freezes over. The following points system will be used:

- Each sample identification (easy level challenge completed) earns **5** points.
- Each bonus sample identification (hard level challenge completed) earns **3** additional points.
- Sample retrieval bonus is scored based on how many samples are left uncollected.
  - All identified samples collected: **10** points
  - 1-2 identified samples left uncollected: **8** points
  - 3+ identified samples left uncollected: **6** points
  - No samples collected: **0** points

The more points we earn, the more money we earn from NASA for the data collected! So, make the most of our time and collect as much data as we can!

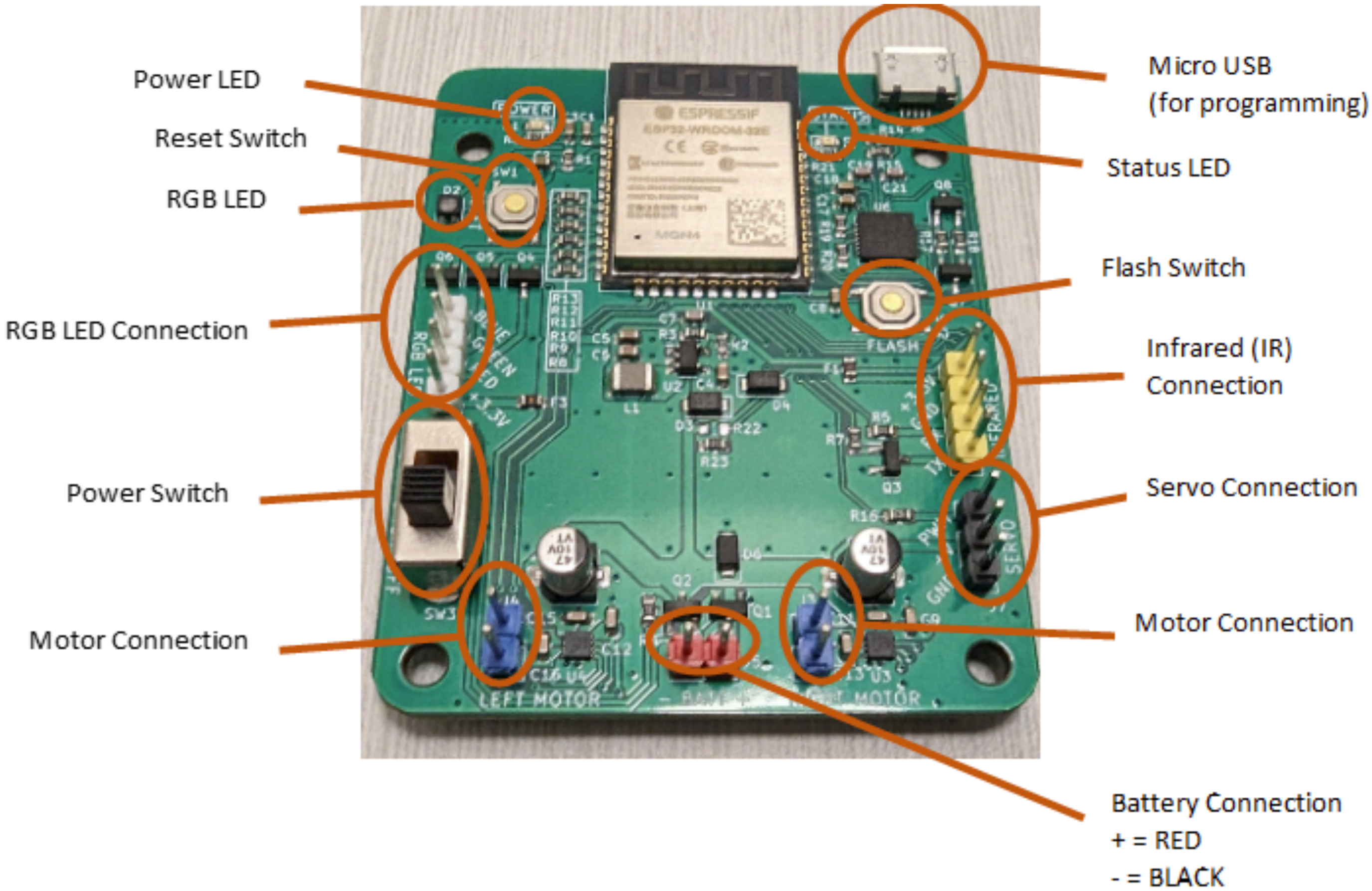
The Kit

Your team received a kit of materials used to make 4 G-Rovers. The kit includes the following materials:

Quantity	Item	
4		Rover Control Boards
6		1:48 Gear Ratio Motors
2		1:120 Gear Ratio Motors
1		Servo
1		IR Communication Kit: IR Communication Board Mounting Bracket Cable
1		LED Kit: RGB LED Board Various Resistors Mounting Bracket Cable
4		Battery Holder
4		Remote Control
32		AA Batteries
8		AAA Batteries
--		Assorted Hardware
4		Flash Drive (Includes software to program rovers)
4		Micro USB cable (for rover programming)

During the week, you will design rover platforms for your parts to mount to. This is not included in this kit but will be delivered later.

Control Board



Remote Control Operation

The remote control you will use has 6 buttons to program and 1 joystick.

For the remote to properly pair with the rover, perform the following steps when powering on the remote:

1) Move the switch on the side of the remote to “GAME”
2) Hold down the “Y” button and keep it held
3) Hold down the power button until the LED turns on.
4) Release both buttons.

Rover Platform Design

Chassis Design Overview:

- The chassis will be the base for the car to attach everything needed to drive and complete the challenges.
- It will be designed in OnShape and then cut out by our shop with a laser cutter.
- It will need to include the hole patterns for mounting your motors and front skid.
- Also, you will want to consider adding holes for attaching sensors/attachments needed for the challenges you have chosen.

Material: Clear Acrylic – 0.118in thick

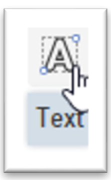
Max dimensions: 6 in x 8 in

Creating the Chassis tips:

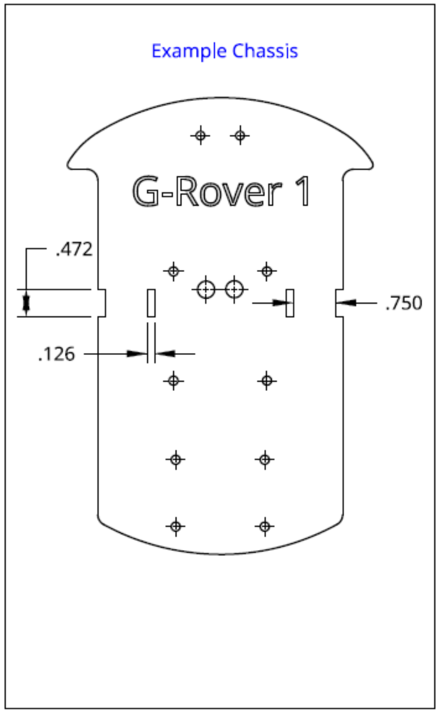
- Create a sketch: <u>SELECT TOP PLANE</u> sketch plane
- Extrude thickness 0.118in
- Remember to cut holes in the chassis for wires to run through for batteries and motors.

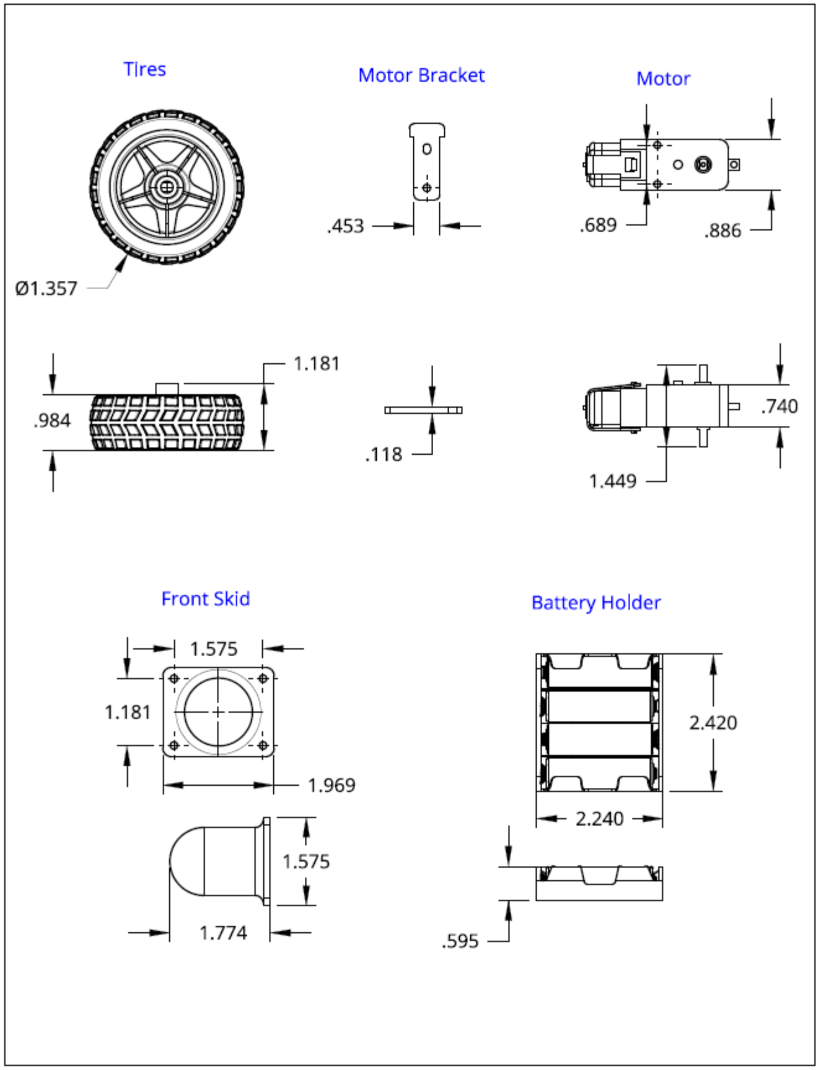
Car Name/Number

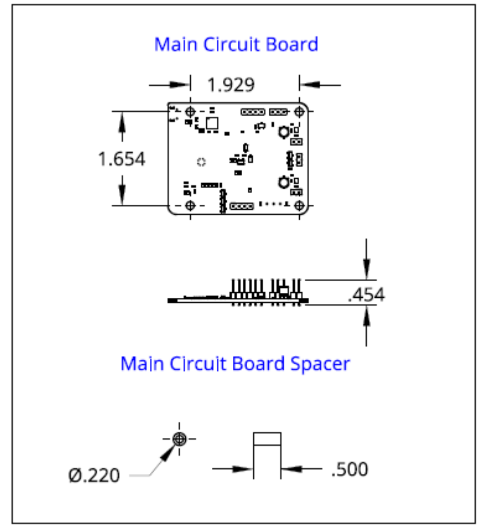
You will want to mark your chassis to be able to identify it after being cut out. The Text feature can be used in OnShape to put a name or number on the car. The laser will only score the lines, so a cut feature can be used, but only cut 0.05 inches. Please only use one-to-two-word names to minimize the cut time.

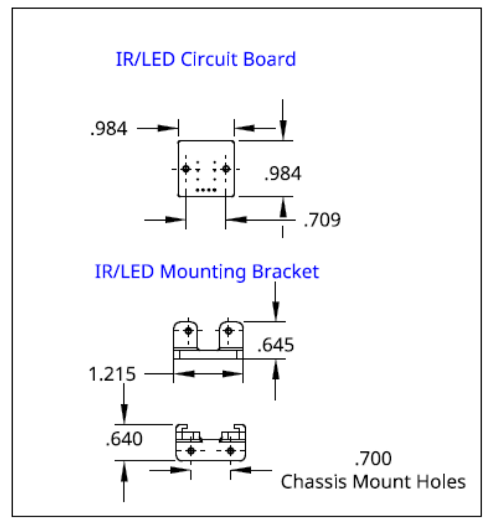


Measurement Cheat Sheet:



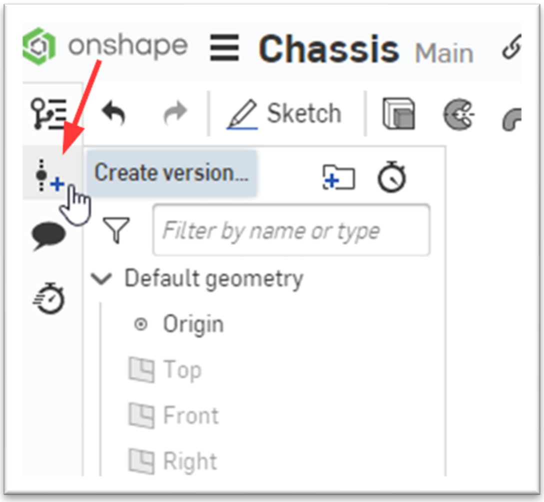




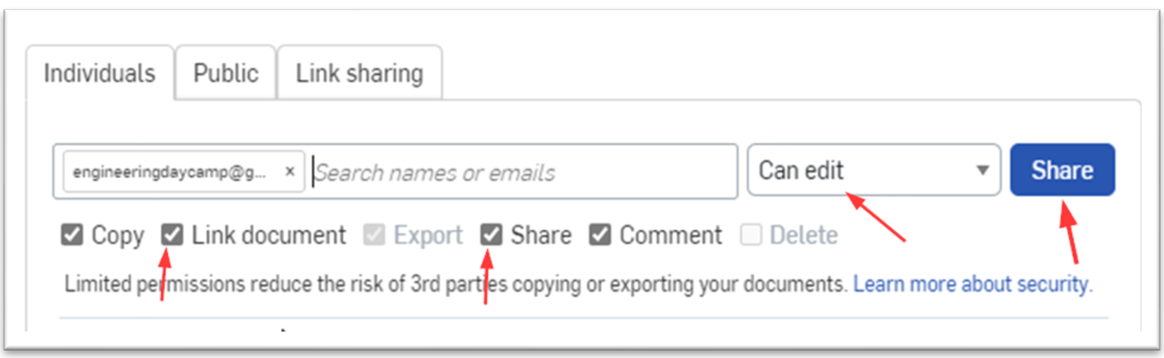


Ready to Cut Out the Chassis?

- Create Version before sharing.



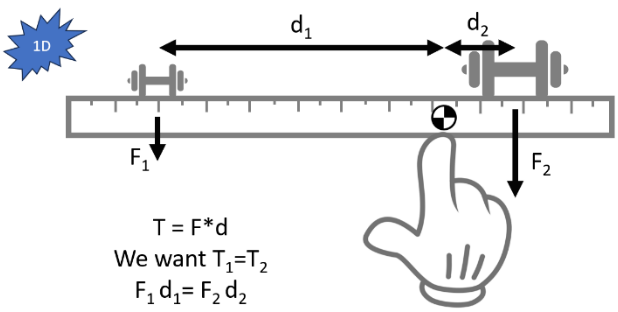
- Share to engineeringdaycamp@gmail.com
- Click on “Link” and “share” check boxes and allow “can edit”.

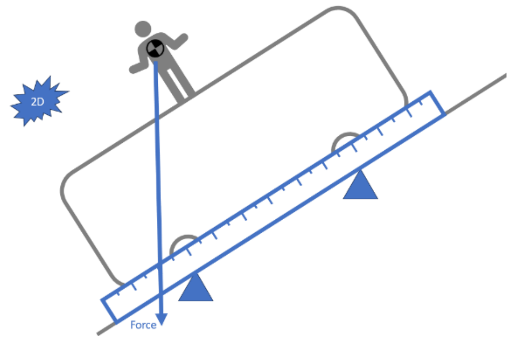


Challenge Specific Material

Climb Mount Fenix: Mechanical Design Considerations

Center of Gravity





Gear Ratios and Torque/Speed

$$\frac{n_1}{n_2} = \frac{\omega_2}{\omega_1} = \frac{T_1}{T_2}$$

- Note how the 1 and 2 flip in each fraction.
- The higher n2 is, the more torque you can produce, but you have less speed
- 1:48 = wheel:motor, for every 1 turn of wheel, the motor turns 48 times

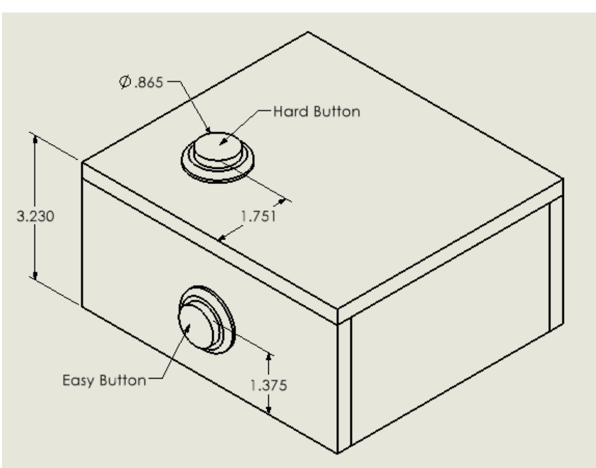
Traction

Friction =

Probe Button Jumpstart: Mechanical Design Considerations

The center of the easy button location is 1.375in from the field surface and the hard button location is 3.58in from the field surface.

Each button is 0.865in in diameter.



You must first press the easy button and then the hard button to unlock both samples.

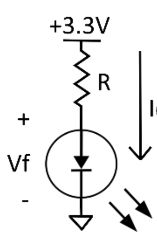
Descent Cave Spectral Analysis: Electrical Design Considerations

Our kit includes a common-anode RGB (red-green-blue) LED (light emitting diode). This LED has three LEDs built into one part, each one making red, green, and blue light. This type of LED is commonly used as it can be used to generate many different colors by dimming each LED (red, green, and blue).

The table below shows examples of colors that can be created using a RGB LED:

Color	Name	Red	Green	Blue
	Red	100%	0%	0%
	Green	0%	100%	0%
	Blue	0%	0%	100%
	Yellow	100%	100%	0%
	Pink	100%	0%	100%
	Cyan	0%	100%	100%
	White	100%	100%	100%

LEDs, like all diodes, will self-destruct if the current through them exceeds limits. For LEDs, the manufacturer recommends a fixed current value. We will use a resistor in series with the LED to limit the amount of current through the LED. We can use ohm's law with the power supply voltage (+3.3V), recommended current through the LED (Id from the table below), and the LED forward voltage (Vd from the table below) to pick the correct resistor value for each color.

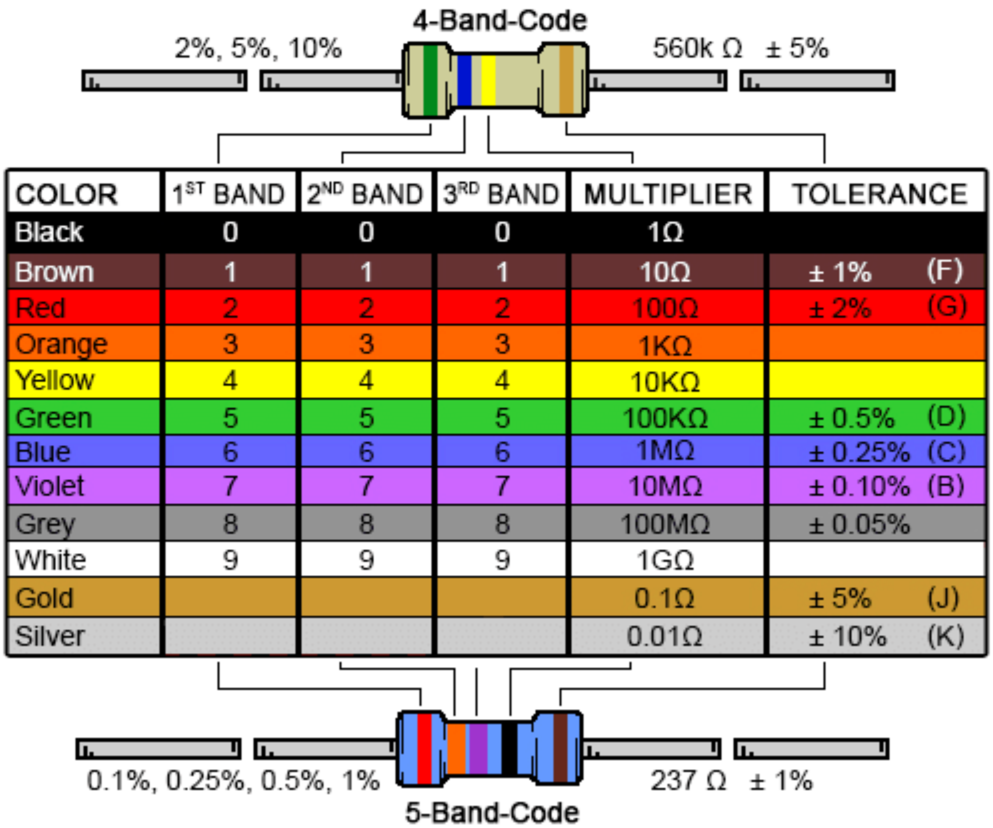


$$R = \frac{3.3V - Vf}{Id}$$

Color	Id	Vf
Red	0.004 A	2.1 V
Green	0.001 A	3.1 V
Blue	0.001 A	3.1 V

Your kit includes the following resistor values to choose from:

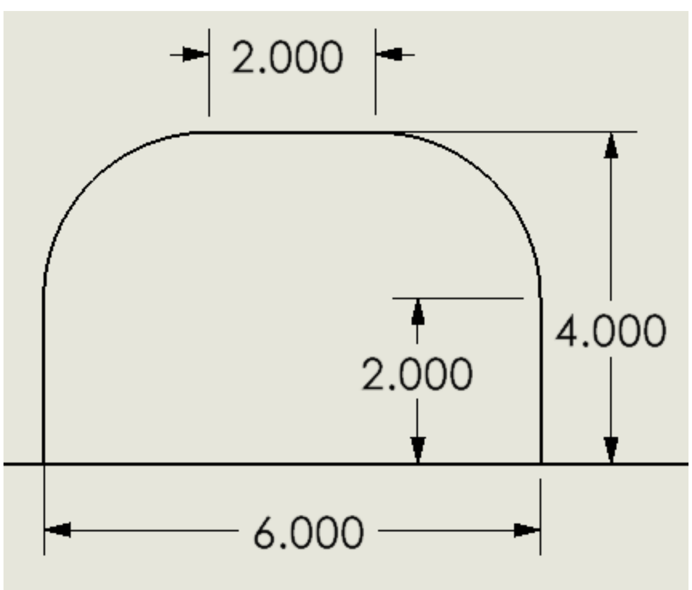
- 200 ohm
- 300 ohm
- 2000 ohm
- 3000 ohm



COLOR	1ST BAND	2ND BAND	3RD BAND	MULTIPLIER	TOLERANCE
Black	0	0	0	1Ω	
Brown	1	1	1	10Ω	± 1% (F)
Red	2	2	2	100Ω	± 2% (G)
Orange	3	3	3	1KΩ	
Yellow	4	4	4	10KΩ	
Green	5	5	5	100KΩ	± 0.5% (D)
Blue	6	6	6	1MΩ	± 0.25% (C)
Violet	7	7	7	10MΩ	± 0.10% (B)
Grey	8	8	8	100MΩ	± 0.05%
White	9	9	9	1GΩ	
Gold				0.1Ω	± 5% (J)
Silver				0.01Ω	± 10% (K)

[Resistor Color Chart Courtesy digikey.com]

Cave entrance dimensions, **keep this in mind when designing your car**. Either your entire car or the part of your car with the LEDs will need to fit into this space. The sensor that is detecting your LED's color is on the far wall of the cave.



Uplink With Satellite: Software Design Considerations

This challenge works with IR which is a way for two devices to communicate wirelessly. IR stands for infrared, which is the type of LED that is used to send signals in this kind of communication. This is actually how your TV remote communicates with your TV!

The **IR transceiver** is the component you attach to your board. The **IR receiver** is on the field.

When you send data wirelessly, the data is passed in chunks we call **packets**. The packets in this challenge include 2 numbers, an **address** and a **command. Packet = (address, command).**

Challenge Communication Steps

1. You will send the receiver an address = a number of your choosing. Packet = (your number, 0)

Easy Level Complete

2. The receiver will send you a packet back. Packet = (address, command)
3. You must complete an operation using the data you received. Use the table below to know which operation to perform based on the address you received.

Your Answer = Command [Operation to Perform]

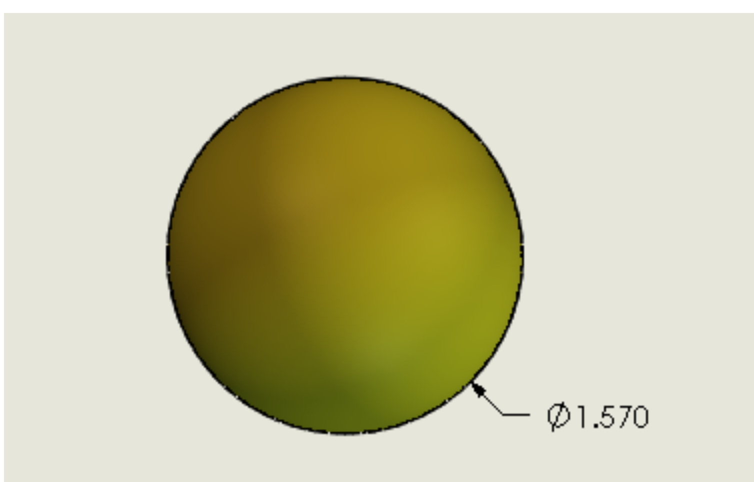
Address Received	Operation To Perform
0	+ 5
1	- 7
2	+ 28
3	- 50
4	+ 64

4. You will send the receiver a response with your answer to the equation as the command.
   Packet = (your number, your answer)

Hard Level Complete

Sample Retrieval: Design Considerations

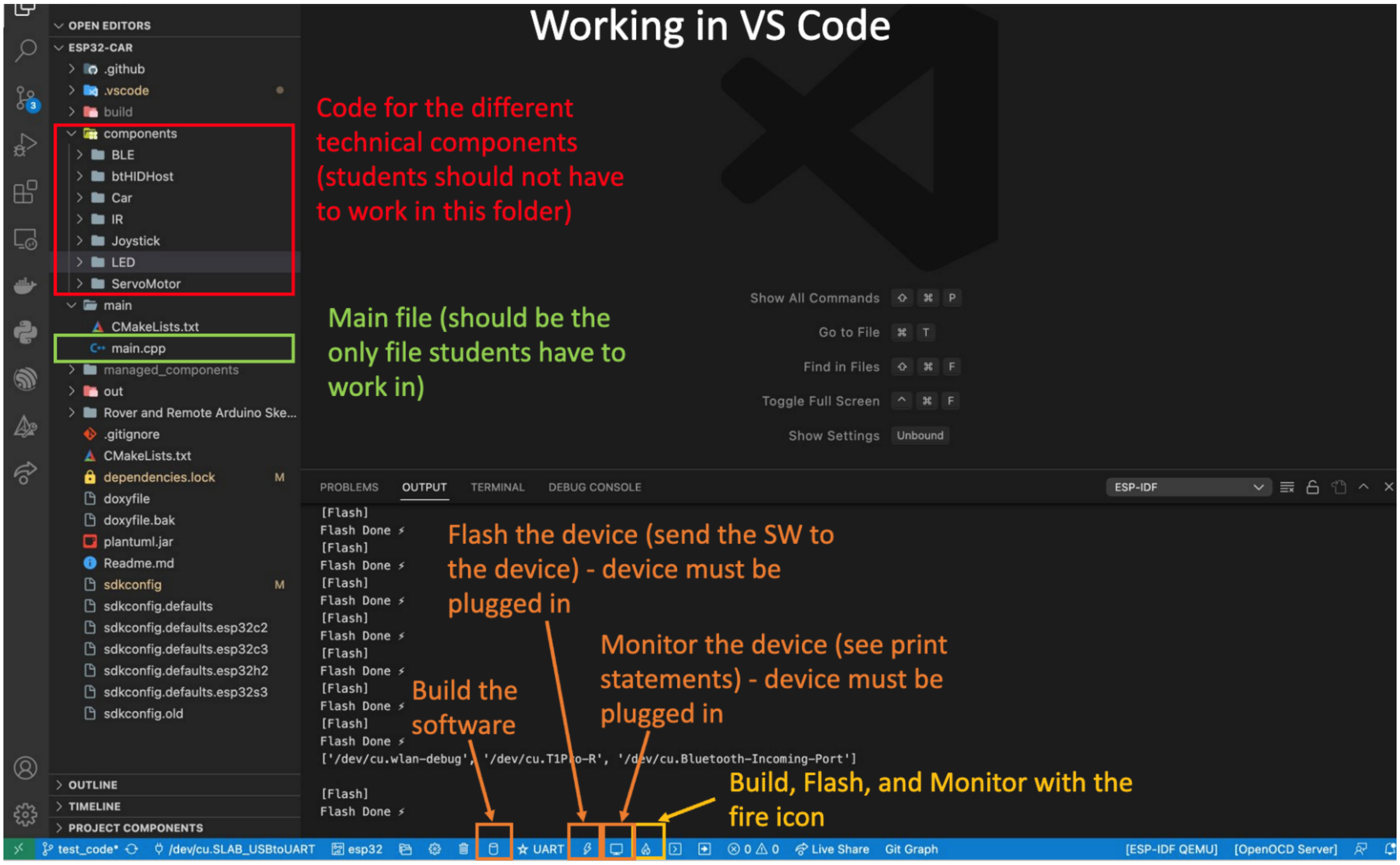
The samples consist of red and yellow ping-pong balls. The ball diameter is 1.57”. The ball weighs 2.7 grams.



Software Reference Material

Working in Visual Studio Code (VSCode)

Running the script (double clicking the file) on your flash drive will open the code folder in VSCode for you. Look at the photo below to know the important



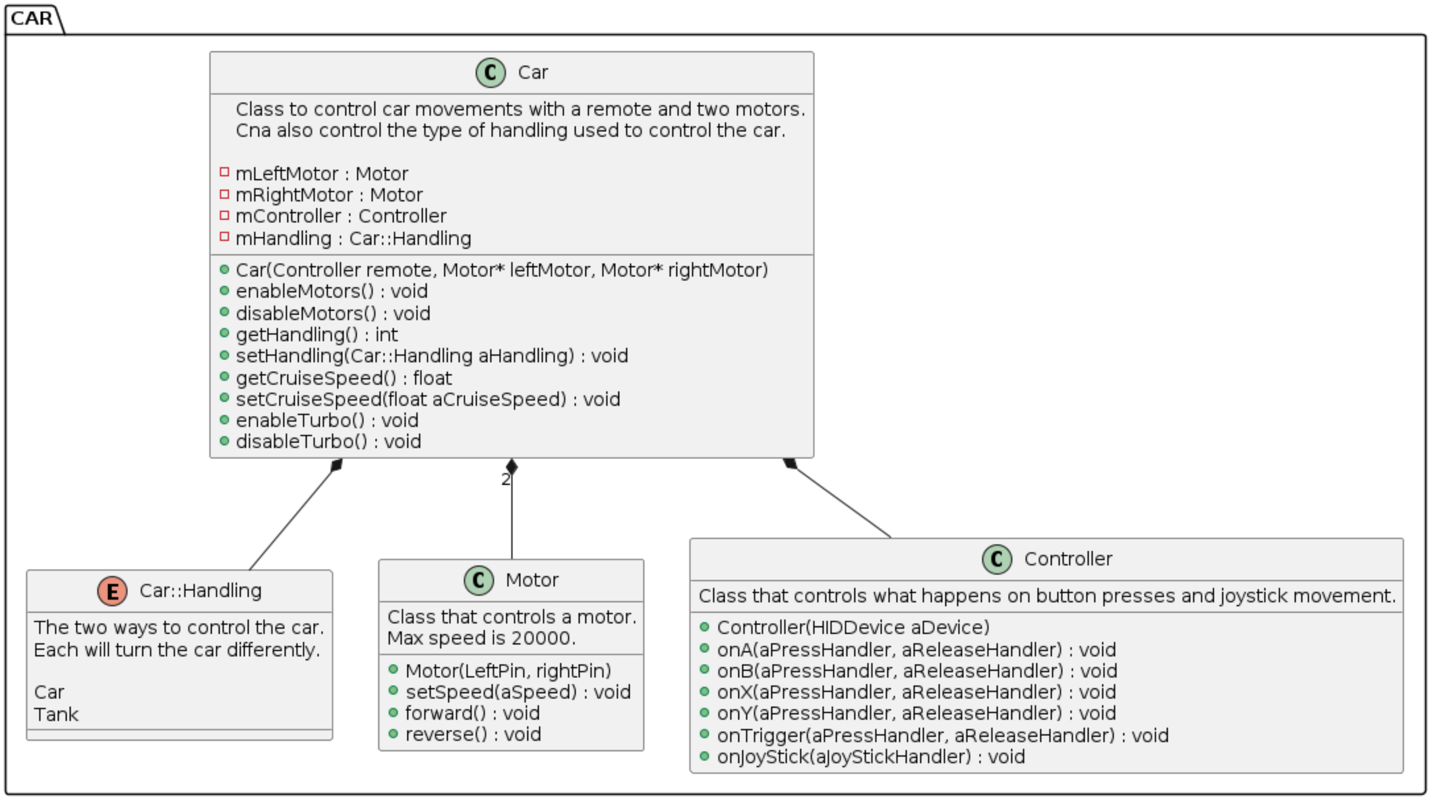
(If you want to download VSCode to use it yourself outside of this project, use [https://code.visualstudio.com/download.)](https://code.visualstudio.com/download)

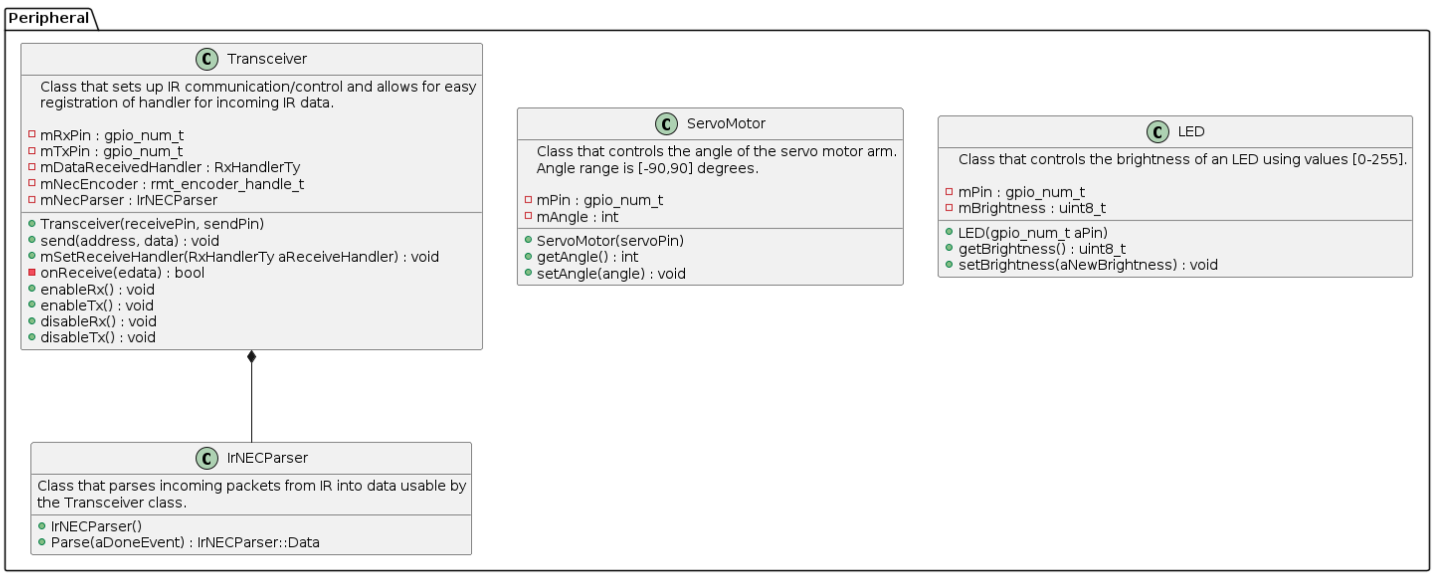
Project Files & GitHub

Project files can be found on the GitHub: [https://github.com/MatthewColvin/ESP32-Car.](https://github.com/MatthewColvin/ESP32-Car)

The project should already be cloned ("cloned" = copied over to your computer) onto your flash drive. Make sure to run "git pull" in the VSCode Terminal before beginning to code.

GitHub is a way to share programming files among developers. We have started you off with some starter code. Most of the code we have written is in the components folder. *You will work in main/main.cpp. You will not have to edit any files in the components folder. * Each folder in the components folder holds classes you will utilize. Those available to you are in diagrams below


CAR
Car
Class to control car movements with a remote and two motors.
Cna also control the type of handling used to control the car.
mLeftMotor : Motor
mRightMotor : Motor
mController : Controller
mHandling : Car::Handling
Car(Controller remote, Motor* leftMotor, Motor* rightMotor)
enableMotors() : void
disableMotors() : void
getHandling() : int
setHandling(Car::Handling aHandling) : void
getCruiseSpeed() : float
setCruiseSpeed(float aCruiseSpeed) : void
enableTurbo() : void
disableTurbo() : void
2
Car::Handling
The two ways to control the car.
Each will turn the car differently.
Car
Tank
Motor
Class that controls a motor.
Max speed is 20000.
Motor(LeftPin, rightPin)
setSpeed(aSpeed) : void
forward() : void
reverse() : void
Controller
Class that controls what happens on button presses and joystick movement.
Controller(HIDDevice aDevice)
onA(aPressHandler, aReleaseHandler) : void
onB(aPressHandler, aReleaseHandler) : void
onX(aPressHandler, aReleaseHandler) : void
onY(aPressHandler, aReleaseHandler) : void
onTrigger(aPressHandler, aReleaseHandler) : void
onJoyStick(aJoyStickHandler) : void


Peripheral
Transceiver
Class that sets up IR communication/control and allows for easy registration of handler for incoming IR data.
mRxPin : gpio_num_t
mTxPin : gpio_num_t
mDataReceivedHandler : RxHandlerTy
mNecEncoder : rmt_encoder_handle_t
mNecParser : IrNECParser
Transceiver(receivePin, sendPin)
send(address, data) : void
mSetReceiveHandler(RxHandlerTy aReceiveHandler) : void
onReceive(edata) : bool
enableRx() : void
enableTx() : void
disableRx() : void
disableTx() : void
ServoMotor
Class that controls the angle of the servo motor arm.
Angle range is [-90,90] degrees.
mPin : gpio_num_t
mAngle : int
ServoMotor(servoPin)
getAngle() : int
setAngle(angle) : void
LED
Class that controls the brightness of an LED using values [0-255].
mPin : gpio_num_t
mBrightness : uint8_t
LED(gpio_num_t aPin)
getBrightness() : uint8_t
setBrightness(aNewBrightness) : void
IrNECParser
Class that parses incoming packets from IR into data usable by the Transceiver class.
IrNECParser()
Parse(aDoneEvent) : IrNECParser::Data

Sketch Paper

Sketch Paper

Sketch Paper

Sketch Paper

Sketch Paper