UAA ROBOTICS CLUB

2023/24 Rookie Challenge Solutions

GENERAL MECHANICAL

CHALLENGE 1 (Difficulty: Easy)

Using the tools and materials in the lab, build a catapult to launch a triball. 10” by 10” max. If you use any of the power tools, make sure you have supervision from experienced club members.

There may be a competition at the end!

CHALLENGE 2 (Difficulty: Medium)

Download Solid Works. Create a license plate with your name or other info.

VEX MECHANICAL

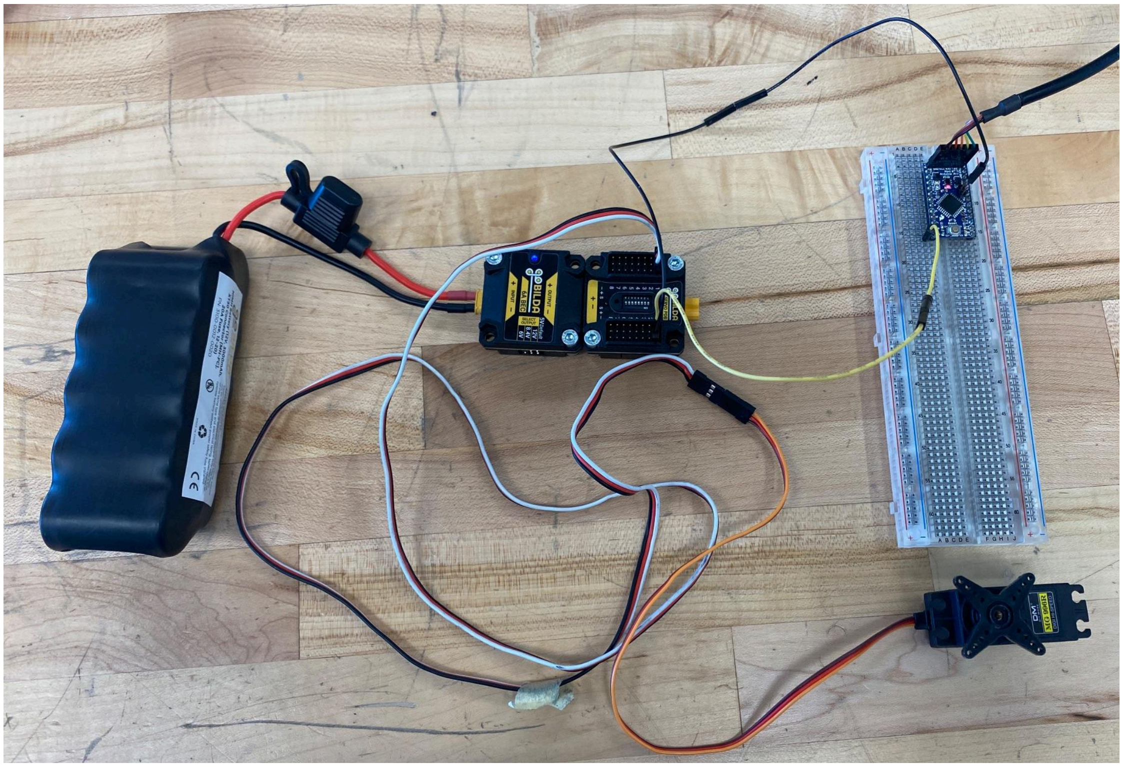
CHALLENGE 1 (Difficulty: Easy)

Drive Granny to score the most points.

GENERAL PROGRAMMING

CHALLENGE 1 (Difficulty: Easy)

1. Connect the servo to the positive side of the PDP Board (to power the servo). The servo black wire should go to the -, the red wire should go to +, and the orange wire should go to the S (signal). On the negative side of the PDP, connect the S to a PWM pin on the arduino (for this example, we're using pin 9). PWM pins have a white circle around them. Connect the - to Ground on the arduino (you do not need to connect + on this side)



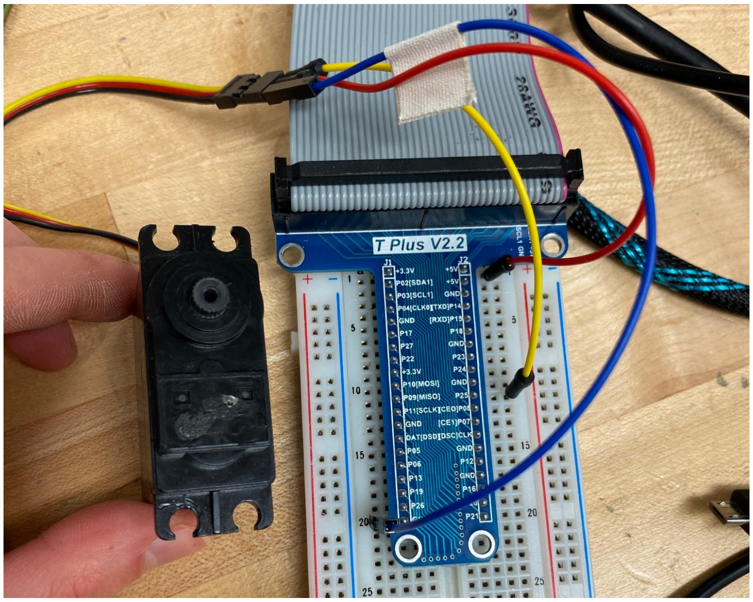
2. Connect the Arduino to your computer via the TTL to USB cable. Make sure to select the correct board (Arduino Pro or Pro Mini) and select Tools>Processor>A328P(3.3V, 8MHz) in the IDE.
3. Look at the Arduino Servo documentation and this example and create your program. Here is one example:

```
#include <Servo.h>

Servo myservo;
```

```
// Tell the Arduino that the servo is on pin 9
void setup() {
   myservo.attach(9);
}

// Move the Servo back and forth between 30 and 170 degrees
void loop() {
    myservo.write(30);
    delay(1000);
    myservo.write(170);
    delay(1000);
}
```

CHALLENGE 2 (Difficulty: Medium)

1. Check out the RPi docs for pin diagrams. On the breadboard, connect the servo red wire to +5 (power), the black wire to GND, and the yellow wire to one of the PWM pins (GPIO pin 2 in this setup.). PWM pins are GPIO12, GPIO13, GPIO18, and GPIO19



2. SSH into the the Pi using the command:

```
ssh roboticspi@roboticspi.local
```

The password is

```
icebergLettuce
```

3. Navigate to NewbieChallenges folder using

```
cd NewbieChallenges
```

4. Create and open a new python file using

```
nano <myProgramName>.py
```

5. Use the gpiozero library to write a program. Here is an example program.

```
from gpiozero import Servo
from time import sleep

servo = Servo(2)

while True:
    servo.min()
    sleep(1)
    servo.max()
    sleep(1)
```

6. Save and exit using control-S and control-X. Then run your program using

```
python <myProgramName>.py
```

VEX-U PROGRAMMING

CHALLENGE 1 (Difficulty: Easy)

1. Connect your computer to the brain via USB. Connect the brain to the battery. Connect the motor to a port on the brain (here we’re using port 10)
2. Install VS code and get the VEX Robotics extension.
3. Open a C++ V5 Empty Template Project.
4. In main.cpp write your program. Look at the Example Projects. Also, Right click on VEX Built-In functions and select “Go to Definition” to see VEX documentation in the code. Make sure you initialize the motor to the correct port. Here is an example:

```
#include "vex.h"

using namespace vex;

vex::brain Brain;
vex::motor Motor = vex::motor(PORT10);

int main()
{
    Motor.spin(forward, 100, rpm);
    while (1)
    {
        // Allow other tasks to run
        this_thread::sleep_for(10);
    }
}
```

5. Compile and run your program by pressing the arrow at the bottom of VS Code,.

CHALLENGE 2 (Difficulty: Easy)

The drivetrain for the new VEX-U Over Under challenge is already built. Using the VEX library's functions, create a project to move the drivetrain forward a set distance (like a foot) purely using the robot brain, don't worry about the controller for this one.

Hint: "using the robot brain" means that just uploading your project and hitting run in VS Code should move the drivetrain.

CHALLENGE 3 (Difficulty: Medium)

Write a program that allows you to control the movement of the drivetrain with the controller. You should be able to move forward/backward, stop, turn, and drive in arcs with the joysticks.

Hint: Account for how many wheels are being used in the drivetrain.

CHALLENGE 4 (Difficulty: Hard)

Granny, which is one of the robots from last year's VEX-U Spin Up challenge, still works and can be used to test code. For last year's challenge, we used an intake mechanism on the bottom of the robot to collect disks and a flywheel mechanism to launch the disks. We will supply you with the complete body of code for Granny, but take out the code for intake and flywheel, so that you can write it yourself!

The intake should start and stop rolling at command from the controller. Make sure to test it with actual disks! The flywheel should be able to launch 1, 2, or 3 disks one after the other. It does that by spinning the flywheel up to full speed, then flicking the disks one by one so they get launched.

CHALLENGE 5 (Difficulty: Hard)

Assuming the drivetrain starts on a virtual coordinate plane, with a position of {0,0} and a heading of 0 degrees, write a program to autonomously move the drivetrain to these points:

{0ft, 2ft}

{2ft, 2ft}

{-2ft, 0ft}

{-2ft, 2ft}

Run your program on the field on the 4th floor. Every tile is about a foot.

Hint: This will involve trigonometry and other math. You will need to keep track of your position every time you reach a point.

Bonus: Figure out how to move in an arc (as opposed to stopping, turning to a certain heading, then moving again) autonomously. This is very hard to figure out. In fact, we haven't even figured it out yet!