

# California State University, Northridge

College of Engineering and Computer Science



## Motor control using controller in VEX Code PRO

CSUN Vex Robotics Club

Written By: Cristian Robles

Fall 2022

## 1. Introduction:

This is meant to give everyone the ability to program a robot in driver control in order to test when a new mechanism is developed and needs code to be run. We will be using VEX Code PRO which can be downloaded at <https://www.vexrobotics.com/vexcode/install/v5>  
Later coding for competition will be done in PROS <https://pros.cs.purdue.edu/>

## 2. Materials:

- Robot with brain, motors, and radio mounted
- Laptop with VEX Code PRO
- Micro USB cable
- Vex Controller

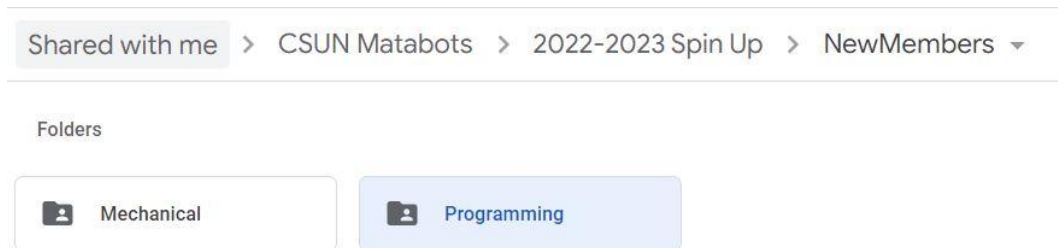
## 3. Procedure:

### - 3.1 Downloading and installing VEX Code PRO

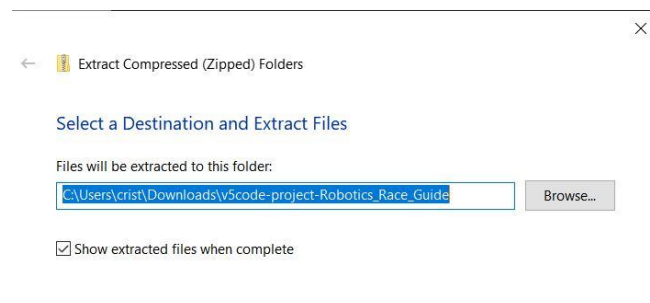


Go to the link <https://www.vexrobotics.com/vexcode/install/v5> and download VEX Code PRO. DO NOT download the blocks and text one

### - 3.2 Download Sample Program

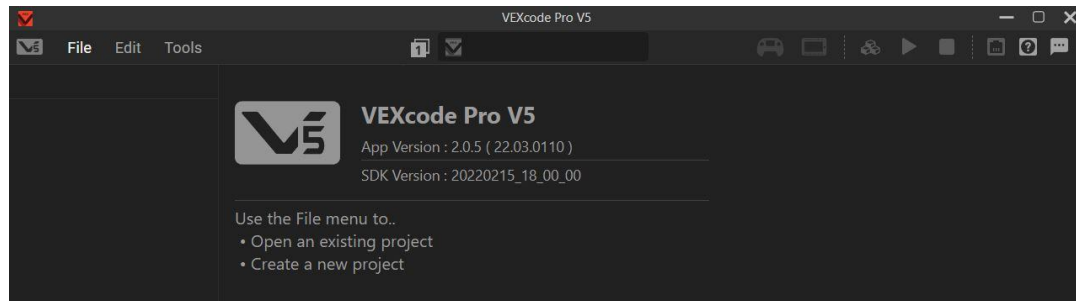


Go to the clubs google drive and download the zip file called RoboticsRaceGuide in CSUN Matabots > 2022-2023 Spin Up > NewMembers > Programming > Code

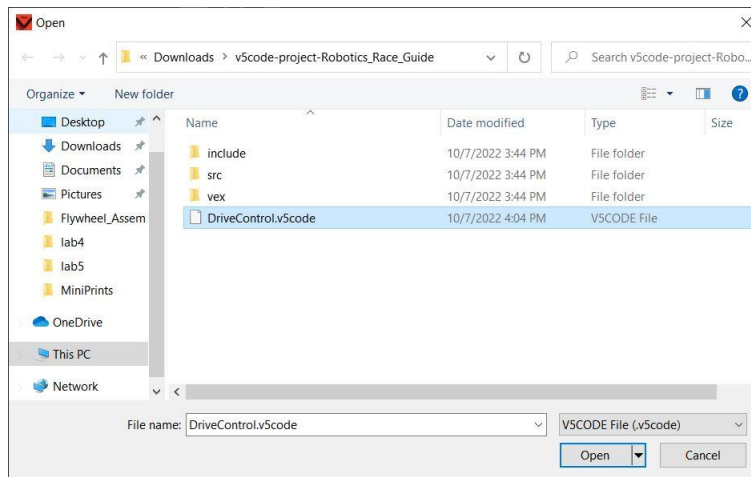


Unzip file and save it to whatever location you want inside your computer

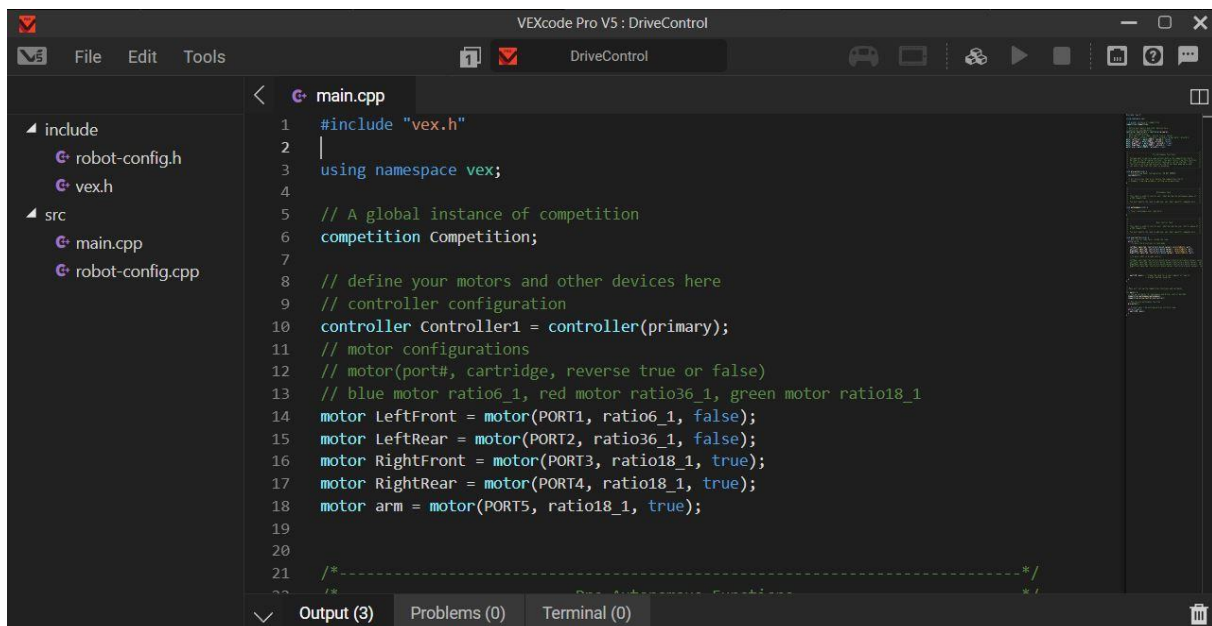
### - 3.3 Open VEX Code PRO



### - 3.4 Open the .v5code file to open your project

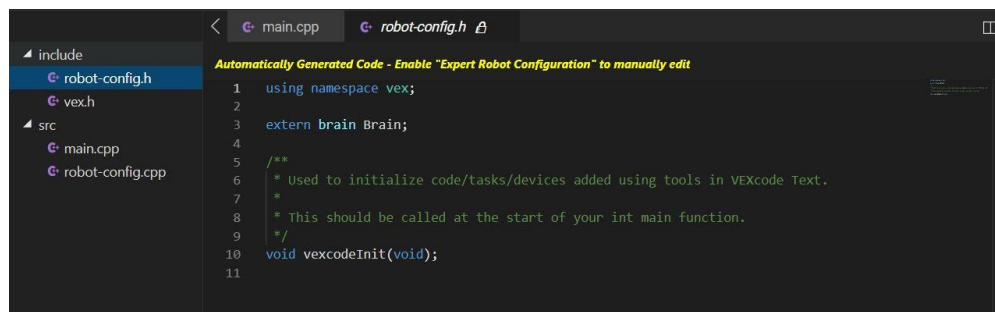


Click File > Open in VEX Code and go to the place where you saved the program and open the .v5code file



This is what you should see

### - 3.5 Explanation of code (robot-config .h and .cpp files)

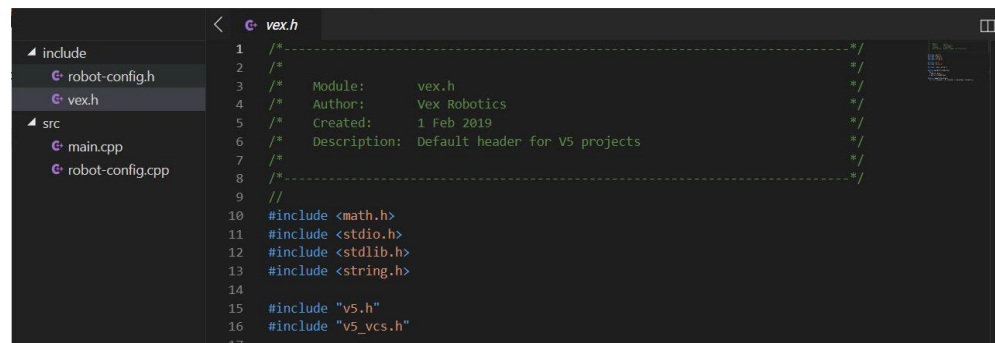


The screenshot shows the VEXcode IDE with the 'robot-config.h' file open. The file is part of a project named 'main.cpp'. The code in the file is as follows:

```
1 using namespace vex;
2
3 extern brain Brain;
4
5 /**
6  * Used to initialize code/tasks/devices added using tools in VEXcode Text.
7  *
8  * This should be called at the start of your int main function.
9  */
10 void vexcodeInit(void);
11
```

In more advanced code you will have to change the robot-config .h and .cpp files but since this is meant to just teach moving motors with joysticks we will not be configuring those

### - 3.6 Explanation of code (vex.h file)

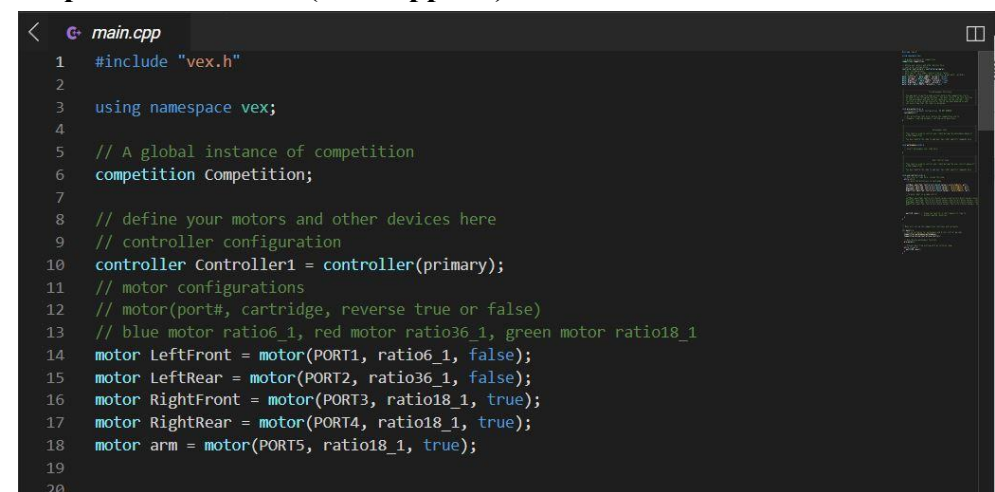


The screenshot shows the VEXcode IDE with the 'vex.h' file open. The file is part of a project named 'main.cpp'. The code in the file is as follows:

```
1 /*-----*/
2 /*
3  * Module: vex.h
4  * Author: Vex Robotics
5  * Created: 1 Feb 2019
6  * Description: Default header for V5 projects
7  */
8 /*-----*/
9 //
10 #include <math.h>
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <string.h>
14
15 #include "v5.h"
16 #include "v5_vcs.h"
17
```

The vex.h file will not change either. It is there to provide the library of functions we use to program stuff with

### - 3.7 Explanation of code (main.cpp file)



The screenshot shows the VEXcode IDE with the 'main.cpp' file open. The code in the file is as follows:

```
1 #include "vex.h"
2
3 using namespace vex;
4
5 // A global instance of competition
6 competition Competition;
7
8 // define your motors and other devices here
9 // controller configuration
10 controller Controller1 = controller(primary);
11 // motor configurations
12 // motor(port#, cartridge, reverse true or false)
13 // blue motor ratio6_1, red motor ratio36_1, green motor ratio18_1
14 motor LeftFront = motor(PORT1, ratio6_1, false);
15 motor LeftRear = motor(PORT2, ratio36_1, false);
16 motor RightFront = motor(PORT3, ratio18_1, true);
17 motor RightRear = motor(PORT4, ratio18_1, true);
18 motor arm = motor(PORT5, ratio18_1, true);
19
20
```

In this part of the main.cpp file the only thing you have to worry about is the configuration of the controller and motors which is from line 10-18

```

66 void usercontrol(void) {
67     // User control code here, inside the loop
68     while (1) {
69         //To move the drivetrain in tank mode
70
71         LeftRear.spin(fwd, Controller1.Axis3.value(), velocityUnits::pct);
72         LeftFront.spin(fwd, Controller1.Axis3.value(), velocityUnits::pct);
73         RightRear.spin(fwd, Controller1.Axis2.value(), velocityUnits::pct);
74         RightFront.spin(fwd, Controller1.Axis2.value(), velocityUnits::pct);
75
76         //To move robot in arcade control
77         /*
78         LeftRear.spin(fwd, Controller1.Axis3.value()+Controller1.Axis1.value(), velocityUnits::pct);
79         LeftFront.spin(fwd, Controller1.Axis3.value()+Controller1.Axis1.value(), velocityUnits::pct);
80         RightRear.spin(fwd, Controller1.Axis3.value()-Controller1.Axis1.value(), velocityUnits::pct);
81         RightFront.spin(fwd, Controller1.Axis3.value()-Controller1.Axis1.value(), velocityUnits::pct);
82         */
83     }
84 }

```

After the motor configuration you really don't have to worry about any of the code until about line 66 where the user control function starts. In here is where you will program the motors and include other code that will run in the driver control portion of the program which is the only one that will be active for this guide

### - 3.8 How to configure motors (PORT)

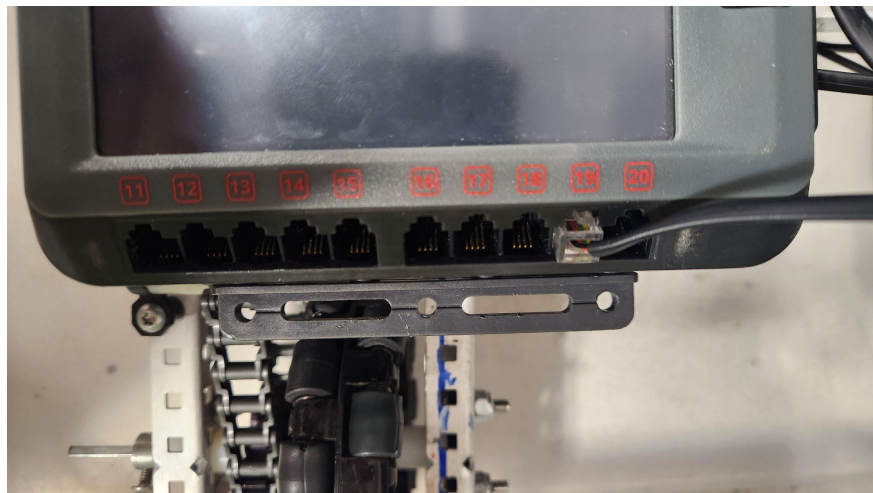
```

motor LeftFront = motor(PORT1, ratio6_1, false);
motor LeftRear = motor(PORT2, ratio36_1, false);
motor RightFront = motor(PORT3, ratio18_1, true);
motor RightRear = motor(PORT4, ratio18_1, true);
motor arm = motor(PORT5, ratio18_1, true);

```

As you can see the motor() takes in 3 arguments

The first is the number port your motor is connected to on the brain



Here the wire is connected to port 19. Assuming it was connected to the Left front motor the code will be

```
motor LeftFront = motor(PORT19, ..., ...);
```

### - 3.9 How to configure motors (ratio)

The vex motors have internal cartridges in them and in order to properly code you have to specify which one you are using in your code. The options are 200RPM (GREEN), 100 RPM (RED), and 600 RPM (BLUE)



Here we can see the motor is red internally meaning it has a 100 RPM cartridge, the motor the code will be

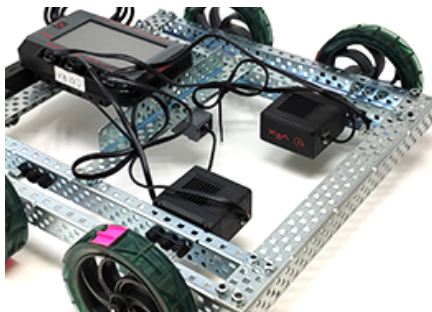
```
motor LeftFront = motor(PORT19, ratio36_1, ...);
```

ratio36\_1 corresponds to 100 rpm red cartridges

ratio18\_1 corresponds to 200 rpm green cartridges

ratio6\_1 corresponds to 600 rpm blue cartridges

### - 3.10 How to configure motors (reverse boolean; true or false)



On a robot the motors on one side of the chassis are mirrored when compared to the motors on the other side so we have to take that into account as they will not be spinning the same way when the same control is applied to them. One side must be reversed so set all your left side motors to forward by passing in false and all your right side ones to reverse by passing in true

The code will be

```
motor LeftFront = motor(PORT18, ratio36_1, false);
```

```
motor RightFront = motor(PORT19, ratio36_1, true);
```



### - 3.11 Using joystick to pass in values to power motors

```
70  
71 LeftRear.spin(fwd, Controller1.Axis3.value(), velocityUnits::pct);  
72 LeftFront.spin(fwd, Controller1.Axis3.value(), velocityUnits::pct);  
73 RightRear.spin(fwd, Controller1.Axis2.value(), velocityUnits::pct);  
74 RightFront.spin(fwd, Controller1.Axis2.value(), velocityUnits::pct);  
75
```

This is the code for running a motor using a controller joystick  
Here we use the motor.spin() function which takes in 3 parameters. The first is the direction of the signal and since we already reversed the motors in the declaration we always set this to fwd which means forward. The second value passed a number. You can put LeftRear.spin(fwd, 100, velocityunits::percent) and this will run the motor at 100 percent but since we want to use the controller we use controller.Axis.value() which returns the current joystick value on the controller which ranges from -100 to 100



### - 3.12 Connecting controller to robot



Make sure you have a radio connected to the brain on your robot

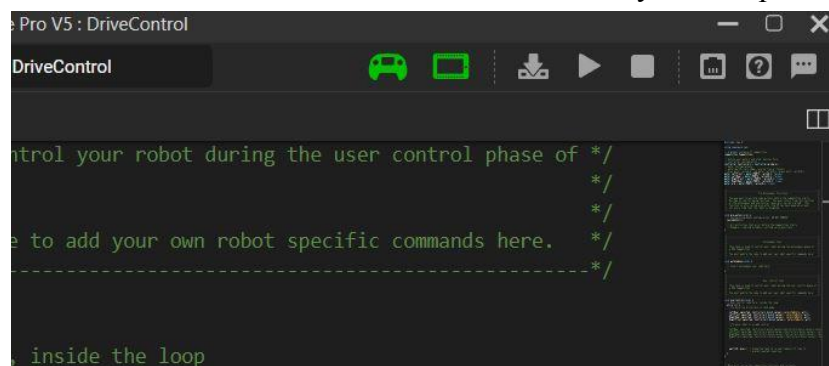


Plug in the controller to any working ports on the brain by using the ports on the top of the controller and a v5 motor cable. Once your robot is connected to the brain you should see the screen above

### - 3.13 Connecting to your computer

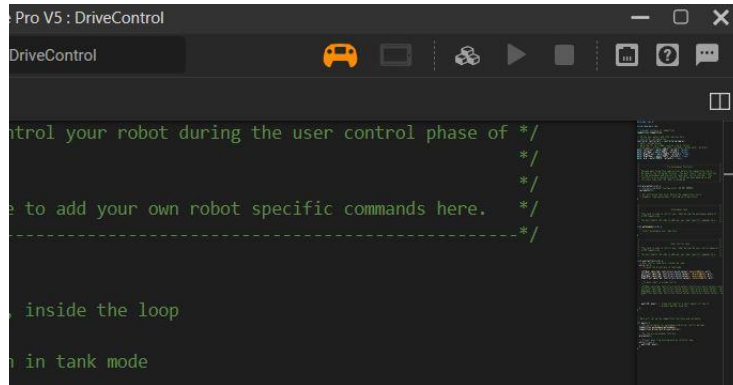


Use a micro usb cable to connect the controller to your computer



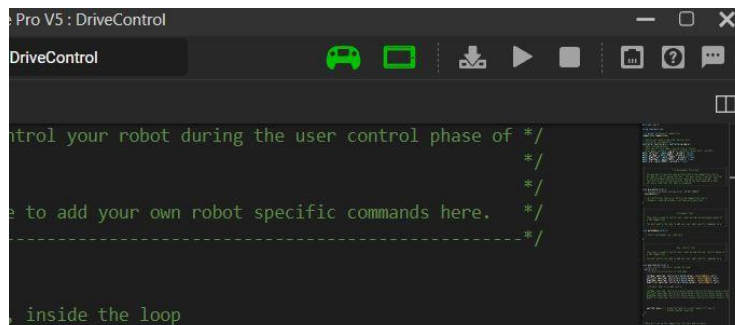
If you get a green brain and controller logo inside of VEX Code Pro you have successfully connected and are ready to upload your code onto the robot





If you get a yellow controller that means the brain is not connected to the controller and you won't be able to upload your program. Reasons for this could be that the robot is off, the cable is bad, or your controller is not connected to the brain (repeat step 3.12)

### - 3.14 Uploading



Hit the button in between the green brain button and the play button and your code should be uploaded to the bot. Sometimes it fails randomly, if it fails and gives you a code error go and fix it. If it fails but no code error was given, try again until it works it may take a couple of tries sometimes.

### - 3.15 Running your code



On the controller go to the middle option which is named programs that is in between the controller and wrench icon



Here look for your program in one of the 8 slots



After selecting the program you uploaded hit run and it should run your code

### - **3.16 Debugging your program**

You will likely notice that your robot is not running the right way, or that you didn't program a motor accidentally, or that you don't have a motor in the correct port. Whatever the case may be, code usually never works the first time. Since I cannot see the future and see your error just figure it out and please ask questions if you need help. If you find any problems with this guide let me know as well so I can update it for future members to use