Microbit Robot With The L9110S Motor Driver Board

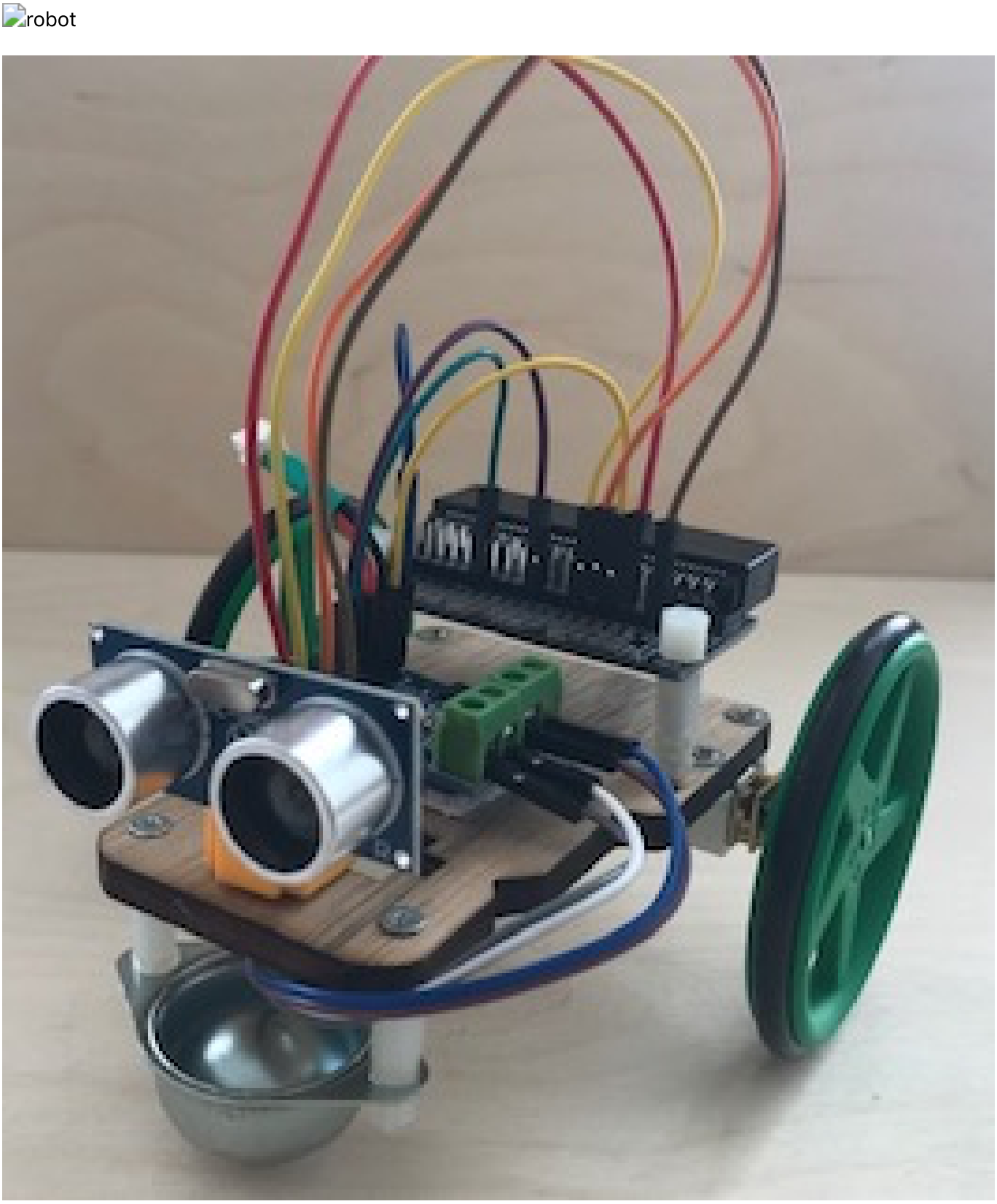

Table of Contents

Description

Build a robot powered by a Microbit that will move and avoid obstacles.

Assembling the Robot

cf pdf file in Assembling Instruction folder

Back To The Top

Understanding How Motors Work

To control the motors, we need to access the pins of the microbit . We do this through the Edge connector and the motor driver L9110s.

Each motor has two pins connected to it. Right motor: Pin 8, Pin 12 Left motor: Pin 0, Pin 16

```
L9110s & Edge connector:
Right Motor: A-1A ---> pin8 of the edge connector
Right Motor: A-1B ---> pin12 of the edge connector

Left Motor: B-1A ---> pin0 of the edge connector
Left Motor: B-1B ---> pin16 of the edge connector
```

The simplest way to make the motors move is to set one pin to HIGH (1)and the the other pin to LOW(0) (to move full speed forwards). We do this by sending power to the respective GPIO pins using write_digital.

To make the make the left motor forwards,we will write in Python:

```
pin0.write_digital(1); pin16.write_digital(0)
```

To move the motor at full speed in reverse, we change which pin is 0 (Low) and 1 (High). Move left motor Reverse:

```
pin0.write_digital(0); pin16.write_digital(1)
```

Back To The Top

How To Make The Robot Turn Left and Right

To turn the robot left we need to tell one motor to go forward, and the other to go backwards.

Motor A is the right motor and it's connected to pins 8 and 12. Motor B is the left motor and is connected to pins 0 and 16.

So to turn the robot left, motor A needs to go forwards and motor B backwards.

```
pin8.write_digital(1); pin12.write_digital(0)
```

```
pin0.write_digital(0); pin16.write_digital(1)
```

The code to turn the robot right is a reverse of what we set for turning left.

```
pin8.write_digital(0); pin12.write_digital(1)
```

```
pin0.write_digital(1); pin16.write_digital(0)
```

How To Make The Robot Move Forward and Backward

To move robot forward, both motors need to go forward:

```
pin8.write_digital(1); pin12.write_digital(0)
```

```
pin0.write_digital(1); pin16.write_digital(0)
```

To move robot backward, both motors need to go backward:

```
pin8.write_digital(0); pin12.write_digital(1)
```

```
pin0.write_digital(0); pin16.write_digital(1)
```

How To Stop The Motor

We will need to set all of the GPIO pins connected to the motors to off:

```
pin8.write_digital(0); pin12.write_digital(0)
```

```
pin0.write_digital(0); pin16.write_digital(0)
```

Back To The Top

Changing The Speed of The Motor

If we want to change the speed of a motor, so that it is not going at full speed all the time, we need to use PWM (Pulse Width Modulation). This is a means of changing the amount of power given to the motor by switching it on and off very fast.

To change the PWM value of a pin, we must use the analog_write commands. These can be set to a value between 0 (always off) to 1023 (always on), so 50% would be 511.(which half of 1023) Here are the

commands to change the speed of the Right motor to approx 50% (value is 511) Move right motor forwards at 50%

> pin8.write_analog(511); pin12.write_digital(0)

What about moving the right motor Reverse at 50%? Doing this for the motors moving in reverse is a little different. We need to change the second pin to 1 for reverse. We then simply take the number (512) in this case) away from 1023, giving 512:

> pin8.write_analog(512); pin12.write_digital(1)

Back To The Top

Now that you understand how motors work, let's move on to our projects!

Radio Control using Accelerometer

Project 1

We are now going to use a radio link to connect two Microbits which enabled strings of data to be sent between devices. We will also use the accelerometer feature of the Microbit.

For this tutorial, you'll need two micro:bits. Our robot will be controlled by micro:bit 1 that will read data from the built-in accelerometer and communicate the information to micro:bit 2 attached to our robot. We will code all of this project in MicroPython, using the Python editor for microbit, on the Microbit.org website.

How to access Python Editor on Microbit.org website.- Click on the link below to play the video

> https://youtu.be/6CN_C24W_hU

The Microbit can measure acceleration forces, measured in G. We will create a variable called gesture and we will store the following gestures: left, right, up and down. In order to indicate the direction of the gesture we will display arrows on the Microbit LEDs (for example, if gesture is right, we will show a right arrow that will point towards the right). A radio signal (left, right, forward, reverse) will be sent to the 2nd Microbit.

1. Coding the controller - Microbit #1

Go to your Microbit Python editor, write the code for the Controller, save it and download it to the Microbit #1

> The Python code can be found in the Microbit Robot folder

2. Coding the Robot - Microbit #2

The second Microbit will receive a radio signal from the 1st Microbit: left, right, forward, reverse. Depending on the message received, the robot will move left, right, forward or backwards.

- Plug the download cable to your Microbit

- Go to your Microbit Python editor, start writing the code for the robot, save it and download it to the Microbit #2

How to download your “code the controller” code to Microbit & save it - From the Python Editor on Microbit.org website.- Click on the link below to play the video

https://youtu.be/ltqHbUjPnq4

> The Python code can be found in the Microbit Robot folder

The Ultrasonic Distance Sensor

Project 2

In this project, we will use the Ultrasonic sensor to detect the distance to the nearest objects. It will be placed on the robot. Once an object is detected by the Ultrasonic sensor, the robot will avoid it.

The Ultrasonic sensor has 4 pins; ground (GND), Echo Pulse Output (ECHO), Trigger Pulse Input (TRIG), and 3V Supply (Vcc). (There are 5V Ultrasonic but we will use the 3V ones as the Microbit can only support 3V)

The Ultrasonic works with sound waves. It will send out a signal on one pin and receive a signal back on another. It uses the timing between those two to determine how far away an object is.

The complete code below has 2 parts:

- a function called sonar ( ) which returns the distance to the nearest object
- and the code for when the sonar meets an object: If the object is less than 15cm from the sonar, the robot will reverse, then turn right at low speed. Once no object detected, the robot will keep going forward.

First, we need to import the utime library: In MicroPython we can use the utime module to measure time at microsecond level

The utime module provides functions for getting the current time and date, measuring time intervals, and for delays.

utime.ticks_us() Returns the number micro seconds since the power was applied.

utime.sleep_us(us) Sleep for the given number of micro seconds.

utime.ticks_diff(ticks1, ticks2) Measure ticks difference between values returned from ticks_us()

> The Python code can be found in the Microbit Robot folder

Back To The Top

Components

What you will need

- 2 x Microbit
- 1 x Microbit Edge Connector
- 1 x Motor Driver L1190S
- 1 x Ultrasonic Sensor
- 1 x Wooden Robot Chassis
- 2 x Gear Motors
- 2 x Wheels
- 1 x Caster wheel
- Jumper wires
- 1 x Mini Breadboard
- 2 x Battery pack
- Screws

Back To The Top

Author Info

- Twitter - @girlsintocoding
- Website - Girls Into Coding

Back To The Top