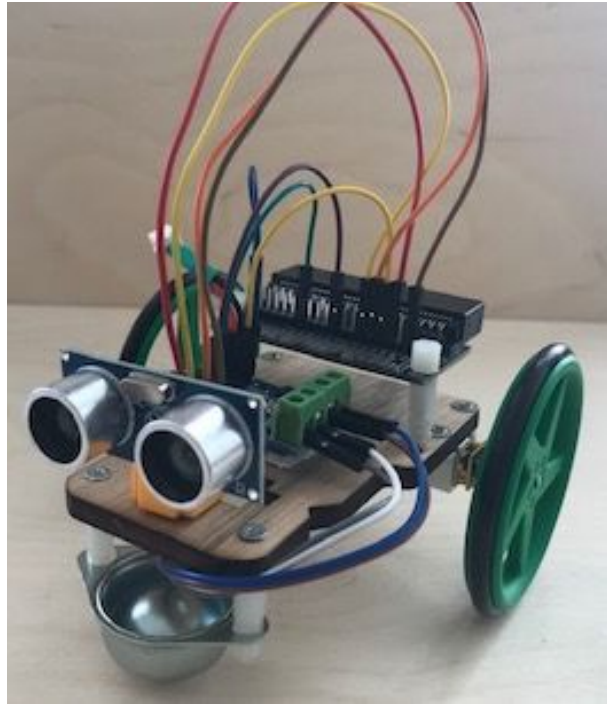




CODING YOUR MICRO:BIT ROBOT



Assembling The Robot

See separate Assembling Instructions

Understanding How The Motors Work

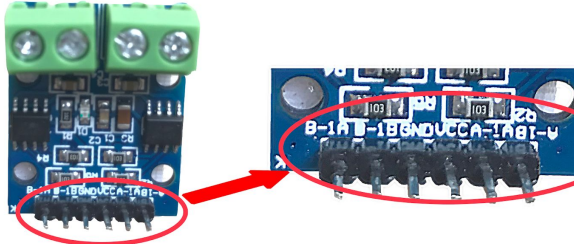

In order to control the motors, they have been connected to the microbit. Each motor is connected to two pins on the microbit via the motor driver L9110s and the Edge connector.

The right motor is connected to Pin 8 & Pin 12

The left motor is connected to Pin 0, Pin 16



The L9110S motor driver & Edge connector connections:

	
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 each motor move is to set one pin to HIGH (1) and 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 move the robot forward, both motors need to go forward, in Python this is how we would write it:

```
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, in Python this is how we would write it:

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

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

LET'S HAVE A GO AT WRITING THE ACTUAL PYTHON CODE TO MAKE THE ROBOT MOVE FORWARD AND BACKWARD IN THE PYTHON EDITOR ON THE MICROBIT WEBSITE -

The link below shows how to access Python Editor on Microbit.org website.-
Double Click on the picture to play the video or click on the link to see the Youtube video

https://youtu.be/6CN_C24W_hU



TASK 1

MAKE THE ROBOT GO FORWARD & BACKWARD:

Enter the code below. We will ask the robot to go continuously forward for 2 second and backward for 2 seconds

Python is case sensitive - Please carefully check each line of your code



```
1 from microbit import *
2
3 while True:
4     #Move the robot forward for 2 seconds
5     pin8.write_digital(1)
6     pin12.write_digital(0)
7     pin0.write_digital(1)
8     pin16.write_digital(0)
9     sleep(2000)
10
11     #Move the robot backward for 2 seconds
12     pin8.write_digital(0)
13     pin12.write_digital(1)
14     pin0.write_digital(0)
15     pin16.write_digital(1)
16     sleep(2000)
```

Give a name to your code:

Connect the microbit to your computer with the download cable.

If you need a refresher on how to download the code to the micro:bit and to save it, then view the video:

Double Click on the picture to play the video or click on the link to see the Youtube video: <https://youtu.be/ySR52WJfhn8> -



If the robot does not move, please carefully check your code and then your wiring.

Task 2

Let's 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.

To turn the robot to the right, motor B needs to go forwards and motor A backwards.

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



EXAMPLE CODE TURN RIGHT & LEFT:

Enter the code below. We will ask the robot to turn RIGHT for 2 second and LEFT for 2 seconds (continuously)

Python is case sensitive - Please carefully check each line of your code

```
1 from microbit import *
2
3 while True:
4     #Robot turns right for 2 second
5     pin8.write_digital(0)
6     pin12.write_digital(1)
7     pin0.write_digital(1)
8     pin16.write_digital(0)
9     sleep(2000)
10    #Robot turns left for 2 seconds
11    pin8.write_digital(1)
12    pin12.write_digital(0)
13    pin0.write_digital(0)
14    pin16.write_digital(1)
15    sleep(2000)
16
```

Connect the microbit to your computer with the download cable, download and save the code

NOTE: The robot will turn really fast. Do not worry about this for now. We will show you later how to adjust the speed.

How to Stop The Motor

We do this by setting 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)
```

Task 3

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 **write_analog** (NOT **write_digital**) commands. These can be set to a value between 0 (always off) to 1023 (always on). So for example, running a motor speed at 40% would have a value of 410 (40% of 1023).

To move the robot forward, at a 40% speed, we will need to write:

```
pin8.write_analog(410)
pin12.write_digital(0)
pin0.write_analog(410)
pin16.write_digital(0)
```

To move the robot smoothly to the right. We will set the left motor at a 58% speed ($1023 * 58\% = 593$), and the right motor at 10% speed ($1023 * 10\% = 102$). We will need to write:

```
pin8.write_digital(0)
pin12.write_analog(102)
pin0.write_analog(593)
pin16.write_digital(0)
```

WARNING! REVERSE IS DIFFERENT!

What about moving the motors Reverse at 40% ?

Doing this for the motors moving in reverse is a little different:

1. We need to change the second pin to 1 for reverse.
2. We then simply take the number (410 in this case, 40% of 1023) away from 1023, giving 613:

```
pin8.write_analog(613)
pin12.write_digital(1)
pin0.write_analog(613)
```



```
pin16.write_digital(1)
```

EXAMPLE CODE CHANGE OF MOTOR SPEED:

Enter the code below. We will ask the robot to MOVE FORWARD at 100% speed for 2 seconds, then continue MOVING FORWARD at 40% for 2 seconds , TURN SMOOTHLY to the RIGHT at 20% speed for 2 seconds, STOP for 2 seconds. REVERSE at 100% speed for 2 seconds, continue REVERSING at 40% for 2 seconds and STOP for 2 seconds.

Python is case sensitive - Please carefully check each line of your code

```
1 from microbit import *
2
3 while True:
4     #forward at 100% speed
5     pin8.write_digital(1)
6     pin12.write_digital(0)
7     pin0.write_digital(1)
8     pin16.write_digital(0)
9     sleep(2000)
10    #forward at 40% speed
11    pin8.write_analog(410)
12    pin12.write_digital(0)
13    pin0.write_analog(410)
14    pin16.write_digital(0)
15    sleep(2000)
16    #smooth right turn
17    pin8.write_digital(0)
18    pin12.write_analog(102)
19    pin0.write_analog(593)
20    pin16.write_digital(0)
21    sleep(2000)
22    #stop motors
23    pin8.write_digital(0)
24    pin12.write_digital(0)
25    pin0.write_digital(0)
26    pin16.write_digital(0)
27    sleep(2000)
```




```
28                                     #reverse at 100% speed
29 pin8.write_digital(0)
30 pin12.write_digital(1)
31 pin0.write_digital(0)
32 pin16.write_digital(1)
33 sleep(2000)
34                                     #reverse at 40% speed
35 pin8.write_analog(613)
36 pin12.write_digital(1)
37 pin0.write_analog(613)
38 pin16.write_digital(1)
39 sleep(2000)
40                                     #stop motors
41 pin8.write_digital(0)
42 pin12.write_digital(0)
43 pin0.write_digital(0)
44 pin16.write_digital(0)
45 sleep(2000)
46
```

Now that you understand how motors work, we can move on to some other projects!

Do not worry if you do not have time to finish the projects, you can continue after the event

Project 1 - Controlling the Microbit Robot using the Radio Control & the Accelerometer

We are now going to use a radio link to connect two Microbits which enables 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 picture 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).

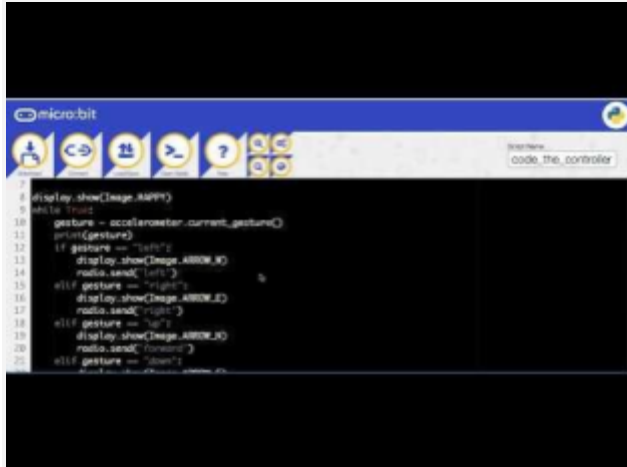
A radio signal (left, right, forward, reverse) will be sent to the 2nd Microbit.

Let's code 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

Note: If you want your radio signal to go a long way, you'll need to increase the power of the transmission (it goes from 0 to 7).

There are 100 channels available. Both controller and robot must have the same channel number.

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



<https://youtu.be/ltqHbUjPnq4>

CODE CONTROLLER:



```
1 # Add your Python code here. E.g.
2 from microbit import *
3 import radio
4 radio.config(channel=20)
5 radio.on()
6 radio.config(power=7)
7
8 display.show(Image.HAPPY)
9 while True:
10     gesture = accelerometer.current_gesture()
11     print(gesture)
12     if gesture == "left":
13         display.show(Image.ARROW_W)
14         radio.send('left')
15     elif gesture == "right":
16         display.show(Image.ARROW_E)
17         radio.send('right')
18     elif gesture == "up":
19         display.show(Image.ARROW_N)
20         radio.send('forward')
21     elif gesture == "down":
22         display.show(Image.ARROW_S)
23         radio.send('reverse')
24
25     elif button_a.was_pressed():
26         display.show(Image.ASLEEP)
27         radio.send('stop')
28
29
```

Let's code 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 robot” code to Microbit & save it - From the Python Editor on Microbit.org website.-

---> Follow the same process as for the “Code the controller”



CODE FOR ROBOT (RECEIVER)

```
1 from microbit import *
2 import radio
3 radio.config(channel=20)
4 radio.on()
5 radio.config(power=7)
6
7
8 while True:
9     incoming = radio.receive()
10    if incoming == "left":           # SMOOTH LEFT TURN
11        display.show(Image.ARROW_E)
12        pin8.write_analog(593)      # Right motor (Motor A) at 58% speed
13        pin12.write_digital(0)
14        pin0.write_digital(0)
15        pin16.write_analog(102)    # Left motor (Motor B) at 10% speed
16        sleep(100)
17    elif incoming == "right":       # SMOOTH RIGHT TURN
18        display.show(Image.ARROW_W)
19        pin8.write_digital(0)
20        pin12.write_analog(102)    # Right motor (Motor A) at 10% speed
21        pin0.write_analog(593)     # Left motor (Motor B) at 58% speed
22        pin16.write_digital(0)
23    elif incoming == "forward":     # MOVE FORWARD AT 100% speed
24        display.show(Image.ARROW_N)
25        pin8.write_digital(1)
26        pin12.write_digital(0)
27        pin0.write_digital(1)
28        pin16.write_digital(0)
29
30    elif incoming == "reverse":     # REVERSE AT 100% speed
31        display.show(Image.ARROW_S)
32        pin8.write_digital(0)
33        pin12.write_digital(1)
34        pin0.write_digital(0)
35        pin16.write_digital(1)
36
37    elif incoming == "stop":        # STOP MOTORS
38        display.show(Image.ASLEEP)
39        pin8.write_digital(0)
40        pin12.write_digital(0)
41        pin0.write_digital(0)
42        pin16.write_digital(0)
43
```

Project 2 - Ultrasonic Distance sensor



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 is 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()

How to download your “Avoid obstacle” code to Microbit & save it - From the Python Editor on Microbit.org website.-



---> Follow the same process as for the “Code the controller”

CODE FOR OBSTACLE AVOID:

```
1 from microbit import *
2 from utime import ticks_us, sleep_us, ticks_diff
3
4 display.show(Image.HAPPY)
5
6
7 def sonar( ):
8     pin13.write_digital(0) # Clear trigger
9     sleep_us(10)
10    pin13.write_digital(1) # Send 10us Ping pulse
11    sleep_us(10)
12    pin13.write_digital(0)
13    while pin14.read_digital() == 0: # ensure Ping pulse has cleared
14        pass
15    start = ticks_us() # define starting time
16    while pin14.read_digital() == 1: # wait for Echo pulse to return
17        pass
18    end = ticks_us() # define ending time
19    cm = ticks_diff(end, start) // 58 # Distance = Time divided by 58
20    return cm
21
22 while True:
23
24     if sonar() < 15:
25         #GO BACKWARD AT 100% SPED
26         pin8.write_digital(0)
27         pin12.write_digital(1)
28         pin0.write_digital(0)
29         pin16.write_digital(1)
30         sleep(1000)
31         #SMOOTH TURN RIGHT AR REDUCED SPEED
32         pin8.write_digital(0)
33         pin12.write_analog(102)
34         pin0.write_analog(593)
35         pin16.write_digital(0)
36         sleep(700)
37     else:
38         #GO FORWARD AT 100% SPEED
39         pin8.write_digital(1)
40         pin12.write_digital(0)
41         pin0.write_digital(1)
42         pin16.write_digital(0)
```