Omni:bit micro:bit Handle control

In this lesson we will learn to use the Handle to remotely control the building blocks Omni:bit.

About wiring

The left front motor is connected to the M1 interface of the Super:bit expansion board. The black line is on the battery side;

The left rear motor is connected to the M2 interface of the Super:bit expansion board, The black line is on the battery side;

The right front motor is connected to the M3 interface of the Super:bit expansion board, The black line is on the battery side;

The right rear motor is connected to the M4 interface of the Super:bit expansion board, The black line is on the battery side.

As shown below.
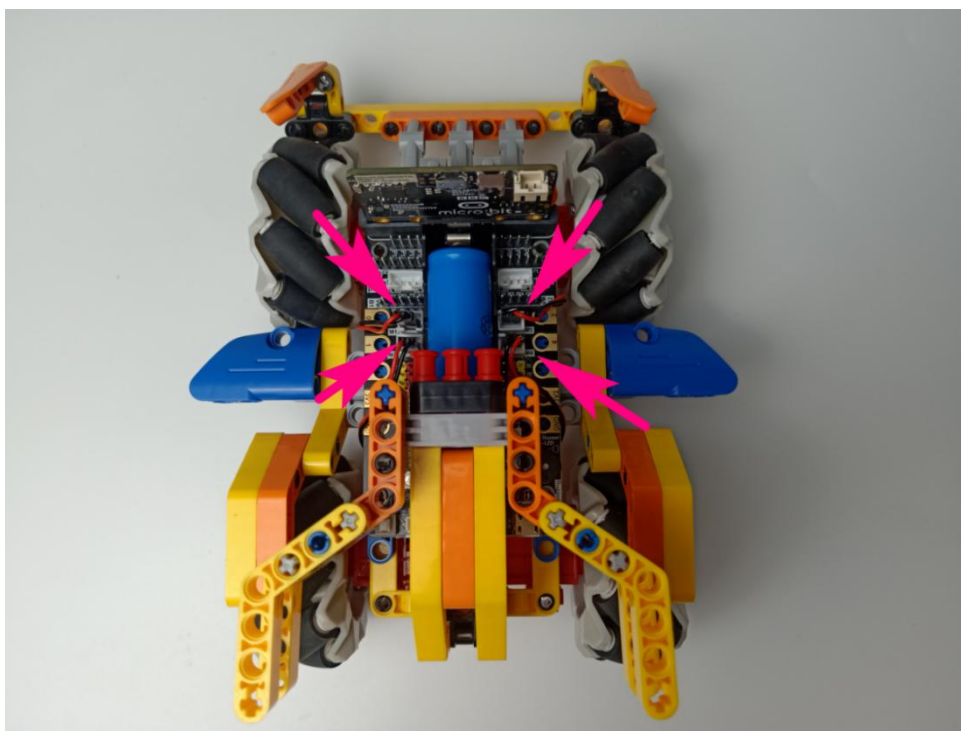
Wireless communication principles

With the micro:bit radio module, different devices can work together through a simple wireless network. When the radio function is turned on for micro:bit, a simple wireless local area network is generated. The micro:bit board with radio function turned on can set parameters within the effective range.

Wireless communication is divided into sending and receiving two program blocks. Set the wireless group of radio to the same group, and the two micro:bit boards can communicate.

About code

Car code:

Please use the MU software to open the Omnibit code.py file we provided.

1) Import the libraries neopixel, super:bit and radio to be used.

display.show(Image.HEART): Show a smile pattern on the micro:bit matrix;

radio.on (): Turn on the wireless function. Because the wireless function consumes more power and occupies memory, it is disabled by default. You can also use **radio.off ()** to turn off the wireless function.

radio.config (group = 1): configure wireless group = 1, so that other micro:bit devices with wireless group = 1 can communicate with each other, the default is 0, Range of group is 0 ~ 255.

np = neopixel.NeoPixel(pin12, 4): Define RGB lights on the expansion board, connect pin 12, there are 4 RGB lights in total.

!Note: the set group value It needs to be consistent with the handle setting, otherwise normal communication cannot be performed.

Code as shown below:

```
from microbit import *
import superbit
import radio
import neopixel

radio.on()
radio.config(group=1)
display.show(Image.HAPPY)
np = neopixel.NeoPixel(pin12, 4)
flag = 0
a = 0
```

2) Control the car advance, back, spin left and spin right functions:
incoming = radio.receive (): Receives the wirelessly transmitted data and saves it to the "incoming" variable.
if incoming is 'up', the car advance; 'down' the car back; 'left' the car turn left; 'right' makes the car turn right; And 'stop' makes the car stop.
Code as shown below:

```
incoming = radio.receive()
if incoming == 'up':
    superbit.motor_control_dual(superbit.M1, superbit.M3, -255, 255, 0)
    superbit.motor_control_dual(superbit.M2, superbit.M4, -255, 255, 0)
if incoming == 'down':
    superbit.motor_control_dual(superbit.M1, superbit.M3, 255, -255, 0)
    superbit.motor_control_dual(superbit.M2, superbit.M4, 255, -255, 0)
if incoming == 'left':
    superbit.motor_control_dual(superbit.M1, superbit.M3, 255, 255, 0)
    superbit.motor_control_dual(superbit.M2, superbit.M4, -255, -255, 0)
if incoming == 'right':
    superbit.motor_control_dual(superbit.M1, superbit.M3, -255, -255, 0)
    superbit.motor_control_dual(superbit.M2, superbit.M4, 255, 255, 0)
if incoming == 'left_rear':
    superbit.motor_control_dual(superbit.M1, superbit.M3, 255, 0, 0)
    superbit.motor_control_dual(superbit.M2, superbit.M4, 0, -255, 0)
if incoming == 'right_after':
    superbit.motor_control_dual(superbit.M1, superbit.M3, 0, -255, 0)
    superbit.motor_control_dual(superbit.M2, superbit.M4, 255, 0, 0)
if incoming == 'left_front':
    superbit.motor_control_dual(superbit.M1, superbit.M3, 0, 255, 0)
    superbit.motor_control_dual(superbit.M2, superbit.M4, -255, 0, 0)
if incoming == 'right_front':
    superbit.motor_control_dual(superbit.M1, superbit.M3, 0, 255, 0)
    superbit.motor_control_dual(superbit.M2, superbit.M4, -255, 0, 0)
if incoming == 'turn_off':
    superbit.motor_control_dual(superbit.M1, superbit.M3, 0, 0, 0)
    superbit.motor_control_dual(superbit.M2, superbit.M4, 0, 0, 0)
```

3) If incoming is 'R', the car headlights are red, 'G' is the car headlights are green, 'B' is the car headlights are blue, and 'Y' is the car headlights are yellow.
Code as shown below:

```
if incoming == 'R':
    np.clear()
    np[0] = (255, 0, 0)
    np[1] = (255, 0, 0)
    np[2] = (255, 0, 0)
    np[3] = (255, 0, 0)
    np.show()
elif incoming == 'G':
    np[0] = (0, 255, 0)
    np[1] = (0, 255, 0)
    np[2] = (0, 255, 0)
    np[3] = (0, 255, 0)
    np.show()
elif incoming == 'B':
    np[0] = (0, 0, 255)
    np[1] = (0, 0, 255)
    np[2] = (0, 0, 255)
    np[3] = (0, 0, 255)
    np.show()
elif incoming == 'Y':
    np.clear()
    np[0] = (255, 255, 0)
    np[1] = (255, 255, 0)
    np[2] = (255, 255, 0)
    np[3] = (255, 255, 0)
    np.show()
```

!Note:
The value of incoming needs to correspond to the value sent by the handle. Only the same value can receive and execute the command.

Handle control code:
Please use the MU software to open the **Handle code.py** file we provided.

1) Import the libraries microbit, ghandle, and radio that you need to use.
display.show(0): Show number 0 on the micro:bit matrix;

radio.on (): Turn on the wireless function;

radio.config (group = 1): set wireless group = 1, which is consistent with the group of the car;

Code as shown below:

```
from microbit import *
import ghandle
import radio

display.show(0)
radio.on()
radio.config(group=1)
```

2) If it detects that **ghandle.rocker (ghandle.up)** is True, it means that the rocker of the handle is pushed up, and the 'up' command is sent wirelessly, and an upward icon is displayed on LED dot matrix.

If it detects that **ghandle.rocker(ghandle.down)** is True, it means that the rocker of the handle is pushed down, and the 'down' command is sent wirelessly, and an down icon is displayed on LED dot matrix.

If it detects that **ghandle.rocker(ghandle.left)** is True, it means that the rocker of the handle is pushed left, and the 'left' command is sent wirelessly, and an left icon is displayed on LED dot matrix.

If it detects that **ghandle.rocker(ghandle.right)** is True, it means that the rocker of the handle is pushed right, and the 'right' command is sent wirelessly, and an right icon is displayed on LED dot matrix.

If it detects that **ghandle.rocker(ghandle. pressed)** is True, it means that the rocker of the handle is pressed, and the 'pressed' command is sent wirelessly, and an "X" icon is displayed on LED dot matrix.

If it does not operate to send 'stop' and clear the display.

Determine whether the button is pressed. The commands 'R', 'G', 'B', 'Y' are sent for B1 (red), B2 (green), B3 (blue), and B4 (yellow).

Code as shown below:

```python
while True:
    x = pin2.read_analog() * -1 + 512
    y = pin1.read_analog() * -1 + 512
    if ghandle.rocker(ghandle.up):
        radio.send('up')
        display.show(Image.ARROW_N)
    elif ghandle.rocker(ghandle.down):
        radio.send('down')
        display.show(Image.ARROW_S)
    elif ghandle.rocker(ghandle.left):
        radio.send('left')
        display.show(Image.ARROW_W)
    elif ghandle.rocker(ghandle.right):
        radio.send('right')
        display.show(Image.ARROW_E)
    elif ghandle.rocker(ghandle.pressed):
        radio.send('turn_off')
        display.show(Image.NO)
    elif x < -100 and y < -100:
        radio.send('left_rear')
    elif x > 100 and y < -100:
        radio.send('right_after')
    elif x < -100 and y > 100:
        radio.send('left_front')
    elif x > 100 and y > 100:
        radio.send('right_front')
    else:
        radio.send('stop')
        display.clear()
    if ghandle.B1_is_pressed():
        radio.send('R')
```

```python
        display.show("R")
    if ghandle.B2_is_pressed():
        radio.send('G')
        display.show("G")
    if ghandle.B3_is_pressed():
        radio.send('B')
        display.show("B")
    if ghandle.B4_is_pressed():
        radio.send('Y')
        display.show("Y")
```
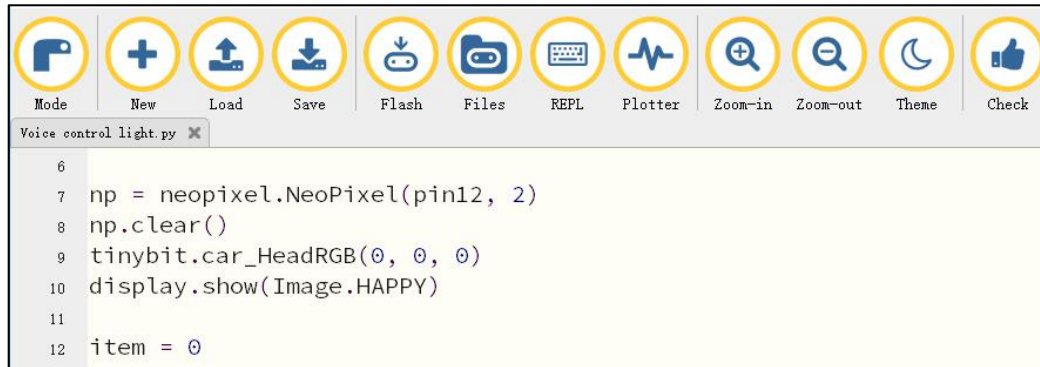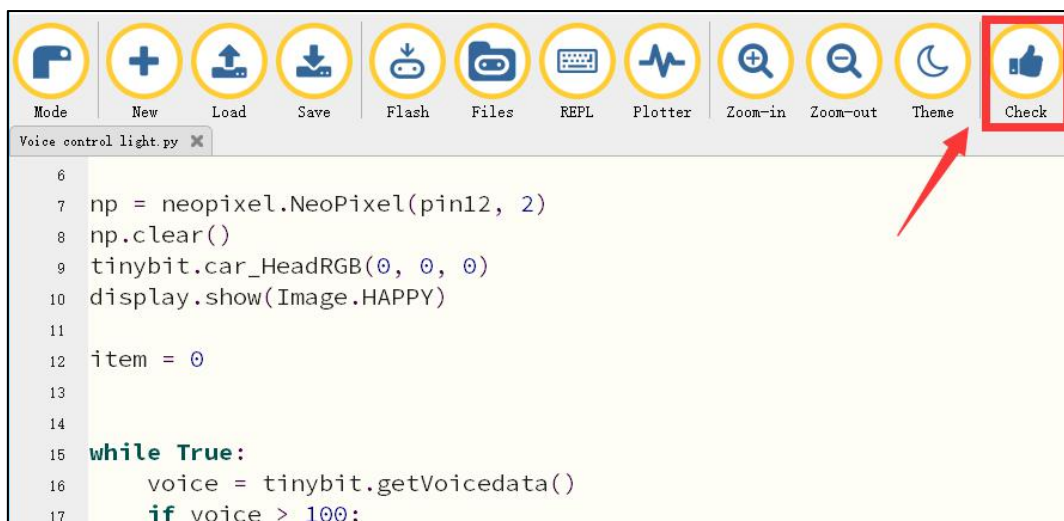
Programming and downloading

1.You should open the Mu software, and enter the code in the edit window, , as shown below.

Note! All English and symbols should be entered in English, and the last line must be a space.


```
6
7   np = neopixel.NeoPixel(pin12, 2)
8   np.clear()
9   tinybit.car_HeadRGB(0, 0, 0)
10  display.show(Image.HAPPY)
11
12  item = 0
```

2.You can click the **"Check"** button to check if our code has an error. If a line appears with a cursor or an underscore, the program indicating this line is wrong.


```
6
7   np = neopixel.NeoPixel(pin12, 2)
8   np.clear()
9   tinybit.car_HeadRGB(0, 0, 0)
10  display.show(Image.HAPPY)
11
12  item = 0
13
14
15  while True:
16      voice = tinybit.getVoicedata()
17      if voice > 100:
```

3.Click **"REPL"** button,check whether the tinybit library has been downloaded. If not, please refer to the [preparation before class]---> [Python programming]
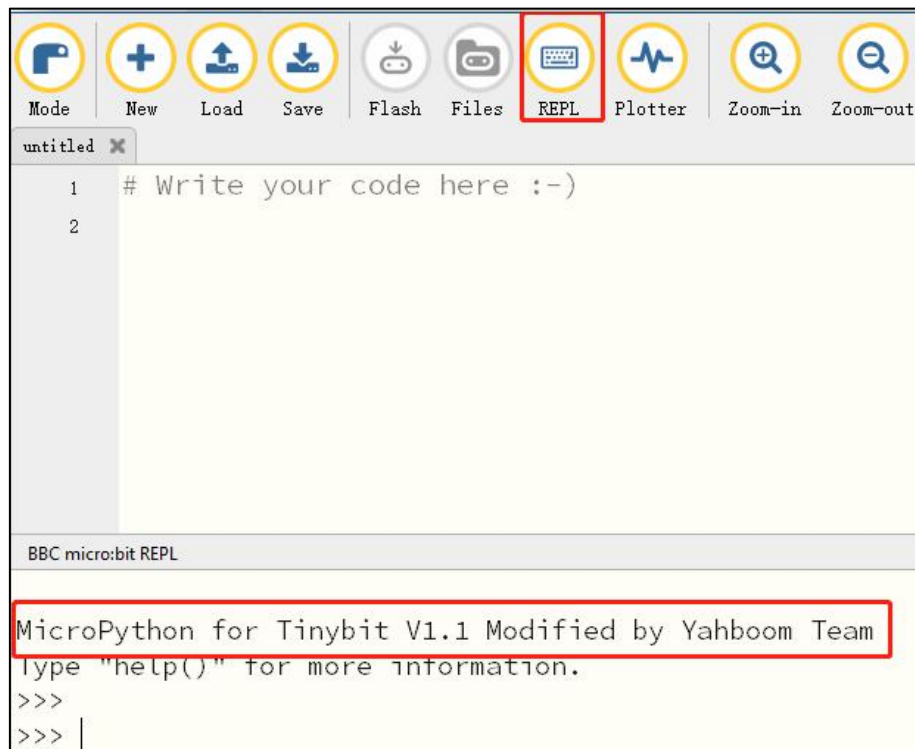
4.Click the **"Flash"** button to download the program to micro:bit board.

If the program is wrong or the experimental phenomenon is wrong after downloading, please confirm whether you have downloaded the Superbit library hex file we provided to the micro: bit board.
For the specific method of adding library files, please refer to 【1.Preparation before class】---【Python programming】

Experimental phenomena
After download is complete,
You can see that the micro:bit dot matrix shows an pattern as shown in Figure 1.1.
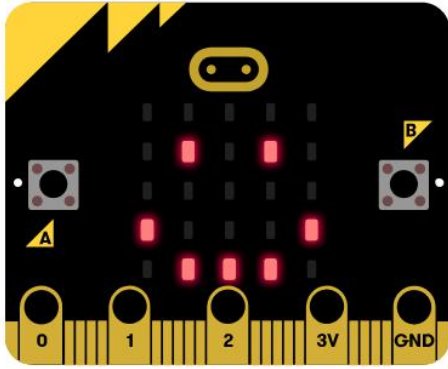You can see that the micro:bit dot matrix shows an pattern as shown in Figure 1.2.
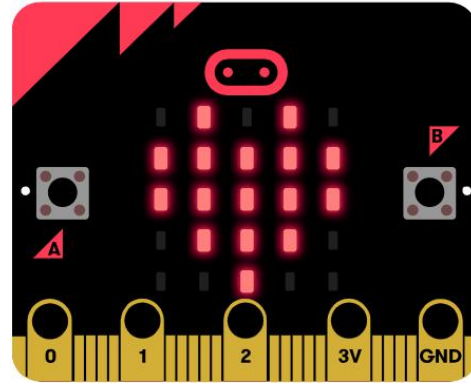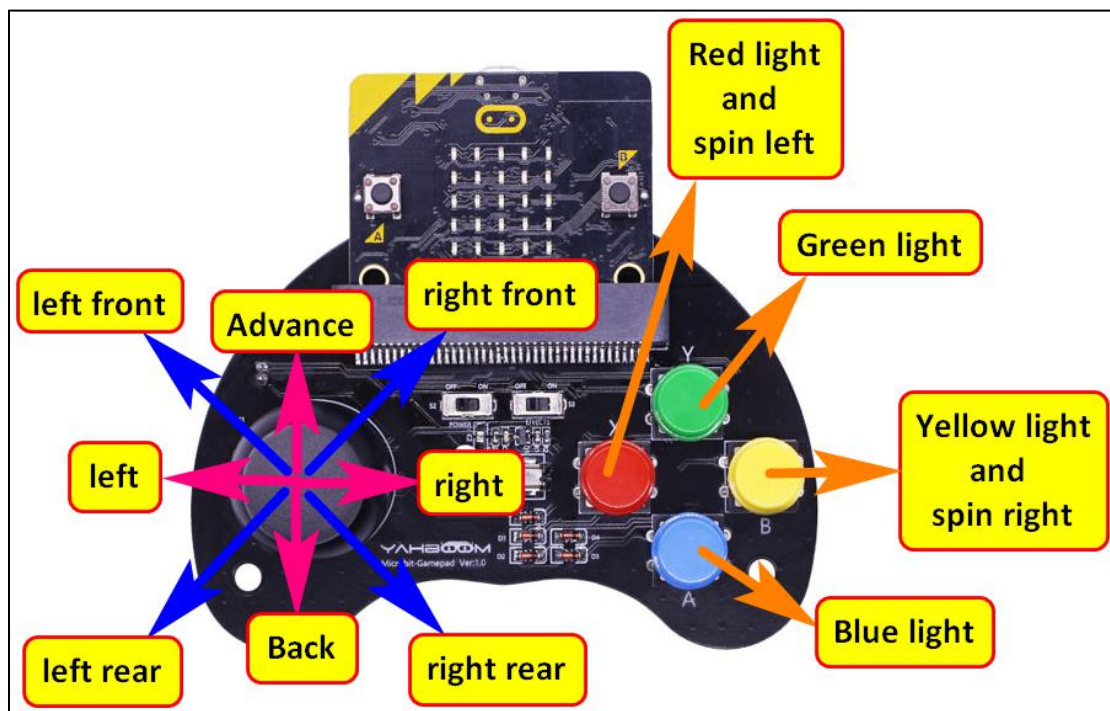
Figure 1.1

Figure 1.2

Handle control function

!Note: Press rocker to stop the car.