Lesson 2 Ultrasonic Detecting and Stacking

1. Working Principle

Ultrasonic sensor can convert ultrasonic signals into other energy signals (usually electrical signals). There are two probes on ultrasonic sensor for receiving and transmitting ultrasound.

Firstly, import the corresponding library and initialize ultrasonic sensor, buzzer, servo and action groups.

Next, the object is detected by ultrasonic sensor and the measured distance is read by I2C protocol. After determining the distance, MaxArm will perform the corresponding action based on the determined result.

Then, execute the functions for controlling action group, buzzer and air pump to suck the detected object to the side and stack it.

The path of the program file: “6. Secondary Development /Python Development/Sensor-extension Game/Program Files/Ultrasonic Detecting and stacking/main.py”.

```
import time
from machine import Pin, I2C
from Ultrasonic import ULTRASONIC
from Buzzer import Buzzer
from espmax import ESPMax
from PWMServo import PWMServo
from BusServo import BusServo
from RobotControl import RobotControl
from SuctionNozzle import SuctionNozzle



pwm = PWMServo()
buzzer = Buzzer()
bus_servo = BusServo()
arm = ESPMax(bus_servo)
robot = RobotControl()
nozzle = SuctionNozzle()

```

2. Preparation

2.1 Hardware

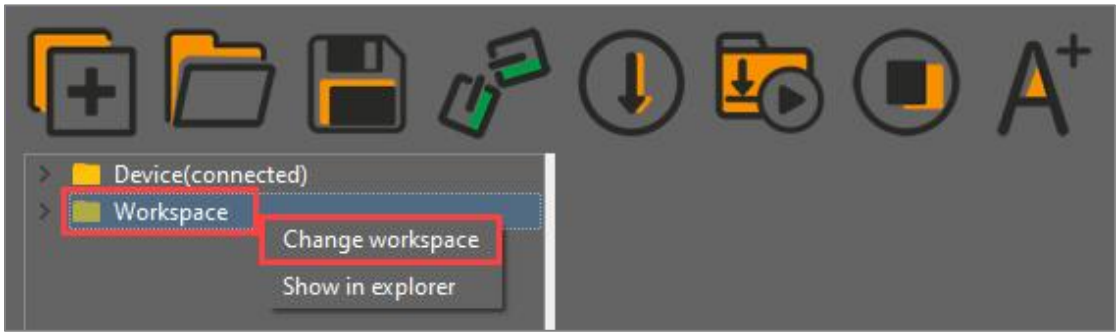
Please assemble the ultrasonic sensor to the corresponding position on MaxArm according to the tutorial in folder “Lesson 1 Sensor Assembly” under the same directory.

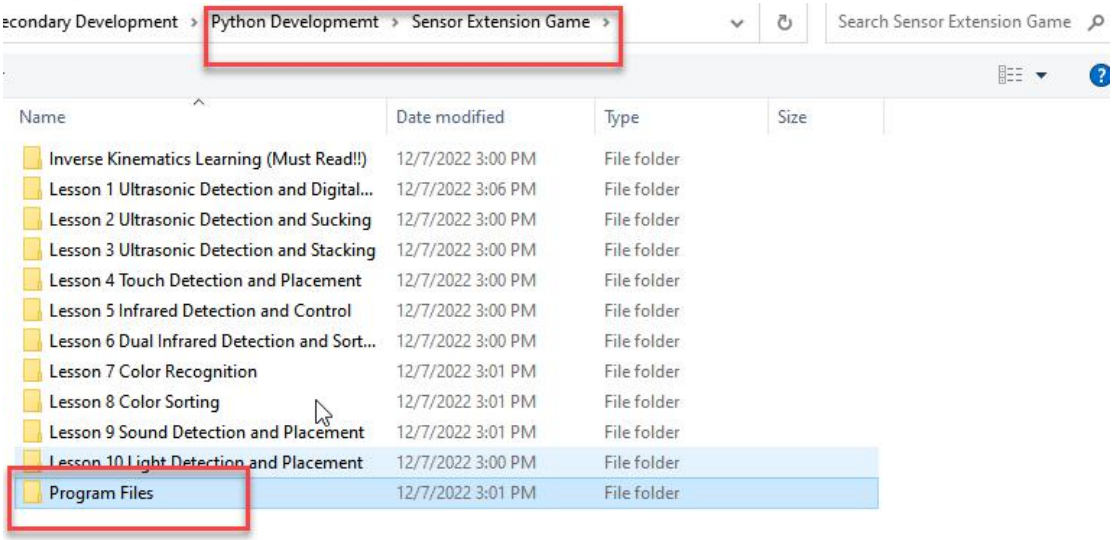
2.2 Software

Please connect MaxArm to Python editor according to the tutorial in folder “4. Underlying Program Learning/Python Development/Lesson 1 Set Development Environment”.

3. Program Download

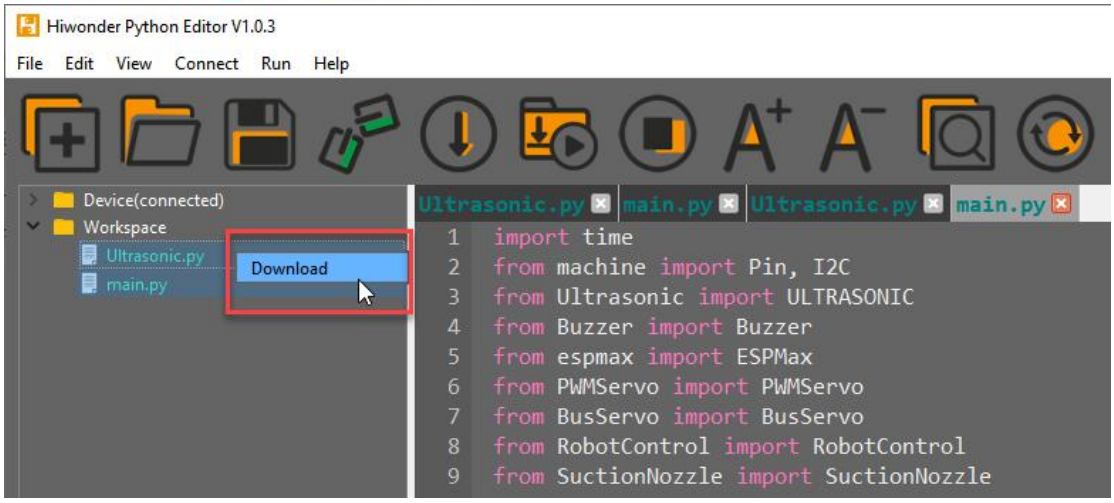
1) After connecting, change the path of Workspace to “6. Secondary Development/ Python Development/Sensor-extension Game”, and select “Program Files”.



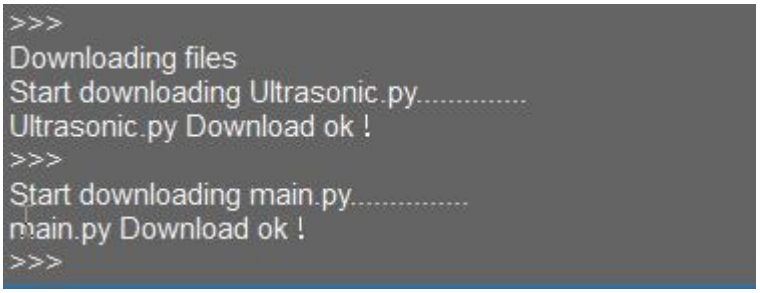


2) Click the folder “Ultrasonic Detecting and Stacking”, and then select all the program files in the folder.

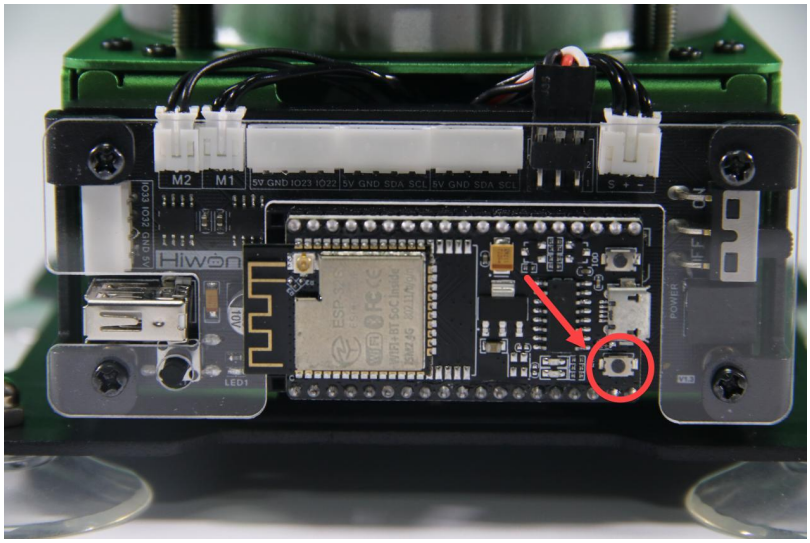
3) Then right click and select “Download” to download all the program files to the controller.



4) When the terminal prints the prompt as shown in the image below, it means download completed.



5) After downloading, click on the reset icon or press the reset button on ESP32 controller to run program.



4. Project Outcome

After putting the block on the fixed detection position, the buzzer will beep. Then MaxArm will suck the block to the left and stack it. The program will be repeated after stacking the three blocks.

5. Program Instruction

5.1 Import library file

The path of the program file: “6. Secondary Development/Python Development/Sensor-extension Game/Program Files/Ultrasonic Detecting and stacking/main.py”.

Before carrying out detection and suction, the Python function libraries related to I2C protocol, ultrasonic sensor, buzzer, PWM servo, bus servo, air pump and other related Python function libraries are imported.

```
1 import time
2 from machine import Pin, I2C
3 from Ultrasonic import ULTRASONIC
4 from Buzzer import Buzzer
5 from espmax import ESPMax
6 from PWMServo import PWMServo
7 from BusServo import BusServo
8 from RobotControl import RobotControl
9 from SuctionNozzle import SuctionNozzle
```

5.2 Ultrasonic detection

Use the hwsr06.getDistance() function to measure distance. Then use print() function to print out the measured distance.

```
40     Distance = hwsr06.getDistance()
41     print("distance = ", Distance)
```

5.3 Control robotic arm

Determine whether there are objects at the distance between 70 and 80, and then perform the corresponding action.

```
    if 70 < Distance < 80:
        buzzer.setBuzzer(100)
        time.sleep_ms(1000) #
        arm.set_position((0,-160,85),1500)
        nozzle.on()
        time.sleep_ms(1600)
        arm.set_position((0,-160,200),1000)
        time.sleep_ms(1000)
        arm.set_position((160,0,200),1500)
        time.sleep_ms(1500)
        arm.set_position((160,0,(88+overlay*40)),1000)

        time.sleep_ms(1200)
        nozzle.off()
        arm.set_position((160,0,200),1000)
        time.sleep_ms(1000)
        arm.go_home()
        time.sleep_ms(2000)
        overlay += 1
        if overlay >= 3: overlay = 0
```

Use the buzzer.setBuzze() function to control the buzzer. Take the code “buzzer.setBuzze()” as example.

The first parameter “100” represents the sounding time of the buzzer and the unit is ms.

Use the arm.set_position() function to control robotic arm. Take the code “arm.set_position((0,-160,85),1500)” as example.

The first parameter “(0，-160，85)” is the position of the suction nozzle on x, y and z axes.

The second parameter “1500” represents the running time and the unit is ms.