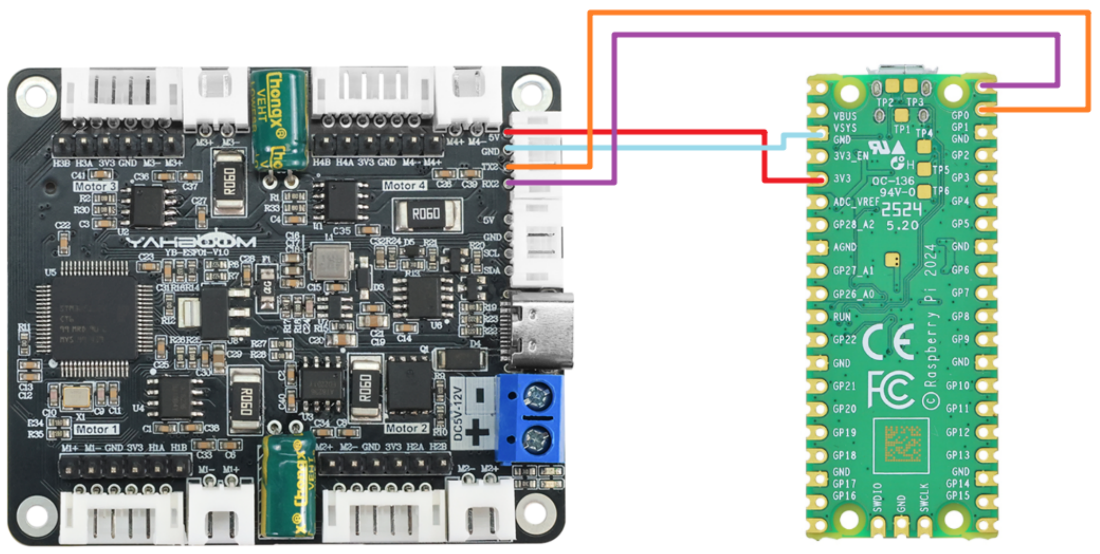
Drive motor and read encoder-USART

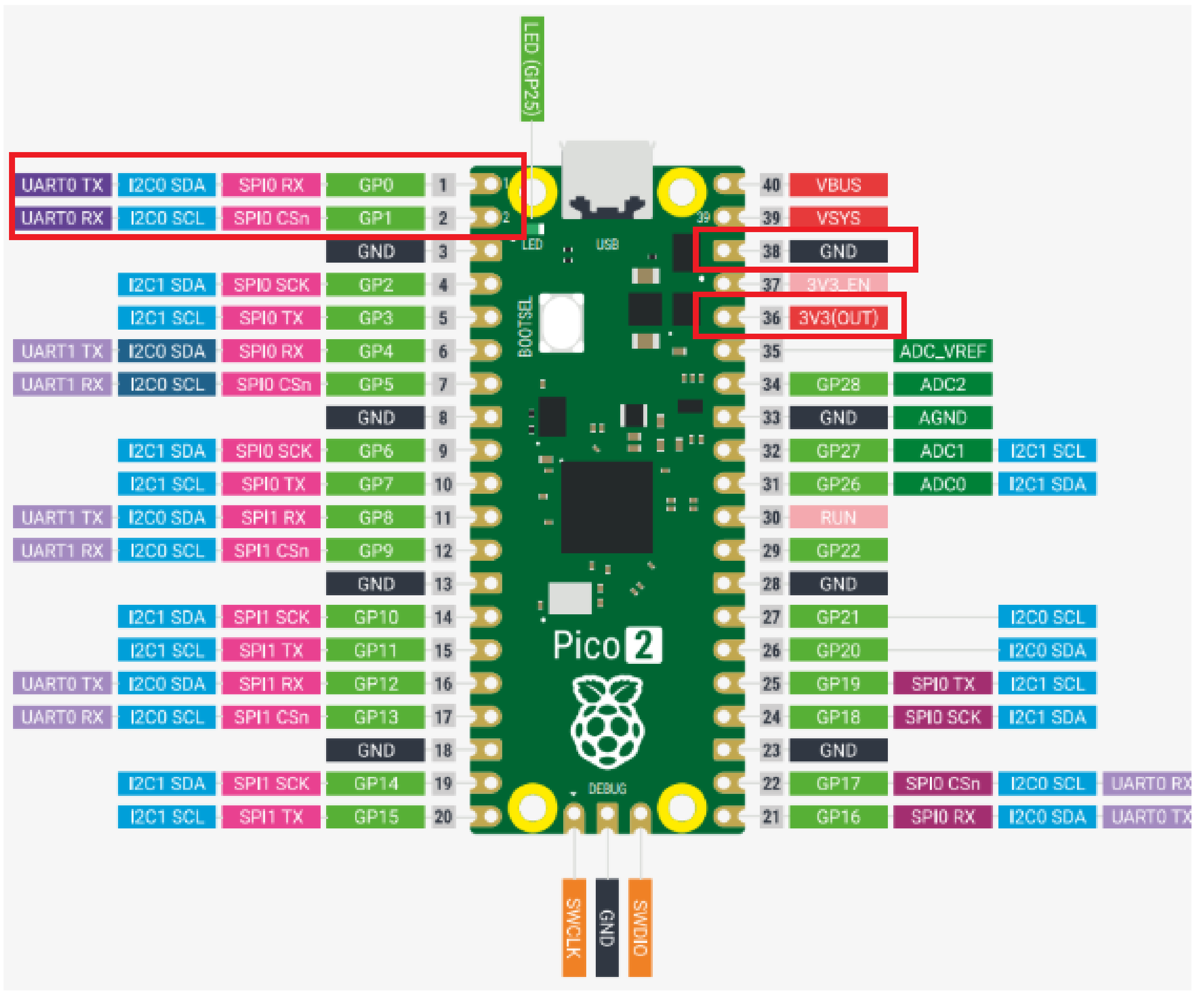
1.1 Explanation

Please read 《0. Motor introduction and usage》first to understand the motor parameters, wiring method, and power supply voltage you are currently using. To avoid improper operation and damage to the driver board or motor.

I2C and serial communication cannot be shared, only one can be selected.

Hardware wiring:



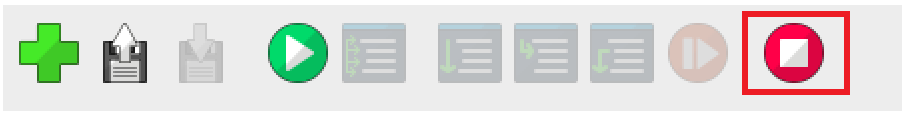


Motor	4-channel motor drive board(Motor)
M2	M-
V	3V3
A	H1A
B	H1B
G	GND
M1	M+

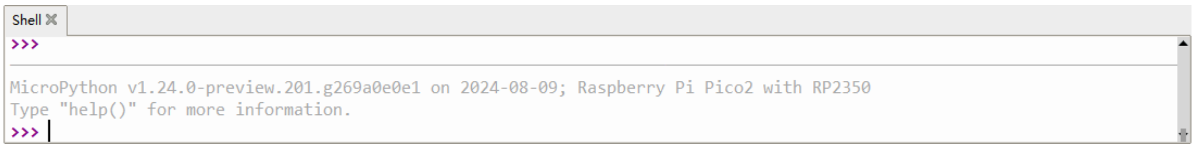
4-channel motor drive board	Pico2
RX2	GP0
TX2	GP1
GND	GND
5V	3V3

1.2 Instructions

Open the code using Thonny software, connect to the Raspberry Pi pico2, and click the stop button on the far right of the upper toolbar.

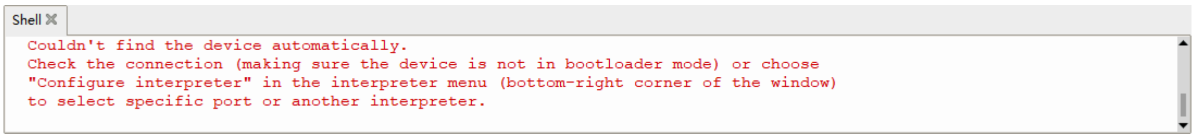


Then, you can see the information of the current firmware in pico2 pop up in the message bar at the bottom, which means that the software has recognized the serial port.



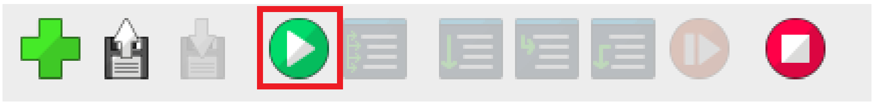
```
>>>

MicroPython v1.24.0-preview.201.g269a0e0e1 on 2024-08-09; Raspberry Pi Pico2 with RP2350
Type "help()" for more information.
>>> 
```

If the message bar here shows that the serial port cannot be recognized, you need to refer to the firmware flashing tutorial and flash the firmware to the motherboard so that the serial port can be recognized.



```
Couldn't find the device automatically.
Check the connection (making sure the device is not in bootloader mode) or choose
"Configure interpreter" in the interpreter menu (bottom-right corner of the window)
to select specific port or another interpreter.
```

After successfully identifying the serial port and printing the firmware information, click the green button to start running the script.



1.3 Code analysis

```
UPLOAD_DATA = 3  #0:不接受数据 1:接收总的编码器数据 2:接收实时的编码器 3:接收电机当前速度 mm/s
                 #0: Do not receive data 1: Receive total encoder data 2: Receive real-time encoder 3: Receive current motor speed mm/s

MOTOR_TYPE = 1  #1:520电机 2:310电机 3:测速码盘TT电机 4:TT直流减速电机 5:L型520电机
                #1:520 motor 2:310 motor 3:speed code disc TT motor 4:TT DC reduction motor 5:L type 520 motor
```

- UPLOAD_DATA: used to set the data of the motor encoder. Set 1 to the total number of encoder pulses and 2 to the real-time pulse data of 10ms.
- MOTOR_TYPE: used to set the type of motor used. Just modify the corresponding numbers according to the comments according to the motor you are currently using. You don't need to modify the rest of the code.

If you need to drive the motor and observe the data, just modify the two numbers at the beginning of the program. No changes are required to the rest of the code.

```
def set_motor_parameter():

    if MOTOR_TYPE == 1:
        set_motor_type(1)  # 配置电机类型
        time.sleep(0.1)
        set_pluse_phase(30)  # 配置减速比，查电机手册得出
        time.sleep(0.1)
        set_pluse_line(11)  # 配置磁环线，查电机手册得出
        time.sleep(0.1)
        set_wheel_dis(67.00)  # 配置轮子直径，测量得出
        time.sleep(0.1)
        set_motor_deadzone(1600)  # 配置电机死区，实验得出
        time.sleep(0.1)

    elif MOTOR_TYPE == 2:
        set_motor_type(2)
        time.sleep(0.1)
        set_pluse_phase(20)
        time.sleep(0.1)
        set_pluse_line(13)
        time.sleep(0.1)
        set_wheel_dis(48.00)
        time.sleep(0.1)
        set_motor_deadzone(1200)
        time.sleep(0.1)

    elif MOTOR_TYPE == 3:
        set_motor_type(3)
        time.sleep(0.1)
        set_pluse_phase(45)
        time.sleep(0.1)
        set_pluse_line(13)
        time.sleep(0.1)
        set_wheel_dis(68.00)
        time.sleep(0.1)
        set_motor_deadzone(1250)
        time.sleep(0.1)

    elif MOTOR_TYPE == 4:
        set_motor_type(4)
        time.sleep(0.1)
        set_pluse_phase(48)
        time.sleep(0.1)
        set_motor_deadzone(1000)
        time.sleep(0.1)

    elif MOTOR_TYPE == 5:
        set_motor_type(1)
        time.sleep(0.1)
        set_pluse_phase(40)
        time.sleep(0.1)
        set_pluse_line(11)
        time.sleep(0.1)
        set_wheel_dis(67.00)
        time.sleep(0.1)
        set_motor_deadzone(1600)
        time.sleep(0.1)
```

This is used to store the parameters of the Yahboom motor. By modifying the MOTOR_TYPE parameter above, one-click configuration can be achieved.

In normally, do not modify the code here when using the Yahboom motor.

If you are using your own motor, or if a certain data needs to be modified according to your needs, you can check the course《1.2 Control command》 to understand the specific meaning of each command.

```
if __name__ == "__main__":
    try:
        t = 0
        print("please wait...")
        send_upload_command(UPLOAD_DATA)#给电机模块发送需要上报的数据 Send the data 
that needs to be reported to the motor module
        time.sleep(0.1)
        set_motor_parameter()#设计电机参数  Design motor parameters

        while True:
            received_message = receive_data()  # 接收消息    Receiving Messages
            if received_message:    # 如果有数据返回 If there is data returned
                parsed = parse_data(received_message) # 解析数据 Parsing the data
                if parsed:
                    print(parsed)  # 打印解析后的数据   Print the parsed data

            t += 10
            M1 = t
            M2 = t
            M3 = t
            M4 = t

            if MOTOR_TYPE == 4:
                control_pwm(M1*2, M2*2, M3*2, M4*2)
            else:
                control_speed(M1, M2, M3, M4)#直接发送命令控制电机  Send commands 
directly to control the motor

            if t> 1000 or t < -1000:
                t = 0

            time.sleep(0.1)
```

In the loop program, the speed of the four motors will be slowly increased from 0 to 1000. If the motor type is 4, that is, the motor without encoder, the motor's PWM is directly controlled.

At the same time, the data sent by the driver board is read and printed out at the same time.

```
# 接收数据  Receiving Data
def receive_data():
    global recv_buffer
    if uart.any() > 0:  # 检查串口缓冲区是否有数据  Check if there is data in the 
serial port buffer
        recv_buffer += uart.read(uart.any()).decode()  # 读取并解码数据  Read and 
decode data
```

```
        # 按结束符 "#" 分割消息 Split the message by the ending character "#"
        messages = recv_buffer.split("#")
        recv_buffer = messages[-1]

        if len(messages) > 1:
            return messages[0] + "#"  #返回一条完整的消息   Return a complete message
    return None

# 解析接收到的数据  Parsing received data
def parse_data(data):
    data = data.strip()  # 去掉两端的空格或换行符   Remove spaces or line breaks at both ends

    if data.startswith("$MAll:"):
        values_str = data[6:-1]  # 去除 "$MAll:" 和 "#" Remove "$MAll:" and "#"
        values = list(map(int, values_str.split(',')))  # 分割并转换为整数  Split and convert to integer
        parsed = ', '.join([f"M{i+1}:{value}" for i, value in enumerate(values)])
        return parsed
    elif data.startswith("$MTEP:"):
        values_str = data[6:-1]
        values = list(map(int, values_str.split(',')))
        parsed = ', '.join([f"M{i+1}:{value}" for i, value in enumerate(values)])
        return parsed
    elif data.startswith("$MSPD:"):
        values_str = data[6:-1]
        values = [float(value) if '.' in value else int(value) for value in values_str.split(',')]
        parsed = ', '.join([f"M{i+1}:{value}" for i, value in enumerate(values)])
        return parsed
```

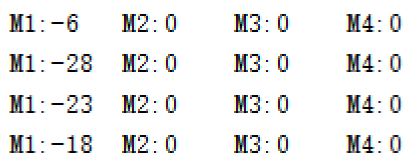
After getting the data from the driver board, it is shifted, and the shift is required to get the correct data.

1.4 Experimental phenomenon

After the wiring is correct, plug the motherboard into the USB port of the computer, and then click the red stop button on the software. Under normal circumstances, you can see the firmware information pop up in the message bar below.

Then, click the green run button, and you can see that the motor will gradually speed up, then stop, and repeat.

At the same time, you can see the printed motor value in the message bar constantly changing.



```
M1:-6   M2:0   M3:0   M4:0
M1:-28  M2:0   M3:0   M4:0
M1:-23  M2:0   M3:0   M4:0
M1:-18  M2:0   M3:0   M4:0
```