**Project 32：RFID**

1. **Introduction**

Nowadays, many residential districts use this function to open the door by swiping the card, which is very convenient. In this lesson, we will learn how to use RFID(radio frequency identification) wireless communication technology and read and write the key chain card (white card) and control the steering gear rotation by RFID-MFRC522 module.

1. **Components Required**

|  |  |  |  |
| --- | --- | --- | --- |
| 17a6d7f241a04d4e932cb06b758197c4 | _DSC2552 | RFID 新 | **1(4)** |
| Raspberry Pi Pico\*1 | Raspberry Pi Pico Expansion Board\*1 | RFID-MFRC522 Module\*1 | Key Chain\*1 |
| 杜邦线-3 |  | **1(4)** |  |
| F-F Dupont Wires | Servo\*1 | White Card\*1 | USB Cable\*1 |

1. **Component Knowledge**

**RFID**：RFID (Radio Frequency Identification) is a wireless communication technology. A complete RFID system generally consists of a transponder and a reader. Usually we use tags as transponders, and each tag has a unique code attached to the object to identify the target object. The reader is a device that reads (or writes) tag information.

Products derived from RFID technology can be divided into three categories: passive RFID products, active RFID products and semi-active RFID products. However, the passive RFID products are the earliest, most mature and most widely used products on the market, which can be seen everywhere in our daily life, such as bus card, meal card, bank card, hotel access card, etc., which are close contact identification. The main operating frequencies of the passive RFID products are 125KHZ(low frequency), 13.56mhz (high frequency), 433MHZ(UHF), and 915MHZ(UHF). The active and the semi-active RFID products operate at higher frequencies.

The RFID module we use is a passive RFID product with a working frequency of 13.56MHz.

**RFID-RC522 Module：**The MFRC522 is a highly integrated reader/writer IC for 13.56MHz contactless communication. Its internal transmitter is capable of driving a read/write antenna , which is designed to communicate with ISO/IEC 14443A /MIFARE cards and transponders without the need for additional active circuits . The receiving module provides an efficient implementation of demodulation and decoding of signals from ISO/IEC 14443 A /MIFARE compatible cards and transponders. The digital module manages complete ISO/IEC 14443A framing and error detection (parity and CRC) features.

The RFID module uses the MFRC522 as the control chip and adopts I2C (Inter-Integrated Circuit) interface.



Specifications:

Operating voltage: DC 3.3V-5V

Operating current: 13—100mA/DC 5V

Idling current: 10-13mA/DC 5V

Sleep current: <80uA

Peak current: <100mA

Operating frequency: 13.56MHz

Maximum power: 0.5W

Supported card types: mifare1 S50、mifare1 S70、mifare UltraLight、mifare Pro、mifare Desfire

Environmental operating temperature: -20 to 80 degrees Celsius

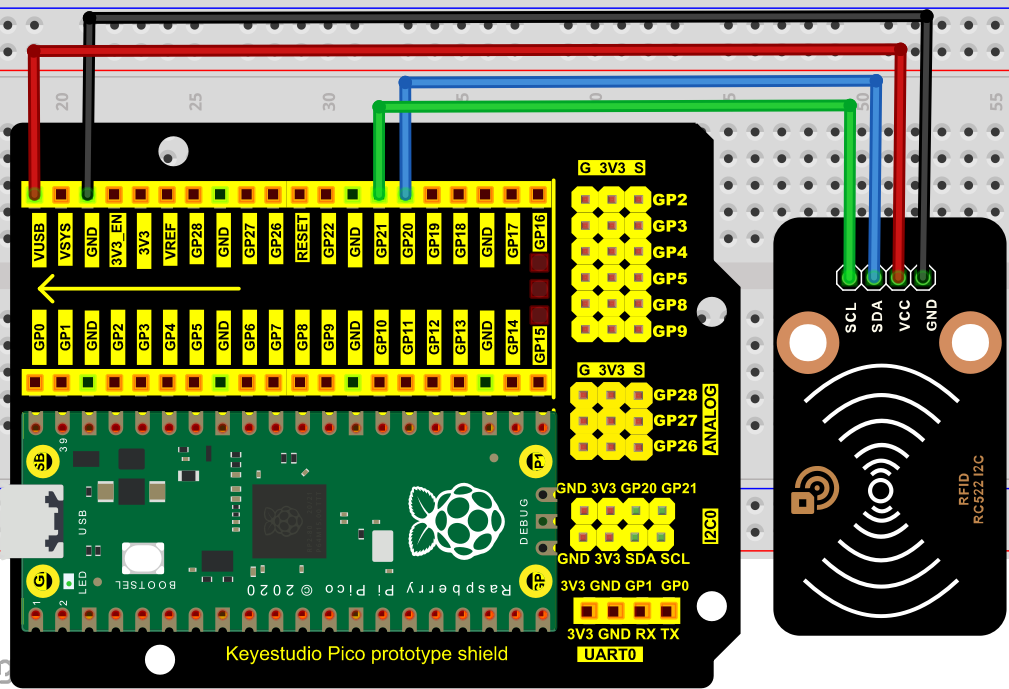
Environment storage temperature: -40 to 85 degrees Celsius

Relative Humidity: 5% to 95%

Data transfer rate: The maximum is 10Mbit/s.

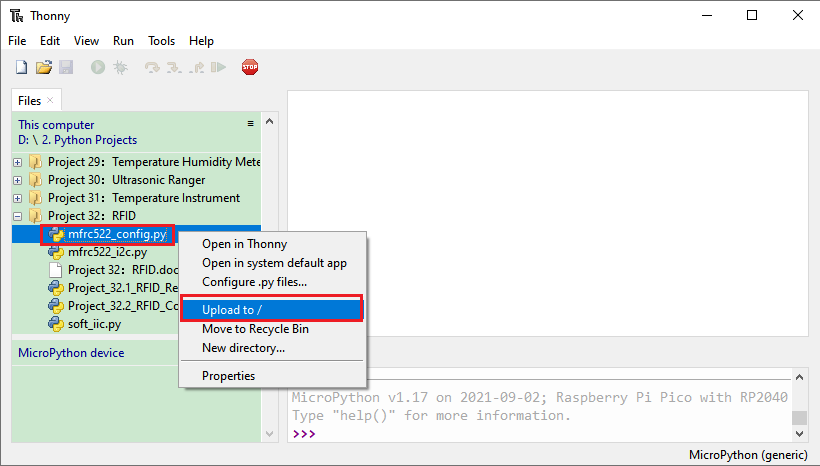
1. **Read the Card Number Value**

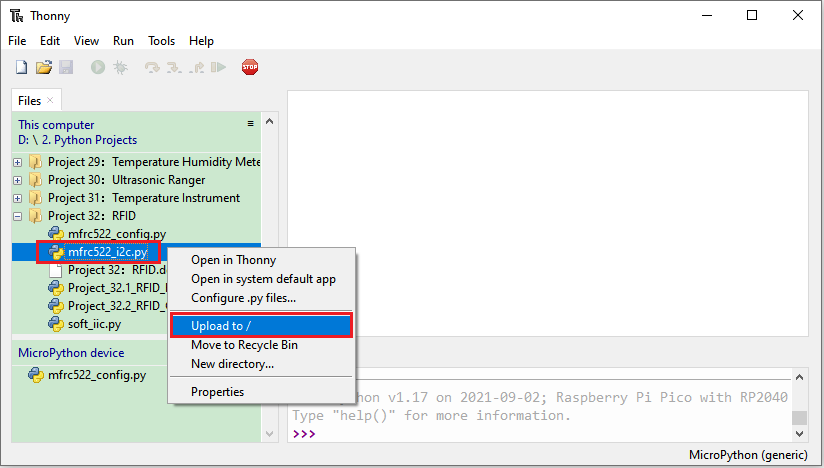
We will read the UNIQUE ID number (UID) of the RFID card and identify its type . And display relevant information through the "Shell" window of Thonny IDE. The wiring diagram is as follows:

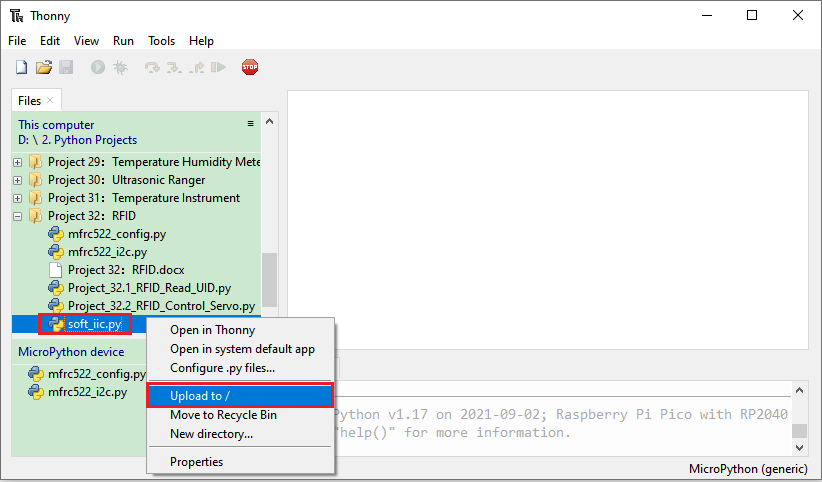


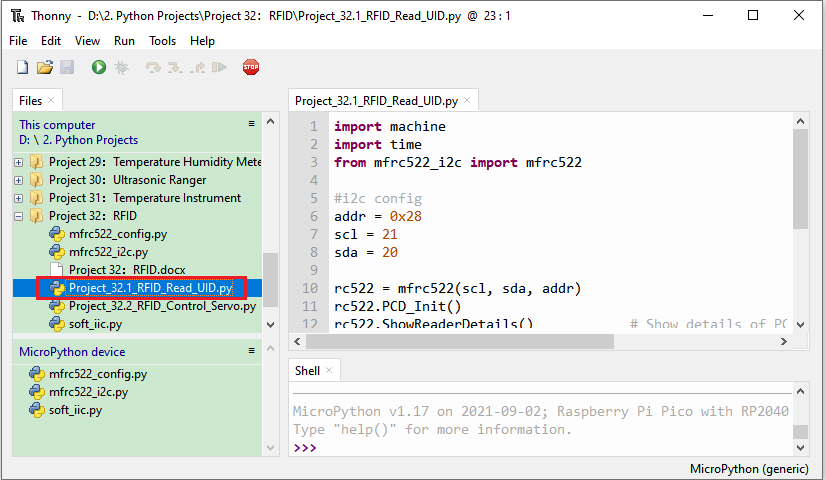
The code used in this project is saved in the file KS3020 Keyestudio Raspberry Pi Pico Learning Kit Ultimate Edition\2. Windows System\1. Python\_Tutorial\2. Python Projects\Project 32：RFID. You can move the code to anywhere, for example, we can save the code in the Disk(D), the route is D:\2. Python Projects.

Open Thonny, click“This computer”→“D:”→“2. Python Projects”→“Project 32：RFID”. Select“mfrc522\_config.py”“mfrc522\_i2c.py”and“soft\_iic.py”，right-click and select“Upload to /”,waiting for the“mfrc522\_config.py”“mfrc522\_i2c.py”and“soft\_iic.py”to be uploaded to the Raspberry Pi Pico. And double left-click the“Project\_32.1\_RFID\_Read\_UID.py”.



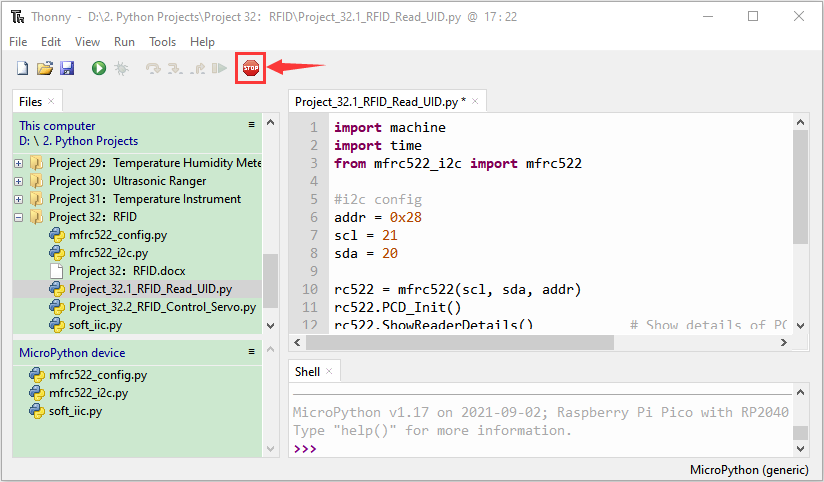






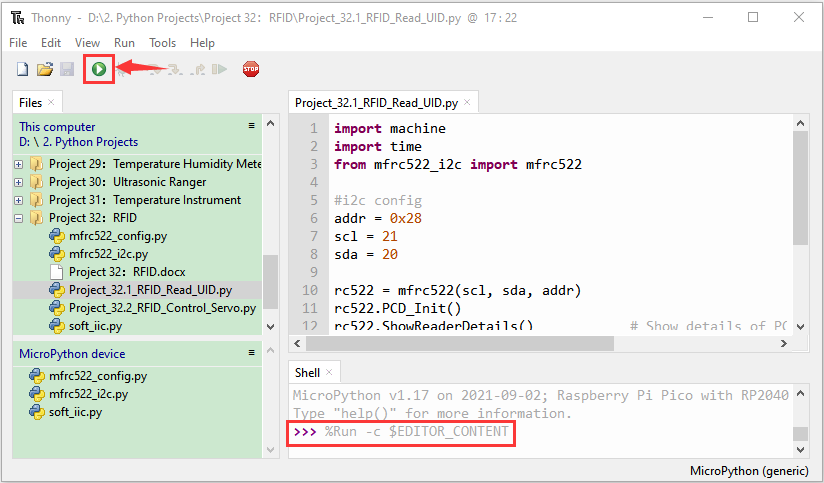
|  |
| --- |
| import machine  import time  from mfrc522\_i2c import mfrc522  #i2c config  addr = 0x28  scl = 21  sda = 20    rc522 = mfrc522(scl, sda, addr)  rc522.PCD\_Init()  rc522.ShowReaderDetails() # Show details of PCD - MFRC522 Card Reader details  while True:  if rc522.PICC\_IsNewCardPresent():  #print("Is new card present!")  if rc522.PICC\_ReadCardSerial() == True:  print("Card UID:")  print(rc522.uid.uidByte[0 : rc522.uid.size])  #time.sleep(1) |

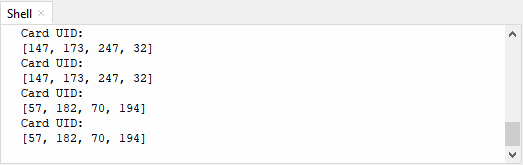
Ensure that the Raspberry Pi Pico is connected to the computer，click“Stop/Restart backend”.



Click “Run current script”, the code starts executing, we will see that

place the door card and key chain close to the module sensor area respectively, the "Shell" window of Thonny IDE will display the card number and key chain value respectively, as shown below. Press“Ctrl+C”or click“Stop/Restart backend”to exit the program.

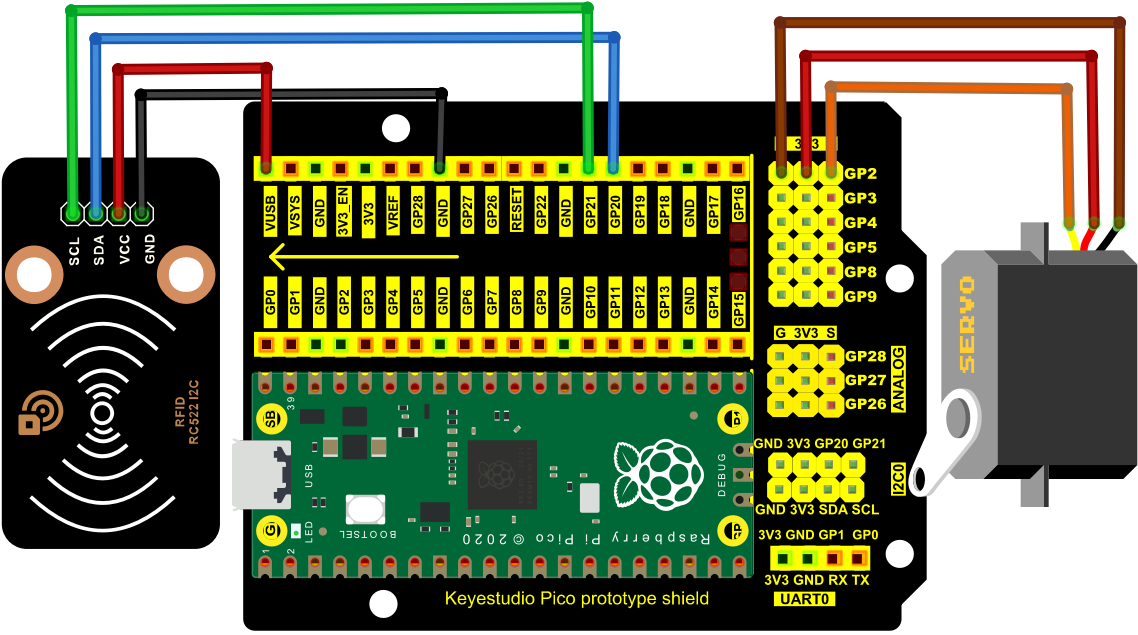




Note: the door card value and key chain value may be different for different RRFID -RC522 door cards and key chains.

1. **Circuit Diagram and Wiring diagram of RFID-RC522 Controlling Steering gear Rotation**

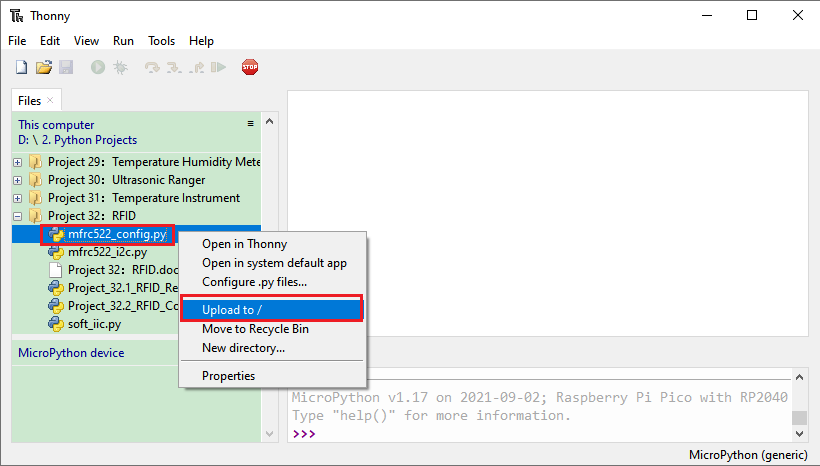
Now we use a RFID-RC522 module, door card/key chain and servo to simulate an intelligent access control system. When the door card is close to the RFID-RC522 module induction area, the servo rotates. Wiring according to the figure below:

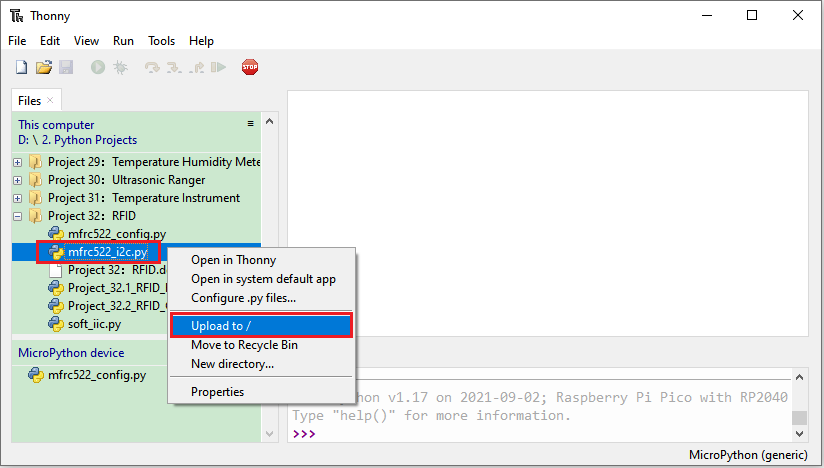


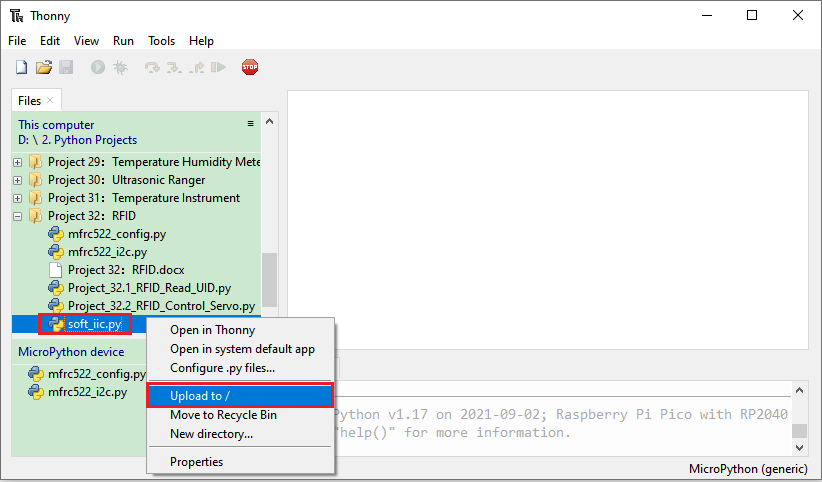
1. **Test Code**

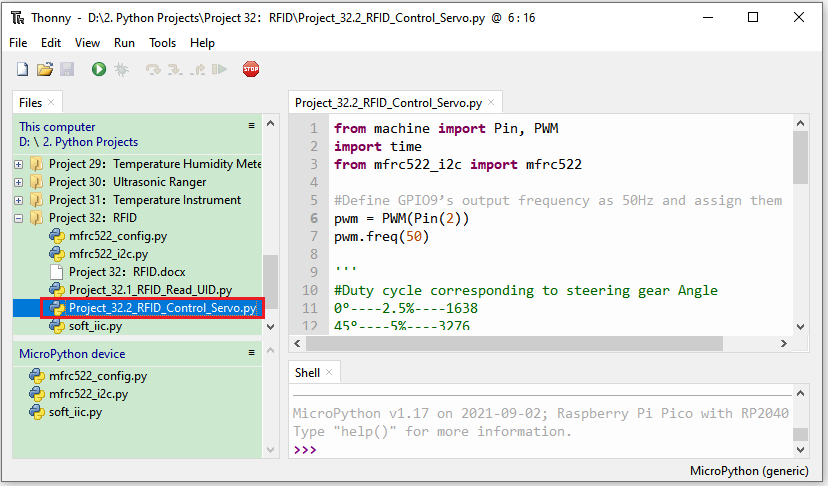
The code used in this project is saved in the file KS3020 Keyestudio Raspberry Pi Pico Learning Kit Ultimate Edition\2. Python\_Tutorial\Window System\2. Projects\Project 32：RFID. You can move the code to anywhere, for example, we can save the code in the Disk(D), the route is D:\2. Python Projects.

Open Thonny, click“This computer”→“D:”→“2. Python Projects”→“Project 32：RFID”. Select“mfrc522\_config.py”“mfrc522\_i2c.py”and“soft\_iic.py”，right-click and select“Upload to /”,waiting for the“mfrc522\_config.py”“mfrc522\_i2c.py”and“soft\_iic.py”to be uploaded to the Raspberry Pi Pico. And double left-click the“Project\_32.2\_RFID\_Control\_Servo.py”.







****

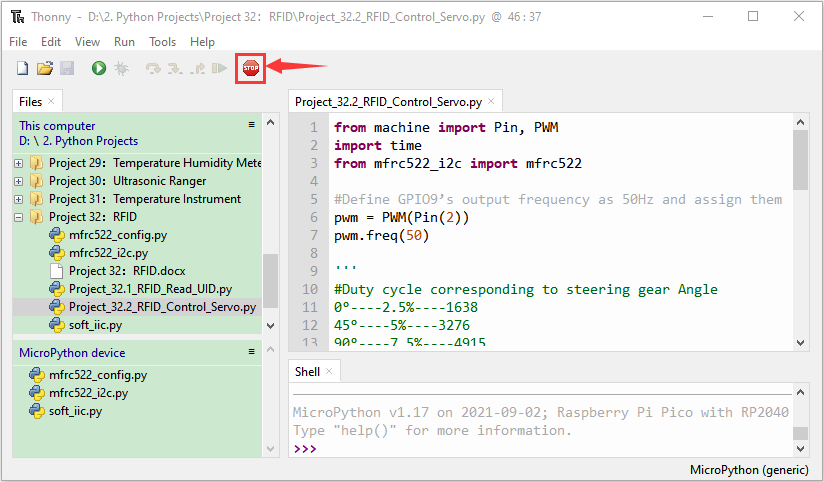
|  |
| --- |
| from machine import Pin, PWM  import time  from mfrc522\_i2c import mfrc522  #Define GPIO2’s output frequency as 50Hz and assign them to PWM.  pwm = PWM(Pin(2))  pwm.freq(50)  '''  #Duty cycle corresponding to steering gear Angle  0°----2.5%----1638  45°----5%----3276  90°----7.5%----4915  135°----10%----6553  180°----12.5%----8192  '''  #steering gear Angle are fit to its duty cycle.  angle\_0 = 1638  angle\_45 = 3276  angle\_90 = 4915  angle\_135 = 6553  angle\_180 = 8192  #i2c config  addr = 0x28  scl = 21  sda = 20    rc522 = mfrc522(scl, sda, addr)  rc522.PCD\_Init()  rc522.ShowReaderDetails() # Show details of PCD - MFRC522 Card Reader details  uid1 = [147, 173, 247, 32]  uid2 = [57, 182, 70, 194]  pwm.duty\_u16(angle\_180)  time.sleep(1)  while True:  if rc522.PICC\_IsNewCardPresent():  #print("Is new card present!")  if rc522.PICC\_ReadCardSerial() == True:  print("Card UID:", end=' ')  print(rc522.uid.uidByte[0 : rc522.uid.size])  if rc522.uid.uidByte[0 : rc522.uid.size] == uid1 or rc522.uid.uidByte[0 : rc522.uid.size] == uid2:  pwm.duty\_u16(angle\_0)  else :  pwm.duty\_u16(angle\_180)  time.sleep(500) |

Note: For different RFID-RC522 modules, white cards and key chains, its RFID-RC522 modules may read different UID1 and UID2 values. You should replace the uID1 and UID2 values of the white cards and key chains read by your RRFID -RC522 module with the corresponding values in the program code, otherwise, click "Run current script" to run the code may cause your swipe cards and key chains can not control the servo.

For example, you replace the  UID1 and UID2 values in the program code with the white card and key chain values read by your rFID-RC522 module.

1. **Test Result**

Ensure that the Raspberry Pi Pico is connected to the computer，click“Stop/Restart backend”.



Click “Run current script”, the code starts executing, we will see that when using the white card or a key card swiping, the "Shell" window of Thonny IDE will display the card number value respectively, and at the same time, the servo will rotate to the corresponding angle to simulate opening the door. Press“Ctrl+C”or click“Stop/Restart backend”to exit the program.

