Lesson 3 GPIO Pin Output Setting

1. GPIO Pin Introduction

GPIO (General Purpose Input/Output) port is a set of pins on the master of electronic devices used to send and receive electronic signals. You can connect these pins to external hardware devices to achieve functions for external communication, external hardware control, or external hardware data collection.

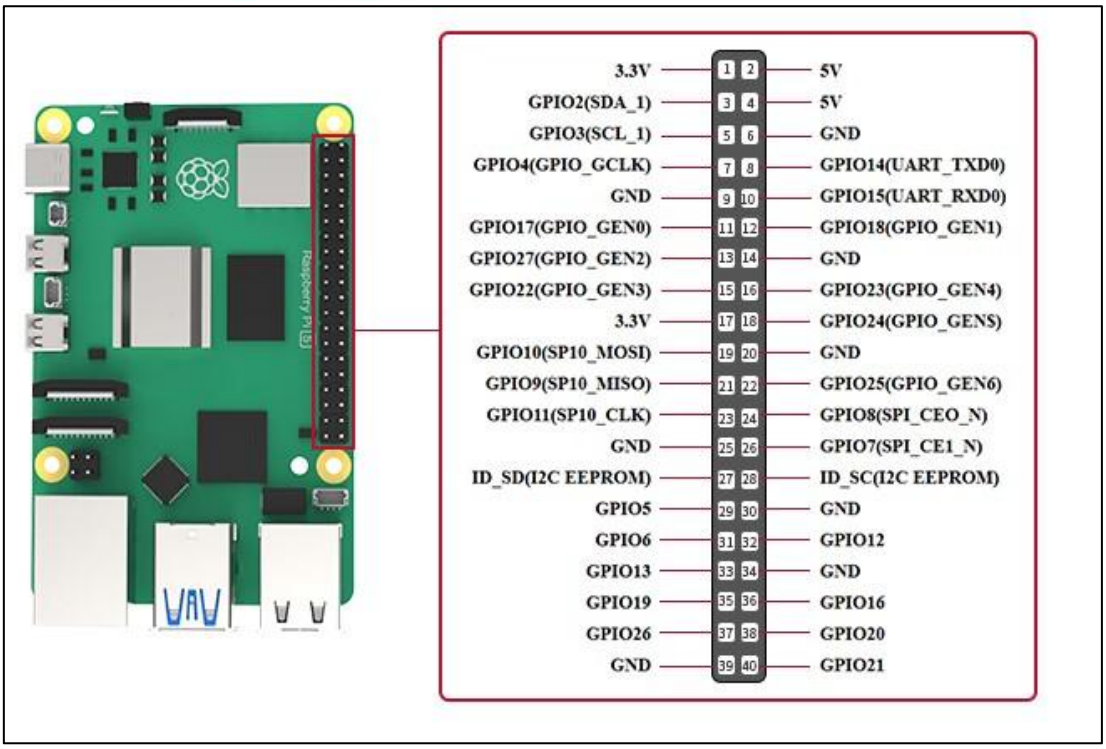
2. Input Introduction

GPIO output is a required function for controlling the high or low voltage level of the GPIO pins. The high and low voltage levels can be represented respectively by “1” and “0”. Take controlling an LED as an example. You need to set the pin connected to the LED to a high voltage level, which creates a voltage difference with the low voltage level side to generate a current and make the LED light up.

3. Raspberry Pi 5 GPIO Pin Introduction

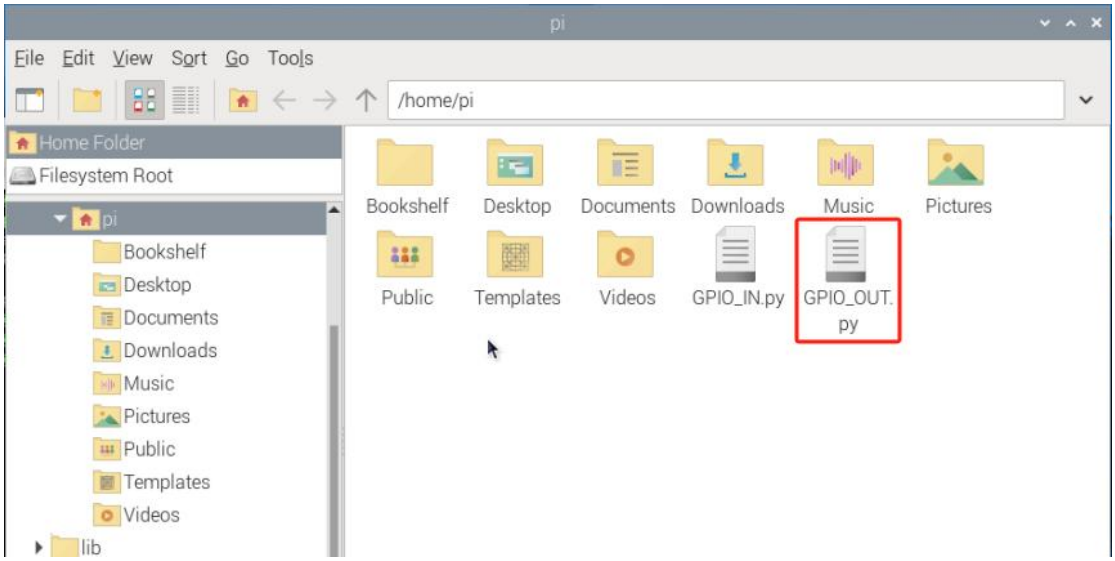
Raspberry Pi 5 features a 40-pin header that allows for easy use with a variety of expansion boards. GPIOD library can control these GPIO pins to read, write, interrupt, PWM, etc.

The distribution diagram of the GPIO pins is as follows:



4. Output Setting

1) Import the program file “GPIO_OUT.py” into the home directory of the Raspberry Pi 5 system, as the diagram shown below:



2) Press “Ctrl+Alt+T” to open the command line terminal and enter the

"sudo python3 GPIO_OUT.py" command, then press "Enter" to execute the program.

3) After running the program, use a multimeter to measure if the pin17 is switching constantly between the high and low voltage levels. For example, if you connect a 3.3v voltage to the pin, you can view the voltage changing between 3.3v and 0v.

5. Program Analysis

1) Import the necessary modules.

2) Initialize the GPIO controller and set the required GPIO port. Use the "line.get_value()" method to set the output status of GPIO pin17.

```
chip = gpiod.Chip('gpiochip0')
line = chip.get_line(17)
line.request(consumer='gpio_output', type=gpiod.LINE_REQ_DIR_OUT,default_vals=[0])
```

3) Use the “line.set_value() ” method to set the status of the GPIO port. Set it to a high voltage level, delay for one second, restore it to a low voltage level, then release the GPIO line and close the GPIO controller.

```
try:
    line.set_value(1)
    time.sleep(1)
    line.set_value(1)

finally:
    line.release()
    chip.close()
```