Program Analysis

1. File Path

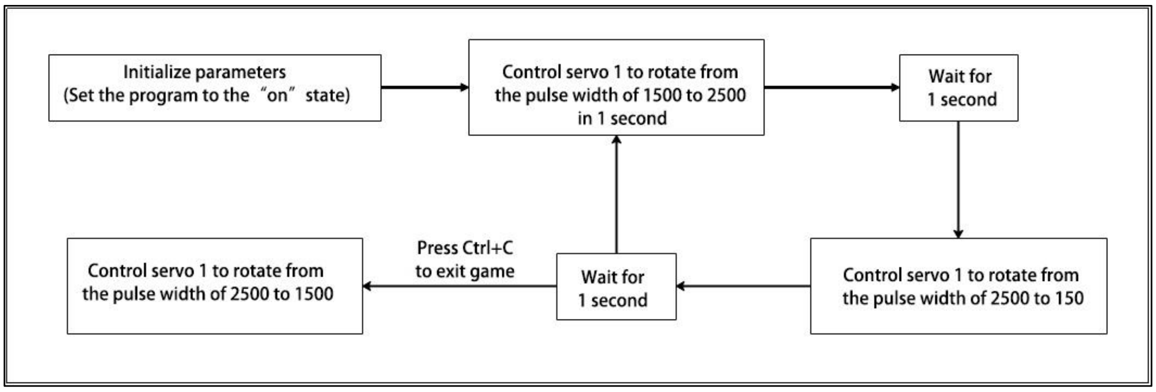
The program corresponding to this lesson is stored in: **/home/pi/MasterPi/HiwonderSDK/BuzzerControlDemo.py**

2. Performance

After the program is started, the servo rotates continuously between the 90° and 180°.

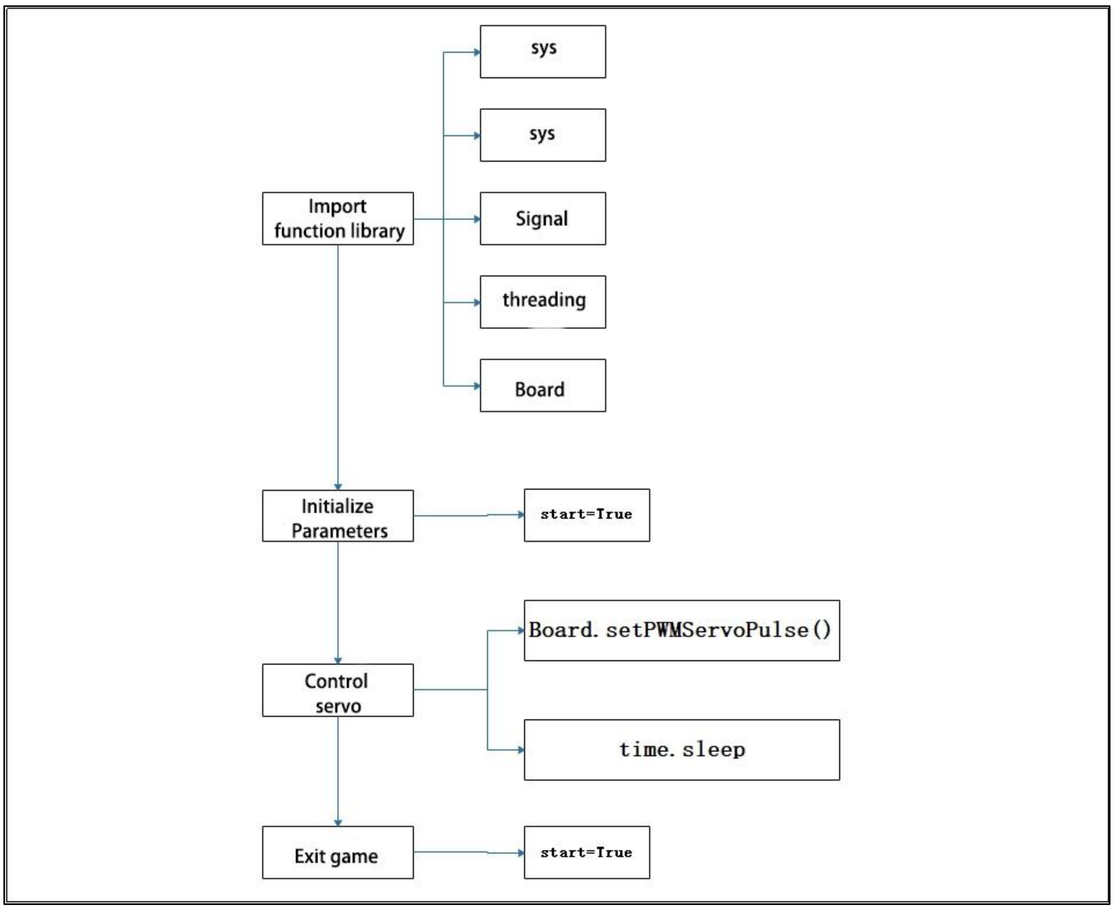
3. Program Analysis

3.1 Program Logic


Initialize parameters (Set the program to the "on" state)
Control servo 1 to rotate from the pulse width of 1500 to 2500 in 1 second
Wait for 1 second
Control servo 1 to rotate from the pulse width of 2500 to 150
Wait for 1 second
Press Ctrl+C to exit game
Control servo 1 to rotate from the pulse width of 2500 to 1500

As shown in the above diagram, the program will first performs the initialization operation, and modifies the parameters within the program. Then, it controls servo 1 to rotate from a pulse width of 1500 to 2500, corresponding to 90 ° to 180°. After 1 second, it executes the rotation from a pulse width of 2500 to 1500, corresponding to the 180° to 90° position, and repeats this process.

3.2 Program Analysis



From the above flow diagram, the program primarily divides into four parts, namely importing the function library, initializing parameters, controlling servo and closing the game. The following content will provide an explanation based on the logic of the program as depicted in the diagram.

- **Import Function Library**

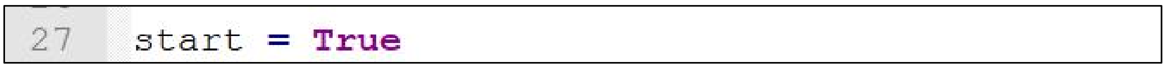
Firstly, import sys, time, signal, threading and Board functions. The below table will explain the library files.

```
import sys
sys.path.append('/home/pi/MasterPi/')
import time
import signal
import threading
import HiwonderSDK.Board as Board
```

Library File	Function
sys	Provide a multitude of functions and variables
Time	The expression for handling time within a program refers to the method of obtaining the system time and formatting its output
signal	Used for receiving and processing the signal.
threading	Providing an environment for running multiple threads
Board	control sensors for executing control operations.

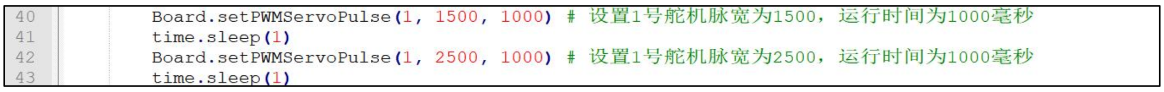
- **Initialization**

When executing a program, it is generally necessary to perform initialization within the program to facilitate the realization of the sequence functionalities. In this case, initialization involves setting the “Start” parameter in the program to True, indicating that the program is running.



```
start = True
```

- **Control Servo**



```
        Board.setPWMServoPulse(1, 1500, 1000) # 设置1号舵机脉宽为1500，运行时间为1000毫秒
        time.sleep(1)
        Board.setPWMServoPulse(1, 2500, 1000) # 设置1号舵机脉宽为2500，运行时间为1000毫秒
        time.sleep(1)
```

After the initialization is completed, the next step is to control the rotation of the servo. In this case, the functions Board.setPWMServoPulse() and time.sleep() are used.

Board.setPWMServoPulse() function: The function takes three parameters inside the parentheses. For example, Board.setPWMServoPulse(1, 1500, 1000), where parameter 1 represents servo 1, parameter 1500 represents the desired pulse width position, and 1000 represents the duration of the movement.

time.sleep() function: This function is used to control the duration of actions, such as the duration of the buzzer sound or the delay between servo motor movements.