Chatper8: PCA9695 control servo

We need to enter **pi/Adafruit_Python_PCA9685/examples**. Copy simpletest.py to the previous directory and run this program. We can learn how to operate the servo via Adafruit_Python_PCA9685 by reading the code of simpletest.py.

1. from __future__ import division: allows the current program to be compatible with future versions.
2. If you need to manually change the address of i2c for the current program, we can use:
Pwm=Adafruit_PCA9685.PCA9685(address=0x41,busnum=2)
0x41 is the i2c address of the expansion board we found.
3. servo_min and servo_max: set the maximum and minimum pulses of the current servo,
4. pwm.set_pwm(1,0,args) : this command to control servo

The first parameter specifies the number of the servo. We have previously inserted the test servo on the S1 port of the drive board, so the parameter is 1. The third parameter is the actual rotate angle of the servo. This parameter is defined as any pulse in the interval [servo_min, servo_max].

!!!Note: the third parameter here can only be entered as an integer.

If you want to check whether the servo of the kit device is normal by the following program, you can insert the servo that controls the camera to rotate left and right into S1 with the parameter 1, and the servo that controls the camera to rotate up and down to insert S2 with the parameter 2. The following program allows the two servos to rotate to the specified position for a short time and then return to the position.

5. pwm.set_pwm_freq(50) : Define the frequency of the reference pulse as 50hz, which is a period of 20ms. In fact between pulse width can be 0.5 ms and 2.5 ms. Pulse width and the servo rotation angle 0° ~ 180° corresponds, as shown below.

0.5ms----------0°
1.0ms----------45°
1.5ms----------90°
2.0ms----------135°
2.5ms----------180°

And the maximum frequency of the servo is 4096, so we have a ratio:

((0°*11)+500)/20000=pulse1/4096
((1°*11)+500)/20000=pulse2/4096
...
((180°*11)+500)/20000=pulseN/4096

11 is a value that converts the angle to 12-bit precision. We convert 20ms into 20,000 us, and the pulse calculated in microseconds is the third parameter required by pwm.set_pwm(channel,0,pulse).

```
# Simple demo of of the PCA9685 PWM servo/LED controller library.
# This will move channel 0 from min to max position repeatedly.
# Author: Tony DiCola
# License: Public Domain
from __future__ import division
import time

# Import the PCA9685 module.
import Adafruit_PCA9685


# Uncomment to enable debug output.
#import logging
#logging.basicConfig(level=logging.DEBUG)

# Initialise the PCA9685 using the default address (0x40).
pwm = Adafruit_PCA9685.PCA9685()

# Alternatively specify a different address and/or bus:
#pwm = Adafruit_PCA9685.PCA9685(address=0x41, busnum=2)

# Configure min and max servo pulse lengths
servo_min = 150  # Min pulse length out of 4096
servo_max = 600  # Max pulse length out of 4096

# Helper function to make setting a servo pulse width simpler.
def set_servo_pulse(channel, pulse):
    pulse_length = 1000000    # 1,000,000 us per second
    pulse_length //= 60       # 60 Hz
    print('{0}us per period'.format(pulse_length))
    pulse_length //= 4096     # 12 bits of resolution
    print('{0}us per bit'.format(pulse_length))
    pulse *= 1000
    pulse //= pulse_length
    pwm.set_pwm(channel, 0, pulse)

# Set frequency to 60hz, good for servos.
pwm.set_pwm_freq(60)

print('Moving servo on channel 0, press Ctrl-C to quit...')
while True:
    # Move servo on channel O between extremes.
    pwm.set_pwm(0, 0, servo_min)
    time.sleep(1)
    pwm.set_pwm(0, 0, servo_max)
    time.sleep(1)

```