2.Servo calibration

1. API Introduction

1.Turn on or turn off the torque of the bus servo： **Arm_serial_set_torque(enable)**

Parameter explanation:

enable: enable=0 turns off the torque, and you can manually turn the angle of the servo without receiving level signals;

enable=1 turns on the torque, and the servo receives level signals to maintain the current angle. The angle can only be changed by sending a command. .

Return value: None.

2. **Arm_serial_servo_write_offset_switch(id)**

Parameter explanation:

id: When id=0, clear the center adjustment of all servos and restore to default;

id=1~6, corresponding to the ID numbers of six servos. After receiving this command, the underlying microcontroller will read the angle data of the servos corresponding to the ID. If it is within a reasonable range, it will be saved. If it exceeds the range or If the servo ID cannot be found, it will not be saved.

Return value: None.

3. **Arm_serial_servo_write_offset_state()**

Parameter explanation:

Return value: Returns the state of setting the median deviation,

state=0 means that the servo is not detected;

state=1 indicates that the median setting is successful;

state=2 means that the set median value exceeds the range.

2.About code

Code path： /home/yahboom/Dofbot/2.sys_settings/2.Offset/offset.ipynb

This code cannot be executed in sequence all at once. It must be executed in steps to ensure that the position of the servo can be adjusted correctly.
The result of the adjustment is to keep the robotic arm upright.

Note: This program can only be used when the center position needs to be adjusted, and cannot be used arbitrarily, otherwise it will cause the center position to be inaccurate and affect the grabbing effect.

```
#!/usr/bin/env   python3
#coding=utf-8
import   time
from   Arm_Lib import Arm_Device

#Create a robotic arm object
Arm   = Arm_Device()
time.sleep(.1)
```

```
# Let the servo return to the neutral and upright position
Arm.Arm_serial_servo_write6(90,   90, 90, 90, 90, 180, 1000)
time.sleep(2)
```

```
#Turn off the torque. At this time, you can adjust the angle of the servo by hand.
Arm.Arm_serial_set_torque(0)
```

```
#   After adjusting the angle of a certain servo, you can set the center deviation of a certain servo independently.
id   = 6
Arm.Arm_serial_servo_write_offset_switch(id)
time.sleep(.1)
state   = Arm.Arm_serial_servo_write_offset_state()
if   state == 1:
    print("set offset ok!")
elif   state == 2:
    print("error! set offset overrun   !")
elif   state == 0:
    print("error! set offset error   !")
```

```
#  You can also set the median deviation of all servos (No. 1-6) at one time
for   i in range(6):
    id = i + 1
      Arm.Arm_serial_servo_write_offset_switch(id)
    time.sleep(.1)
    state =   Arm.Arm_serial_servo_write_offset_state()
    if state == 1:
        print("id:%d set offset   ok!" % id)
    elif state == 2:
        print("error!id:%d set offset   overrun !" % id)
    elif state == 0:
        print("error!id:%d set offset   error !" % id)
```

```
#   After the adjustment is complete, turn on the torque
Arm.Arm_serial_set_torque(1)
```

```
# Clear the median deviation of all servo settings and restore to the default state.
# If you need to clear the median deviation of all servos, please delete the # below and run this unit again
#Arm.Arm_serial_servo_write_offset_switch(0)
```

```
del   Arm  # Release the Arm object
```