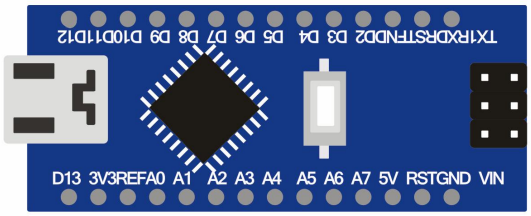
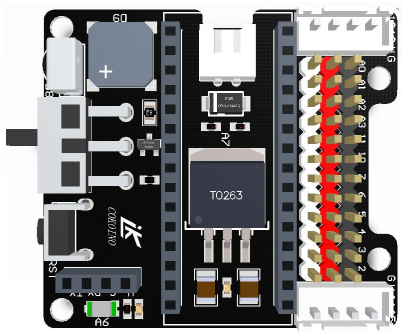
180 Degree Servo

Need to prepare:

- A Nano board
- A Nano shield
- A servo
- A battery case with two 18650 batteries
- A usb cable





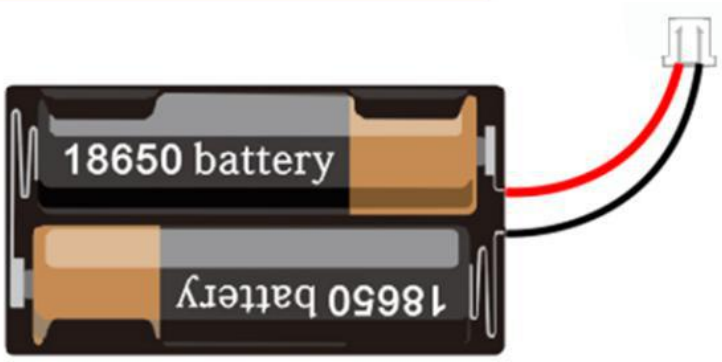
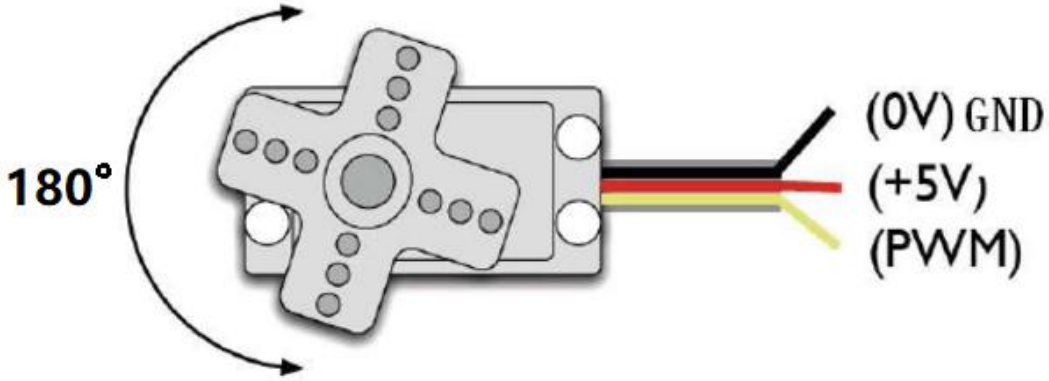


Table of Contents

1、Overview：

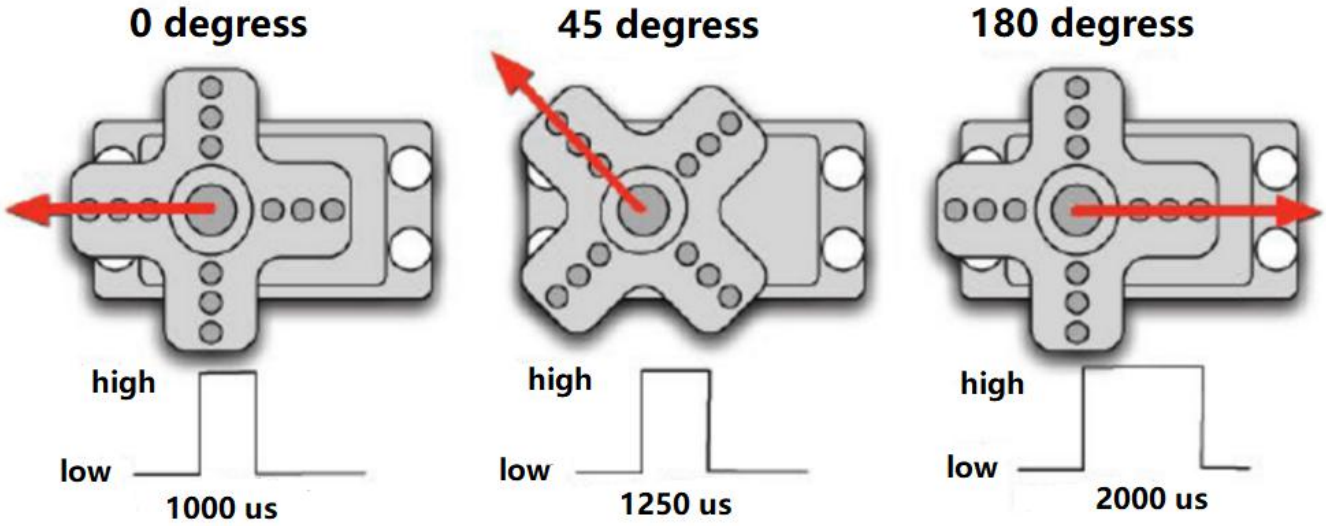
A Servo is a small device that incorporates a two wire DC motor, a gear train, a potentiometer, an integrated circuit, and an output shaft. Of the three wires that stick out from the motor casing, one is for power, one is for ground, and one is a control input line. The shaft of the servo can be positioned to specific angular positions by sending a coded signal. As long as the coded signal exists on the input line, the servo will maintain the angular position of the shaft. If the coded signal changes, then the angular position of the shaft changes.

Servos are constructed from three basic pieces; a motor, a potentiometer (variable resistor) that is connected to the output shaft, and a control board. The potentiometer allows the control circuitry to monitor the current angle of the servo motor. The motor, through a series of gears, turns the output shaft and the potentiometer simultaneously. The potentiometer is fed into the servo control circuit and when the control circuit detects that the position is correct, it stops the motor. If the control circuit detects that the angle is not correct, it will turn the motor the correct direction until the angle is correct. Normally a servo is used to control an angular motion of between 0 and 180 degrees.



The angle of rotation of the steering gear is achieved by adjusting the duty cycle of the PWM (Pulse Width Modulation) signal. The period of the standard PWM (Pulse Width Modulation) signal is fixed at 20ms (50Hz). Theoretically, the pulse width distribution should be 1ms to Between 2ms.

However, in fact, the pulse width can be between 0.5ms and 2.5ms, and the pulse width corresponds to the rotation angle of the steering gear from 0° to 180°.

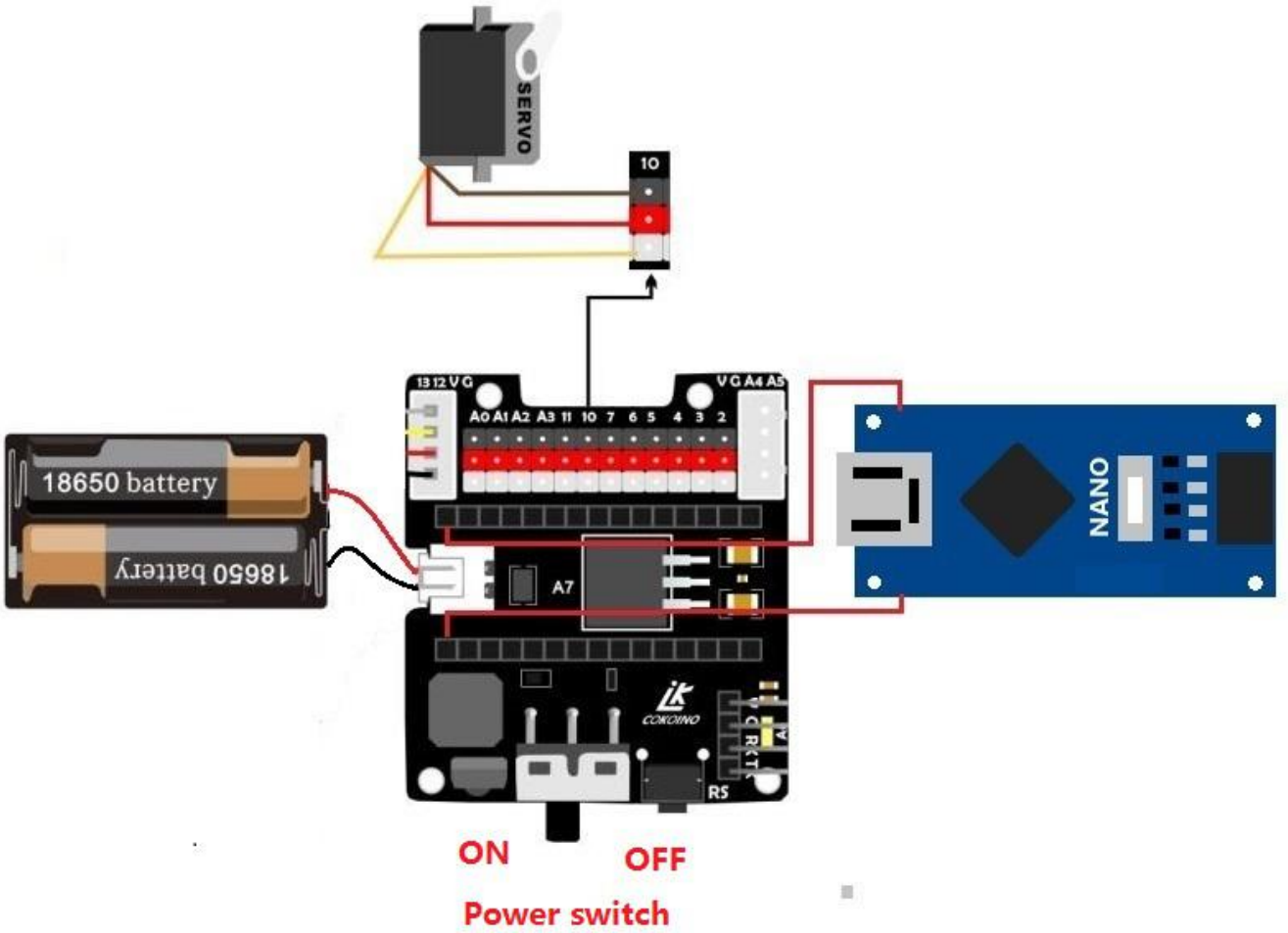


2、Two ways to control the servo

For the Arduino, there are two ways to control the servos.
One is that the Arduino's common digital port generates square waves with different duty cycles to simulate PWM signals for servo positioning.
The second method is to directly control the steering gear by using the Arvoino's own Servo function. The advantage of this control method is that is easy to program. The Arduino has limited drive capability, so an external power supply is required when it is necessary to control more than one servo.

Explain the common functions of the Servo.h library file:
1, attach (IO port) - set the interface of the servo.
2, write (angle) - set the steering angle of the servo, the range of angles that can be set is 0 ° to 180 °.
3, read () - read the steering angle, can be understood as reading the value of the last write () command.
4. attached() - Determines whether the servo parameters have been sent to the interface where the servo is located.
5, detach () - the servo is separated from the interface.

2.1、Connection method



2.2、Method one:

Code:

```
int servopin=10;//Define digital interface 10 to connect the servo signal line
int myangle;//Define the angle variable to be 0-180
int pulsewidth;//Define pulse width variable
int val; //0-9

void servopulse(int servopin,int myangle)//Define a pulse function
{
pulsewidth=(myangle*11)+500;//Convert the angle to a pulse width of 500-2480

digitalWrite(servopin,HIGH);//Set the servo interface level to high

delayMicroseconds(pulsewidth);//The number of microseconds of the delay pulse width value

digitalWrite(servopin,LOW);//Set the servo interface level to low

delayMicroseconds(2500-pulsewidth);
}

void setup()
{
pinMode(servopin,OUTPUT);//Set the servo interface to the output interface

Serial.begin(9600);//Connected to the serial port with a baud rate of 9600

Serial.println("servo=o_seral_simple ready" ) ;
}

void loop()//Convert a number from 0 to 9 to a 0 to 180 angle and let the LED flash the corresponding number of times
{
val=Serial.read();//Read the value of the serial port

if(val>'0'&&val<='9')
{
val=val-'0';//Convert feature quantities to numeric variables
val=val*(180/9);//Convert numbers to angles
Serial.print("moving servo to ");
//DEC:Outputs the ASCII encoded value of b in decimal form, followed by a carriage return and line feed symbol
Serial.print(val,DEC);
Serial.println();
for(int i=0;i<=50;i++) //Give the servos enough time to turn it to the specified angle
{
servopulse(servopin,val);//Reference pulse function
}
}
}
```

2.21、Upload the code：

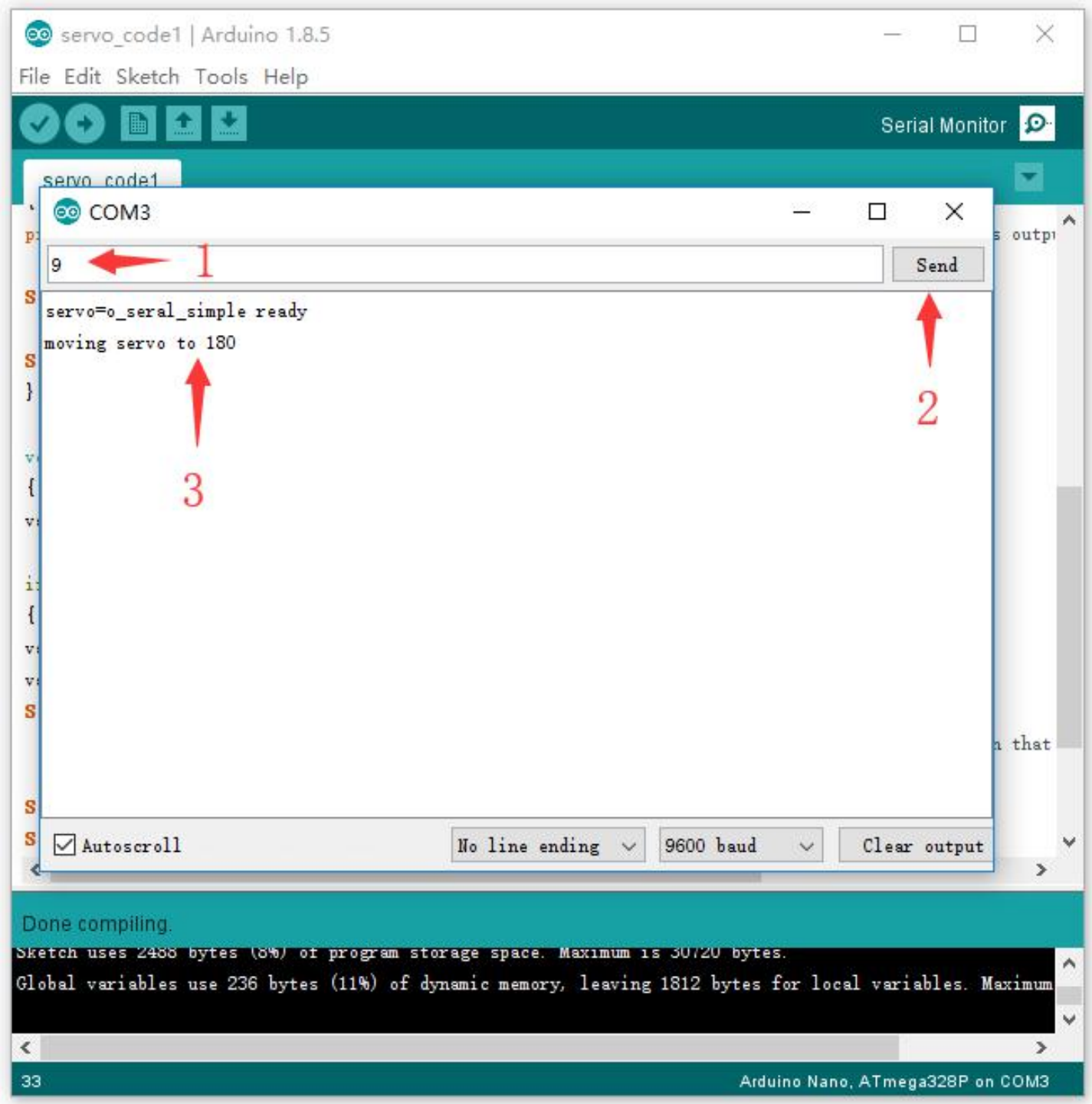
Copy the above code to the Arduino IDE, connect the PC to the NANO motherboard with a USB cable, select the corresponding board type and port in the IDE, upload the code to the NANO motherboard.

2.22、Open the serial monitor and set the baud rate

Set the serial port baud rate to 9600, send number 0, 1, 2, 3, 4, 5, 6, 7, 8 or 9 to the nano board, Note that the number 0-9 corresponds to the steering angle 0-180 of the servo.

2.23、Control the servo

For example, if you enter 9, the rudder turns 180 degrees, if you enter 3, the rudder turns 60 degrees.



2.3、Method Two:

Code：

```
#include<Servo.h>
Servo myservo;    // create servo object to control a servo
                        // a maximum of eight servo objects can be created
int pos = 0;          // variable to store the servo position
void setup()
{
myservo.attach(9);    // attaches the servo on pin 9 to the servo object
}
void loop()
{
for(pos=0;pos<180;pos+=1)    // goes from 0 degrees to 180 degrees
{
  myservo.write(pos); // tell servo to go to position in variable 'pos'
  delay(15);    // waits 15ms for the servo to reach the position
}
for(pos = 180;pos>=1;pos-=1)    // goes from 180 degrees to 0 degrees
  {
  myservo.write(pos); // tell servo to go to position in variable 'pos'
  delay(15);    //waits 15ms for the servo to reach the position
  }
}
```

2.31、Upload the code：

Copy the above code to the Arduino IDE, copy the servo library file to the IED library folder, connect the PC to the NANO motherboard with a USB cable, select the corresponding board type and port in the IDE, upload Code to the NANO motherboard,

2.32、Results：

Turn on the power switch on the expansion board, you can see that the servo starts running from 0 to 180 degrees, and then from 180 degrees to 0 degrees.