Task 26

Deep Das

Question : What is PWM signals and where they are used? How servo works on PWM signals ?

Answer :

PWM Signals (Pulse Width Modulation)

PWM stands for Pulse Width Modulation. It's a technique used to control the amount of power delivered to an electrical load by varying the width (duration) of the pulses in a pulse train. In other words, PWM signals are digital signals that switch between high and low states at a consistent frequency, but the time the signal stays high (the pulse width) varies.

Key Characteristics of PWM Signals:

- **Duty Cycle**: The duty cycle is the percentage of one period in which a signal is active (high). A 100% duty cycle means the signal is always on, while a 0% duty cycle means it's always off. A 50% duty cycle means the signal is on for half the period and off for the other half.
- **Frequency**: This is how often the signal switches between high and low states per second. The frequency remains constant in PWM; only the duty cycle changes.

Uses of PWM Signals:

- **Motor Control**: PWM signals are commonly used to control the speed of DC motors and the position of servos. By adjusting the duty cycle, the effective voltage applied to the motor changes, thus controlling its speed.
- **LED Dimming**: PWM is used to dim LEDs by varying the duty cycle. This makes the LED appear dimmer or brighter to the human eye.
- **Audio Signal Generation**: PWM can be used in digital-to-analog conversion, where the PWM signal is filtered to produce analog audio signals.
- **Power Delivery**: PWM is used in power supplies and power converters to regulate voltage and current.

Why Servos Work Only on PWM Signals

Servos are designed to work specifically with PWM signals due to their precision and reliability in providing controlled movement. Here’s why PWM is ideal for servos:

1. **Precise Control**: PWM allows for precise control of the position of the servo. The width of the pulse determines the exact position the servo should move to, making it ideal for applications where accurate positioning is crucial.
2. **Consistent Timing**: PWM signals maintain a constant frequency, which ensures that the servo receives consistent and regular instructions. This consistency is critical for maintaining stable and predictable movements.

3. **Efficiency**: PWM is an efficient way to control the power delivered to the servo motor. By adjusting the duty cycle, PWM can efficiently regulate the amount of power without generating excessive heat or wasting energy.
4. **Compatibility**: Most servos are designed to interpret PWM signals because they are widely used in various control systems, such as microcontrollers and RC (radio-controlled) devices. This standardization makes it easy to interface servos with different controllers.

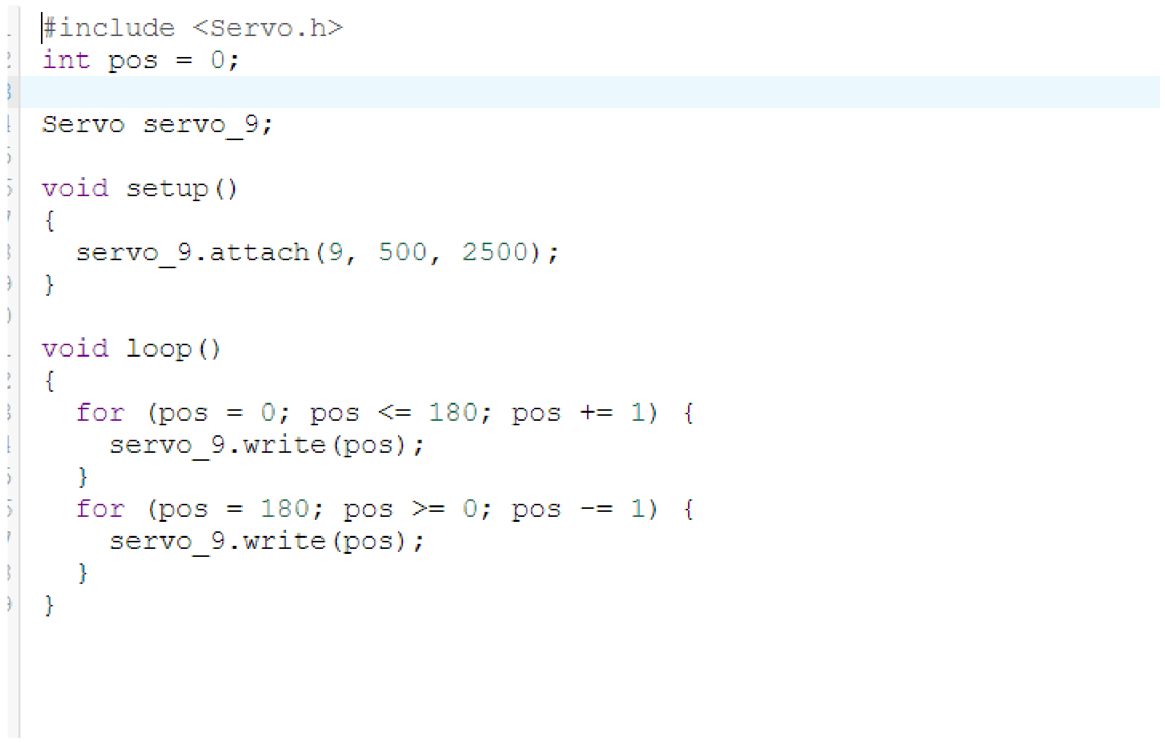
How Servos Work on PWM Signals

A servo motor is a type of motor that can rotate to a specific position based on the PWM signal it receives. Servos are widely used in applications like robotics, RC cars, and airplanes.

Servo Operation with PWM:

1. **PWM Control Signal**: Servos are controlled by sending them a PWM signal. The frequency of this signal is typically around 50 Hz (20 ms period).
2. **Pulse Width**: The width of the pulse (on-time) within each PWM cycle determines the position of the servo.
    - A typical servo expects a pulse width between 1 ms to 2 ms.
    - 1 ms pulse usually corresponds to the servo's 0-degree position.
    - 1.5 ms pulse corresponds to the middle (90-degree) position.
    - 2 ms pulse corresponds to the 180-degree position.
3. **Feedback Mechanism**: Inside the servo, there's a feedback mechanism that includes a potentiometer connected to the output shaft. The servo controller compares the desired position (based on the PWM signal) with the actual position (from the potentiometer) and adjusts the motor's rotation to match the desired position.

Example :



```
#include <Servo.h>
int pos = 0;

Servo servo_9;

void setup()
{
  servo_9.attach(9, 500, 2500);
}

void loop()
{
  for (pos = 0; pos <= 180; pos += 1) {
    servo_9.write(pos);
  }
  for (pos = 180; pos >= 0; pos -= 1) {
    servo_9.write(pos);
  }
}
```