## Лабораторная работа 7: Работа с ШИМ

**Цель:**

* Изучить принцип работы широтно-импульсной модуляции (ШИМ).
* Освоить работу с ШИМ на ESP32.
* Научиться управлять яркостью светодиода и положением сервопривода с помощью ШИМ.
* Интегрировать ШИМ в проект для реализации полезных функций.

## Теоретическая часть

### Широтно-импульсная модуляция (ШИМ)

ШИМ - это метод управления аналоговыми устройствами с помощью цифрового сигнала. Вместо того, чтобы напрямую подавать на устройство фиксированное напряжение, ШИМ использует серию импульсов с переменной шириной.

**Принцип работы:**

1. **Частота:** ШИМ-сигнал имеет фиксированную частоту, которая определяет количество импульсов в секунду.
2. **Скважность:** Скважность ШИМ-сигнала - это отношение длительности импульса к его периоду.
   * Чем больше скважность, тем больше энергии получает устройство.
   * Чем меньше скважность, тем меньше энергии получает устройство.

**Применение:**

* **Управление яркостью светодиодов:** Изменяя скважность ШИМ-сигнала, можно регулировать яркость свечения светодиода.
* **Управление сервоприводами:** Сервоприводы используют ШИМ-сигнал для определения своего положения.
* **Регулировка скорости вращения двигателей:** С помощью ШИМ можно регулировать скорость вращения двигателей постоянного тока.

### ШИМ на ESP32

ESP32 имеет два модуля ШИМ:

* **TPWM:** 12-битный модуль с более высокой точностью и частотой.
* **LEDC:** 8-битный модуль с меньшей точностью и частотой, но более простым использованием.

Для управления светодиодами и сервоприводами обычно используется модуль LEDC.

### Подключение светодиода

**Схема подключения:**

* Светодиод подключен к GPIO пину ESP32 (например, GPIO13).

**Код:**

const int ledPin = 13; // GPIO пин, к которому подключен светодиод  
  
void setup() {  
 pinMode(ledPin, OUTPUT); // Настройка GPIO пина как выходного  
 ledcSetup(0, ledPin, 8); // Настройка канала LEDC 0 для GPIO пина 13, 8-битная разрядность  
 ledcAttachPin(0, ledPin); // Подключение GPIO пина к каналу LEDC 0  
}  
  
void loop() {  
 // Изменение скважности ШИМ-сигнала от 0 до 255  
 for (int i = 0; i <= 255; i++) {  
 ledcWrite(0, i); // Установка скважности ШИМ-сигнала  
 delay(10); // Задержка между изменениями скважности  
 }  
  
 // Изменение скважности ШИМ-сигнала от 255 до 0  
 for (int i = 255; i >= 0; i--) {  
 ledcWrite(0, i);  
 delay(10);  
 }  
}

**Описание:**

* В коде определена константа ledPin, которая указывает на GPIO пин, к которому подключен светодиод.
* В функции setup() настраивается GPIO пин как выходной и инициализируется канал LEDC 0 для управления этим пином.
* В цикле loop() скважность ШИМ-сигнала изменяется от 0 до 255, что приводит к плавному изменению яркости светодиода.
* Затем скважность ШИМ-сигнала изменяется от 255 до 0, что приводит к обратному изменению яркости светодиода.

### Подключение сервопривода

**Схема подключения:**

* Сервопривод подключен к GPIO пину ESP32 (например, GPIO15).

**Код:**

const int servoPin = 15; // GPIO пин, к которому подключен сервопривод  
  
void setup() {  
 ledcSetup(0, servoPin, 8); // Настройка канала LEDC 0 для GPIO пина 15, 8-би  
 ledcAttachPin(0, servoPin); // Подключение GPIO пина к каналу LEDC 0  
 servoAttach(servoPin, 500, 2500); // Настройка сервопривода (зависит от модели)  
}  
  
void loop() {  
 // Управление положением сервопривода  
 for (int angle = 0; angle <= 180; angle += 1) {  
 ledcWrite(0, map(angle, 0, 180, 500, 2500)); // Установка скважности для нужного угла  
 delay(15); // Задержка между изменениями положения  
 }  
  
 for (int angle = 180; angle >= 0; angle -= 1) {  
 ledcWrite(0, map(angle, 0, 180, 500, 2500));  
 delay(15);  
 }  
}

**Описание:**

* В коде определена константа servoPin, которая указывает на GPIO пин, к которому подключен сервопривод.
* В функции setup() настраивается канал LEDC 0 для управления этим пином.
* servoAttach(servoPin, 500, 2500); - функция для настройки сервопривода (значения 500 и 2500 могут отличаться для разных моделей, необходимо сверяться с документацией).
* В цикле loop() с помощью функции map() преобразуется угол поворота сервопривода (0 - 180 градусов) в значение скважности ШИМ-сигнала (500 - 2500).
* Чем больше скважность, тем больше угол поворота сервопривода.
* Цикл обеспечивает постепенное движение сервопривода от 0 до 180 градусов и обратно.

**Вариации:**

* Можно управлять яркостью нескольких светодиодов, используя отдельные каналы LEDC.
* Можно управлять несколькими сервоприводами, используя отдельные каналы LEDC и функции servoAttach().
* Можно управлять яркостью светодиода или положением сервопривода в зависимости от внешних факторов (например, считывая данные с датчика).

## Задачи:

#### 1. Управление яркостью светодиода по датчику освещенности

**Описание:**

Создайте проект, в котором яркость светодиода автоматически регулируется в зависимости от освещенности. При низкой освещенности яркость светодиода будет увеличиваться, а при высокой - уменьшаться.

**Требования:**

* ESP32
* Датчик освещенности
* Светодиод
* Соединительные провода

**Решение:**

1. Подключите датчик освещенности и светодиод к ESP32.
2. Напишите программу, которая считывает данные с датчика освещенности и управляет яркостью светодиода с помощью ШИМ.
3. Определите функцию, которая преобразует значение освещенности в значение скважности ШИМ-сигнала для светодиода.

#### 2. Управление роботизированной рукой с помощью сервоприводов

**Описание:**

Создайте роботизированную руку, которая может управлять положением своих суставов с помощью сервоприводов.

**Требования:**

* ESP32
* Несколько сервоприводов
* Соединительные элементы для создания роботизированной руки

**Решение:**

1. Подключите сервоприводы к ESP32.
2. Напишите программу, которая управляет положением каждого сервопривода с помощью ШИМ-сигнала.
3. Разработайте алгоритмы для управления движениями роботизированной руки (например, считывая команды с пульта управления или используя заранее запрограммированные движения).

## Примеры кода для лабораторной работы:

### 1. Управление яркостью светодиода по датчику освещенности

**Схема подключения:**

* Датчик освещенности подключен к аналоговому входу ESP32 (например, A0).
* Светодиод подключен к GPIO пину ESP32 (например, GPIO13).

**Код:**

const int analogPin = A0; // Аналоговый вход, к которому подключен датчик освещенности  
const int ledPin = 13; // GPIO пин, к которому подключен светодиод  
  
void setup() {  
 Serial.begin(115200);  
 pinMode(ledPin, OUTPUT); // Настройка GPIO пина как выходного  
 ledcSetup(0, ledPin, 8); // Настройка канала LEDC 0 для GPIO пина 13, 8-битная разрядность  
 ledcAttachPin(0, ledPin); // Подключение GPIO пина к каналу LEDC 0  
}  
  
void loop() {  
 // Считывание значения с датчика освещенности  
 int sensorValue = analogRead(analogPin);  
  
 // Преобразование аналогового значения в значение освещенности  
 float lightLevel = sensorValue \* (100.0 / 4096.0);  
  
 // Определение скважности ШИМ-сигнала  
 int pwmValue = map(lightLevel, 0, 100, 0, 255);  
  
 // Установка скважности ШИМ-сигнала  
 ledcWrite(0, pwmValue);  
  
 // Вывод информации на Serial  
 Serial.print("Значение датчика: ");  
 Serial.println(sensorValue);  
 Serial.print("Освещенность: ");  
 Serial.println(lightLevel, 2);  
 Serial.print("Скважность: ");  
 Serial.println(pwmValue);  
  
 delay(100); // Задержка между измерениями  
}

**Описание:**

* В коде определены константы analogPin и ledPin, которые указывают на аналоговый вход и GPIO пин, к которым подключены датчик освещенности и светодиод соответственно.
* В функции setup() инициализируется UART, настраивается GPIO пин светодиода и канал LEDC 0.
* В цикле loop() считывается значение с датчика освещенности, преобразуется в процентное содержание освещенности и выводится на Serial.
* С помощью функции map() значение освещенности преобразуется в значение скважности ШИМ-сигнала (0 - 255).
* Установленная скважность ШИМ-сигнала будет регулировать яркость светодиода.

### 2. Управление роботизированной рукой с помощью сервоприводов

**Описание:**

В качестве примера рассмотрим управление простейшей роботизированной рукой с одним сервоприводом, который может двигаться вверх и вниз.

**Схема подключения:**

* Сервопривод подключен к GPIO пину ESP32 (например, GPIO15).

**Код:**

const int servoPin = 15; // GPIO пин, к которому подключен сервопривод  
  
void setup() {  
 Serial.begin(115200);  
 ledcSetup(0, servoPin, 8); // Настройка канала LEDC 0 для GPIO пина 15, 8-битная разрядность  
 ledcAttachPin(0, servoPin); // Подключение GPIO пина к каналу LEDC 0  
 servoAttach(servoPin, 500, 2500); // Настройка сервопривода (зависит от модели)  
}  
  
void loop() {  
 // Управление положением сервопривода  
 for (int angle = 0; angle <= 180; angle += 5) {  
 ledcWrite(0, map(angle, 0, 180, 500, 2500)); // Установка скважности для нужного угла  
 delay(25); // Задержка между изменениями положения  
 }  
  
 for (int angle = 180; angle >= 0; angle -= 5) {  
 ledcWrite(0, map(angle, 0, 180, 500, 2500));  
 delay(25);  
 }  
}

**Схема подключения**

## Подключение сервопривода :

+-------------------------------------------------+  
| |  
| ESP32 |  
| |  
| 3.3V <--> Сервопривод (VCC) |  
| GPIO 15 <--> Сервопривод (сигнальный провод) |  
| GND <--> Сервопривод (GND) |  
| |  
+-------------------------------------------------+