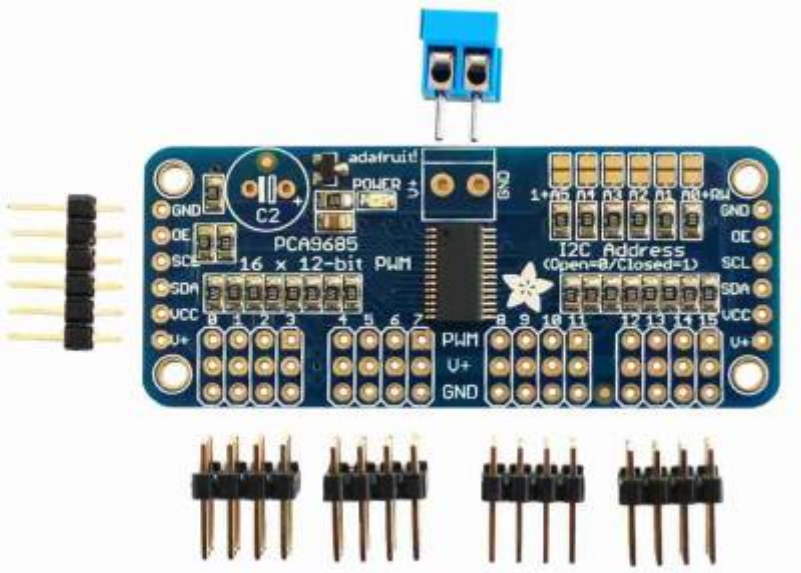
PCA9685

PCA9685 ADAFRUIT



Permet de contrôler jusqu'à 16 servo-moteurs (ou LEDs) en PWM.

Peut être monté en série pour obtenir 32,64,128,... canaux et ce jusqu'à 62 x 16 = 992 canaux avec seulement 2 fils d'interface en I2C

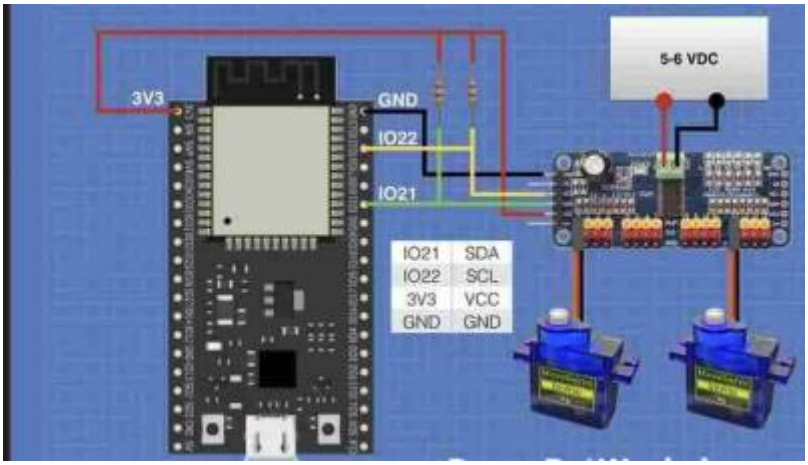
PCA9685 -- ADAFRUIT

Datasheet PCA9685

Adafruit PCA9685 PWM Servo Driver Library

Control 32 Servo motor using PCA9685 Module and ESP32 V4 EN

exemple avec un esp32



Les connexions I2C à l'ESP32 sont les suivantes :

- SDA-IO21
- SCL-IO22

En fait, vous pouvez utiliser deux broches quelconques sur l'ESP32 pour vos connexions I2C et les définir dans le logiciel.

Les broches GPIO 21 et 22 sont les broches par défaut si de nouvelles broches ne sont pas déclarées.

Vous remarquerez l'utilisation de résistances pullup pour les données I2C et les lignes d'horloge.

Comme nous utilisons une logique de 3,3 volts, vous devez utiliser des valeurs comprises entre 2,4k et 3,3k.

Connexion des servomoteurs aux connecteurs 0 et 12, mais vous pouvez en choisir deux autres et ajouter d'autres servomoteurs.

Lorsque vous regardez le code, vous verrez comment définir le branchement de votre moteur et comment adresser chaque moteur.

Pour une alimentation, j'ai utilisé un porte-piles avec quatre piles de type AA. N'importe quelle alimentation 5 ou 6 volts devrait suffire, surtout si vous utilisez de petits servomoteurs SG90

Code ESP32 PCA9685

Adafruit est une excellente source pour obtenir des modules PCA9685, et si vous connaissez Adafruit, vous saurez qu'ils ne vendent rien sans créer une documentation complète, des articles, des exemples de code et des bibliothèques. Le PCA9685 n'est pas différent, nous utiliserons donc une bibliothèque Adafruit qui fonctionne à la fois avec l'Arduino et l'ESP32.

Voici le croquis utilisé pour contrôler les deux servomoteurs à l'aide de l'ESP32 et du PCA9685.

PCA9685_ESP32.ino

```
/*
  ESP32 PCA9685 Servo Control
  esp32-pca9685.ino
  Driving multiple servo motors with ESP32 and PCA9685 PWM module
  Use I2C Bus

  DroneBot Workshop 2020
  https://dronebotworkshop.com
*/

// Include Wire Library for I2C
#include <Wire.h>

// Include Adafruit PCA9685 Servo Library
#include <Adafruit_PWMServoDriver.h>
```

```
// Creat object to represent PCA9685 at default I2C address
Adafruit_PWMServoDriver pca9685 = Adafruit_PWMServoDriver(0x40);

// Define maximum and minimum number of "ticks" for the servo motors
// Range from 0 to 4095
// This determines the pulse width

#define SERVOMIN  80  // Minimum value
#define SERVOMAX  600  // Maximum value

// Define servo motor connections (expand as required)
#define SER0  0   //Servo Motor 0 on connector 0
#define SER1  12  //Servo Motor 1 on connector 12

// Variables for Servo Motor positions (expand as required)
int pwm0;
int pwm1;

void setup() {

  // Serial monitor setup
  Serial.begin(115200);

  // Print to monitor
  Serial.println("PCA9685 Servo Test");

  // Initialize PCA9685
  pca9685.begin();

  // Set PWM Frequency to 50Hz
  pca9685.setPWMFreq(50);

}

void loop() {

  // Move Motor 0 from 0 to 180 degrees
  for (int posDegrees = 0; posDegrees <= 180; posDegrees++) {

    // Determine PWM pulse width
    pwm0 = map(posDegrees, 0, 180, SERVOMIN, SERVOMAX);
    // Write to PCA9685
    pca9685.setPWM(SER0, 0, pwm0);
    // Print to serial monitor
    Serial.print("Motor 0 = ");
    Serial.println(posDegrees);
    delay(30);
  }

  // Move Motor 1 from 180 to 0 degrees
```

```
  for (int posDegrees = 180; posDegrees >= 0; posDegrees--) {

    // Determine PWM pulse width
    pwm1 = map(posDegrees, 0, 180, SERVOMIN, SERVOMAX);
    // Write to PCA9685
    pca9685.setPWM(SER1, 0, pwm1);
    // Print to serial monitor
    Serial.print("Motor 1 = ");
    Serial.println(posDegrees);
    delay(30);
  }

  // Move Motor 0 from 180 to 0 degrees
  for (int posDegrees = 180; posDegrees >= 0; posDegrees--) {

    // Determine PWM pulse width
    pwm0 = map(posDegrees, 0, 180, SERVOMIN, SERVOMAX);
    // Write to PCA9685
    pca9685.setPWM(SER0, 0, pwm0);
    // Print to serial monitor
    Serial.print("Motor 0 = ");
    Serial.println(posDegrees);
    delay(30);
  }


  // Move Motor 1 from 0 to 180 degrees
  for (int posDegrees = 0; posDegrees <= 180; posDegrees++) {

    // Determine PWM pulse width
    pwm1 = map(posDegrees, 0, 180, SERVOMIN, SERVOMAX);
    // Write to PCA9685
    pca9685.setPWM(SER1, 0, pwm1);
    // Print to serial monitor
    Serial.print("Motor 1 = ");
    Serial.println(posDegrees);
    delay(30);
  }


}
```