QUICK REFERENCE · PCA9685

PCA9685 Servo Driver

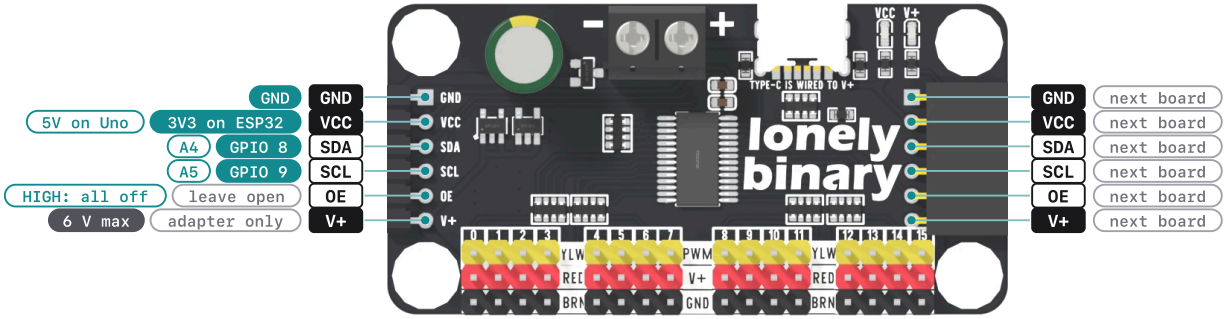
Sixteen servo channels on two I²C wires, 12-bit, powered from USB-C or a screw terminal.

V+ 6 V at most. VCC at your board's logic voltage.

V+ runs the servos, from a 5 V adapter into the USB-C socket or the terminal, never a computer's USB port; above 6 V the 5 V regulator on V+ is past its rating. VCC runs the chip, 3 to 5 V, and **SDA and SCL are pulled up to it: 5 V on VCC puts 5 V on an ESP32's pins.**

The header, and what to wire

Drawn from above. The left edge's pins go to your board; the socket on the right carries the same six in the same order, for the next board to plug into. Filled tags are the ESP32-S3, outlined ones the Uno.



VCC sets the bus voltage

Both 10 kΩ pull-ups on SDA and SCL go to VCC, not to your board. Wire VCC to 3V3 on an ESP32 or a Pico and to 5V on an Uno. Or bridge the 3V3 pad underneath and VCC comes from V+ through the board's own regulator; then leave the VCC pin unwired. Never bridge both pads.

What the supply has to carry

4 × SG90, moving

0.8 A, about

Inside a 5 V 3 A adapter.

A 0.80

16 × SG90, moving

3.2 A, about

1.1 times a 3 A adapter.

A 3.20

4 × MG996R, stalled

10.0 A, about

3.3 times a 3 A adapter.

A 10.00

Size V+ for the servos that can stall at once, not for the ones at rest. A 5 V 3 A adapter runs a handful of small servos; four big ones holding a load need a bench supply.

Arithmetic on typical currents for the class, not measurements: about 0.2 A moving for a 9 g servo, about 2.5 A stalled for an MG996R. The 1000 µF capacitor on V+ covers the milliseconds when several start together, not a second of stall.

The first servo, on channel 0

Arduino IDE · ESP32S3 Dev Module (USB CDC On Boot: Enabled) or Arduino Uno · Library: Adafruit PWM Servo Driver Library · Serial Monitor 115200

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>

// 25 MHz is the datasheet's typical figure, not a
// promise. Adafruit's servo example assumes 27 MHz.
// Measure yours: osc = Hz x 4096 x (prescale + 1).
const uint32_t OSC_HZ = 25000000;

const uint8_t CHANNEL = 0;
const uint16_t END_A = 1000;    // us
const uint16_t END_B = 2000;    // us

Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver(0x40);

void moveTo(uint16_t us) {
  pwm.writeMicroseconds(CHANNEL, us);
  Serial.print(us);
  Serial.println(" us");
}

void setup() {
  Serial.begin(115200);
  delay(1000);

#if defined(ESP32)
  Wire.begin(8, 9);             // SDA, SCL
#else
  Wire.begin();                 // A4, A5 on an Uno
#endif

  if (!pwm.begin()) {
    Serial.println("No PCA9685 at 0x40.");
    Serial.println("Check GND, VCC, SDA and SCL.");
    while (true) delay(1000);
  }
  pwm.setOscillatorFrequency(OSC_HZ);
  pwm.setPWMFreq(50);           // one pulse every 20 ms
  Serial.println("PCA9685 at 0x40, 50 Hz");
}

void loop() {
  moveTo(END_A);
  delay(1500);
  moveTo(END_B);
  delay(1500);
}
```

Prints `PCA9685 at 0x40, 50 Hz`, then `1000 us` and `2000 us` as the servo swings end to end. If it buzzes at one end, bring END_A up or END_B down by 50.

Wiring

PIN	ESP32-S3	UNO
GND	GND	GND
VCC	3V3	5V
SDA	GPIO 8	A4
SCL	GPIO 9	A5
OE	–	–
V+	–	–

OE: 10 kΩ to GND on the board, so outputs are on. V+: the adapter feeds it. Servo on channel 0: brown or black to GND, red to V+, yellow or orange to PWM.

Address pads, underneath

ADDRESS	BRIDGED
0x40	none
0x41	A0
0x42	A1
0x43	A0 + A1
0x70	A4 + A5

Open is 0, bridged is 1: 0x40 plus A5..A0. **Skip 0x70**: every chip also answers it as the All Call address. VCC pads: **3V3** for 3.3 V, **5V** for a little under 5 V off a 5 V adapter, one or neither.

When it does not work

The board never answers

Check the LEDs: red is VCC, blue is V+. Red dark means the chip has no supply. Then swapped SDA and SCL, or no shared GND.

The servo buzzes at the ends

The pulse is past its end stop. Use writeMicroseconds from 1000 to 2000 and widen 50 µs at a time.

Every servo a few degrees off

The oscillator constant is not your chip's. Measure a channel's period and correct it.

It answers, nothing moves

Blue dark: V+ has no supply, and the chip takes every command. Or OE is wired HIGH, which turns every channel off.

The board resets when servos move

V+ is sagging. A dedicated adapter, not a computer port or your board's 5V pin, and sized for stall.

Two boards, one answers

Both are at 0x40. Bridge a pad on the second one, and never make one 0x70.

Specifications

Chip	NXP PCA9685, TSSOP-28	**Channels**	16, 12-bit, one shared frequency
PWM	24 to 1526 Hz; servos at 50 Hz	**Addresses**	0x40 to 0x7F, six pads
VCC	3 to 5 V, sets the I²C level	**V+**	3.3 to 6 V, reverse-protected
Per channel	220 Ω in series	**Board**	64 × 32 mm, four Ø4.8 mm holes