Test run of servos and stepper motor using Arduino

Connecting to the Arduino

The PWM/Servo Driver uses I2C so it take only 4 wires to connect to your Arduino:  
  
**"Classic" Arduino wiring:**

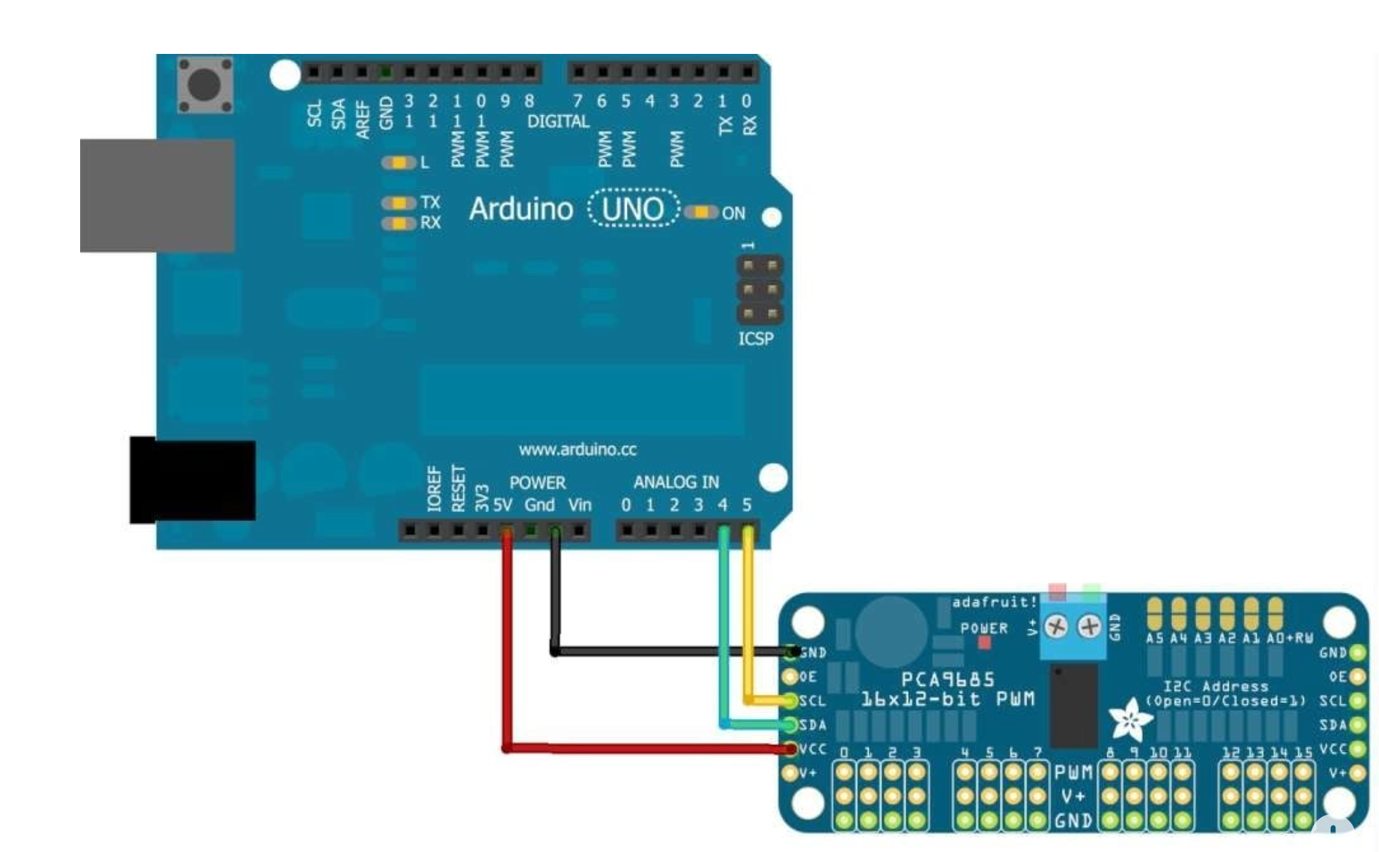
* +5v -> VCC (this is power for the BREAKOUT only, NOT the servo power!)
* GND -> GND
* Analog 4 -> SDA
* Analog 5 -> SCL

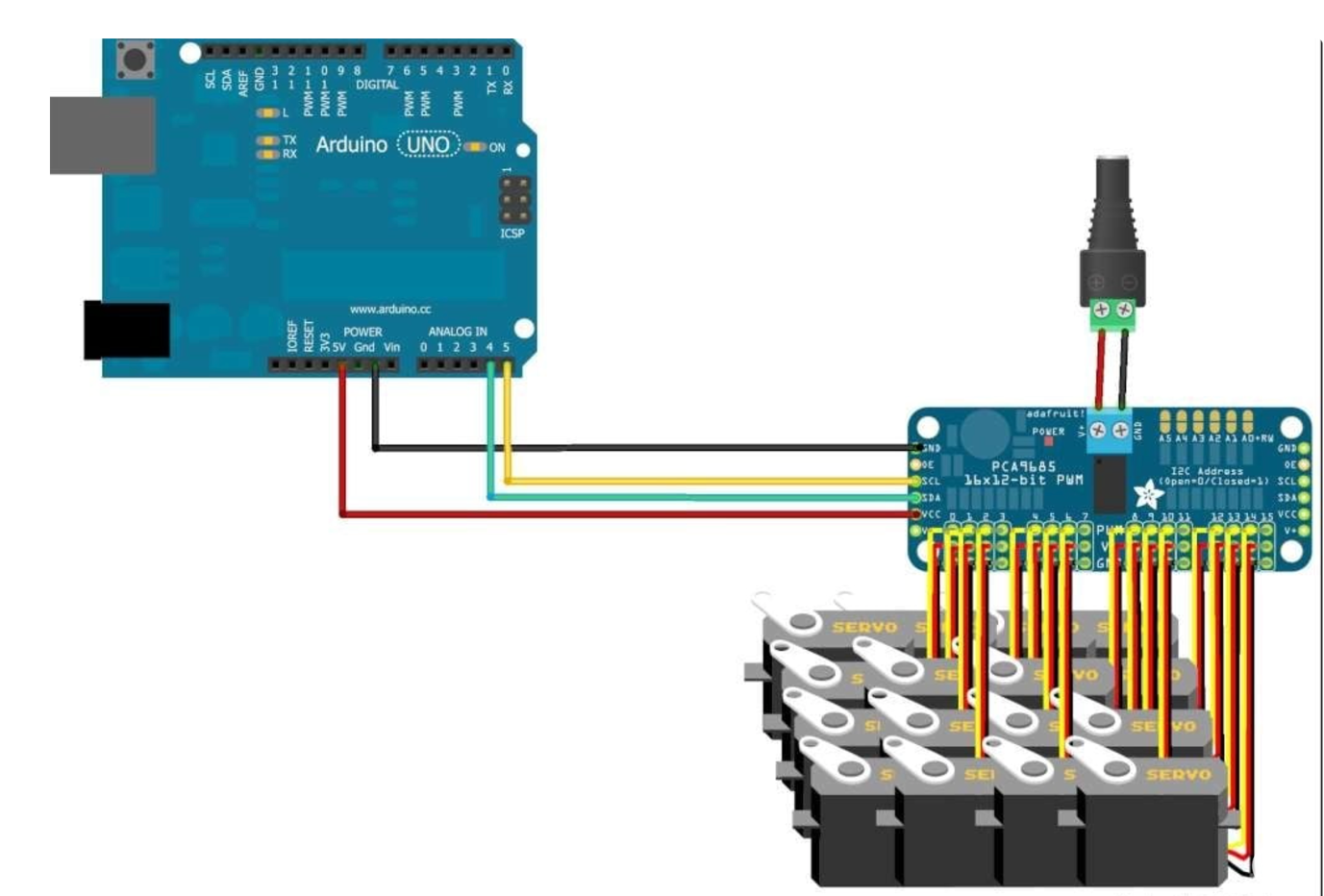
**Older Mega wiring:**

* +5v -> VCC (this is power for the BREAKOUT only, NOT the servo power!)
* GND -> GND
* Digital 20 -> SDA
* Digital 21 -> SCL

**R3 and later Arduino wiring (Uno, Mega & Leonardo):**(These boards have dedicated SDA & SCL pins on the header nearest the USB connector)

* +5v -> VCC (this is power for the BREAKOUT only, NOT the servo power!)
* GND -> GND
* SDA -> SDA
* SCL -> SCL





* The VCC pin is just power for the chip itself. If you want to connect servos or LEDs that use the V+ pins, you MUST connect the V+ pin as well. The V+ pin can be as high as 6V even if VCC is 3.3V (the chip is 5V safe). We suggest connecting power through the blue terminal block since it is polarity protected.

**TEST CODE(FOR 6 SERVOS):**

#include <Wire.h>

#include <Adafruit\_PWMServoDriver.h>

Adafruit\_PWMServoDriver pwm = Adafruit\_PWMServoDriver();

#define SERVOMIN 150 // Minimum pulse length count

#define SERVOMAX 600 // Maximum pulse length count

void setup() {

Serial.begin(9600);

pinMode(2,OUTPUT);//for output enable pin

digitalWrite(2,LOW);

pwm.begin();

pwm.setPWMFreq(50); // Set frequency to 50 Hz for servos

delay(10); // Short delay to stabilize

}

void loop() {

for (int pulselen = SERVOMIN; pulselen <= SERVOMAX; pulselen=pulselen+1) {

pwm.setPWM(0, 0, pulselen);

pwm.setPWM(1,0,pulselen);

pwm.setPWM(2,0,pulselen);

pwm.setPWM(3,0,pulselen);

pwm.setPWM(4,0,pulselen);

pwm.setPWM(5,0,pulselen);

delay(10);

}

for (int pulselen = SERVOMAX; pulselen >= SERVOMIN; pulselen=pulselen-1) {

pwm.setPWM(0, 0, pulselen);

pwm.setPWM(1,0,pulselen);

pwm.setPWM(2,0,pulselen);

pwm.setPWM(3,0,pulselen);

pwm.setPWM(4,0,pulselen);

pwm.setPWM(5,0,pulselen);

delay(10);

}

digitalWrite(2,HIGH);

delay(10000);

digitalWrite(2,LOW);

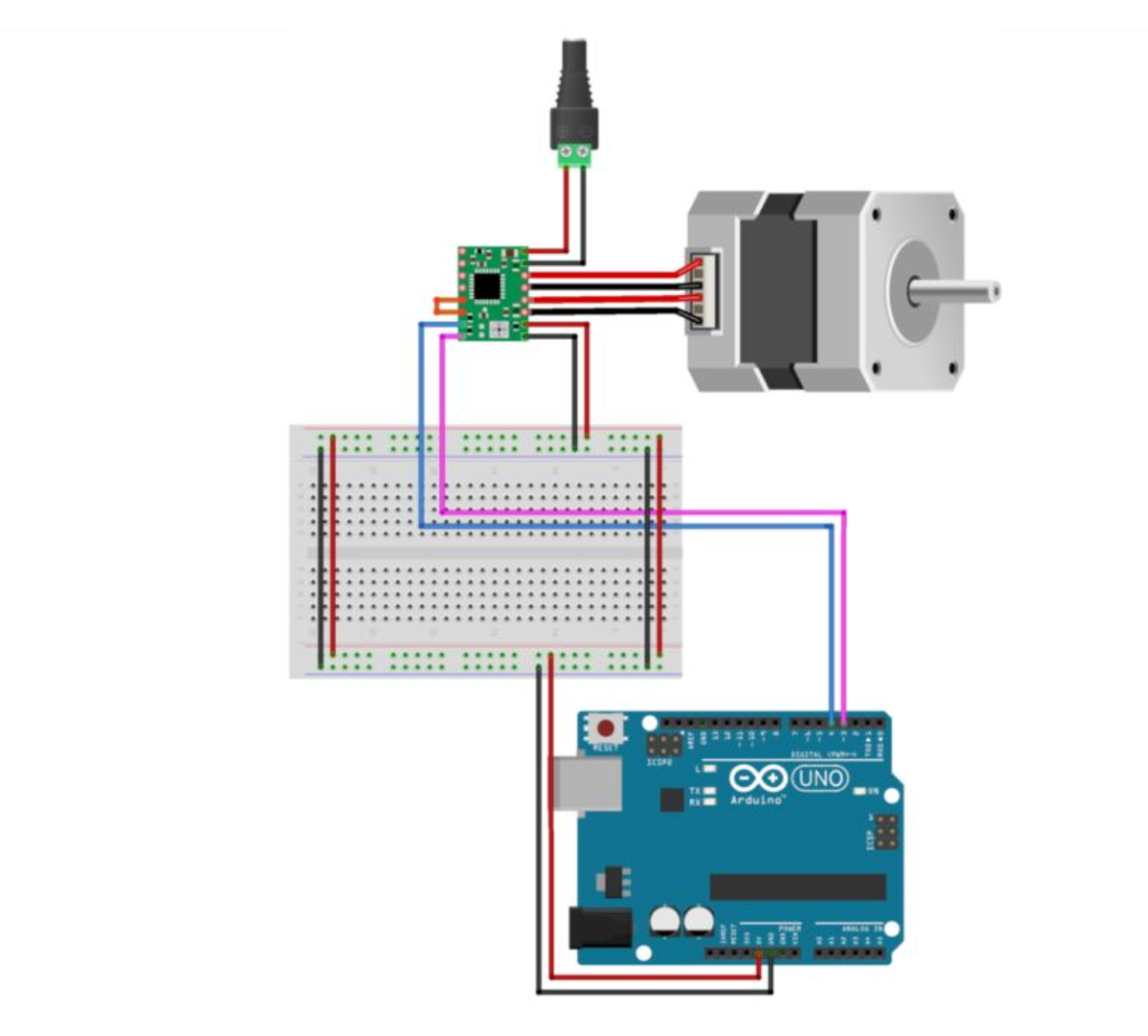
//You can control other servos similarly

//pwm.setPWM(1, 0, 375); // Center position for servo on channel 1

//delay(1000); // Hold position for 1 second

}

**STEPPER MOTOR - CONNECTION:**



**TEST CODE(FOR STEPPER MOTOR USING A4988):**

// Define connection pins

#define STEP\_PIN 3 // Connect to STEP pin of A4988

#define DIR\_PIN 4 // Connect to DIR pin of A4988

#define ENABLE\_PIN 2 // Connect to ENABLE pin of A4988 (optional)

// Define steps per revolution

int stepsPerRevolution = 200; // Adjust this based on your motor

void setup() {

// Set pin modes

pinMode(STEP\_PIN, OUTPUT);

pinMode(DIR\_PIN, OUTPUT);

pinMode(ENABLE\_PIN, OUTPUT);

// Enable the driver

digitalWrite(ENABLE\_PIN, LOW); // LOW to enable the driver

}

void loop() {

// Rotate clockwise

digitalWrite(DIR\_PIN, HIGH); // Set direction to clockwise

for (int i = 0; i < stepsPerRevolution; i++) {

stepMotor(500); // 500 microseconds delay (adjust for speed)

}

delay(1000); // Wait for 1 second

// Rotate counterclockwise

digitalWrite(DIR\_PIN, LOW); // Set direction to counterclockwise

for (int i = 0; i < stepsPerRevolution; i++) {

stepMotor(500); // 500 microseconds delay (adjust for speed)

}

delay(1000); // Wait for 1 second

}

// Function to take one step

void stepMotor(int stepDelay) {

digitalWrite(STEP\_PIN, HIGH);

delayMicroseconds(stepDelay); // Time for high pulse

digitalWrite(STEP\_PIN, LOW);

delayMicroseconds(stepDelay); // Time for low pulse

}

COMBINIG BOTH THE TEST CODES :

#include <Wire.h>

#include <Adafruit\_PWMServoDriver.h>

#define STEP\_PIN 3 // Stepper motor STEP pin

#define DIR\_PIN 4 // Stepper motor DIR pin

#define ENABLE\_PIN 2 // Stepper motor ENABLE pin (also used in your existing servo control)

Adafruit\_PWMServoDriver pwm = Adafruit\_PWMServoDriver();

#define SERVOMIN 150 // Minimum pulse length count for servo

#define SERVOMAX 600 // Maximum pulse length count for servo

int stepsPerRevolution = 200; // Steps for one full rotation

int stepDelay = 500; // Initial delay for stepper motor speed

void setup() {

Serial.begin(9600);

// Stepper motor setup

pinMode(STEP\_PIN, OUTPUT);

pinMode(DIR\_PIN, OUTPUT);

pinMode(ENABLE\_PIN, OUTPUT);

digitalWrite(ENABLE\_PIN, LOW); // Enable stepper motor driver

// Servo driver setup

pwm.begin();

pwm.setPWMFreq(50); // Set frequency to 50 Hz for servos

delay(10); // Short delay for stabilization

}

void loop() {

// 1. Control stepper motor (clockwise rotation)

digitalWrite(DIR\_PIN, HIGH); // Set stepper direction

for (int i = 0; i < stepsPerRevolution; i++) {

stepMotor(stepDelay);

}

delay(1000); // Pause for 1 second

// 2. Control stepper motor (counterclockwise rotation)

digitalWrite(DIR\_PIN, LOW); // Set stepper direction

for (int i = 0; i < stepsPerRevolution; i++) {

stepMotor(stepDelay);

}

delay(1000); // Pause for 1 second

// 3. Sweep servos back and forth

for (int pulselen = SERVOMIN; pulselen <= SERVOMAX; pulselen += 1) {

for (int ch = 0; ch < 6; ch++) {

pwm.setPWM(ch, 0, pulselen); // Sweep all 6 servos

}

delay(10); // Small delay for smooth movement

}

for (int pulselen = SERVOMAX; pulselen >= SERVOMIN; pulselen -= 1) {

for (int ch = 0; ch < 6; ch++) {

pwm.setPWM(ch, 0, pulselen); // Sweep all 6 servos

}

delay(10); // Small delay for smooth movement

}

// 4. Optional: Disable stepper temporarily

digitalWrite(ENABLE\_PIN, HIGH); // Disable stepper motor

delay(10000); // Wait for 10 seconds

digitalWrite(ENABLE\_PIN, LOW); // Enable stepper motor again

}

// Function to control the stepper motor (one step)

void stepMotor(int delayTime) {

digitalWrite(STEP\_PIN, HIGH);

delayMicroseconds(delayTime); // Control speed

digitalWrite(STEP\_PIN, LOW);

delayMicroseconds(delayTime); // Control speed

}