Circuit connections

| **TB6612FNG Pin** | **Arduino Nano Pin** | **Description** |
| --- | --- | --- |
| AIN1 | D4 | Left motors direction 1 |
| AIN2 | D5 | Left motors direction 2 |
| PWMA | D6 | Left motors speed (PWM) |
| BIN1 | D7 | Right motors direction 1 |
| BIN2 | D8 | Right motors direction 2 |
| PWMB | D9 | Right motors speed (PWM) |
| STBY | 5V | Enable driver |
| VM | 7.4 V battery + | Motor power supply |
| VCC | 5V | Logic power |
| GND | Common GND | Common ground |

| **Motor Side** | **TB6612 Pins** | **Description** |
| --- | --- | --- |
| Left 2 Motors | A01 & A02 | Both connected in parallel |
| Right 2 Motors | B01 & B02 | Both connected in parallel |

| **Joystick Pin** | **Arduino Pin** | **Function** |
| --- | --- | --- |
| VRx | A0 | Forward/Backward speed |
| VRy | — | Not used |
| SW | — | Not used |
| +5V | 5V | Power |
| GND | GND | Ground |

| **Joystick Pin** | **Arduino Pin** | **Function** |
| --- | --- | --- |
| VRx | A1 | Left/Right turn |
| VRy | — | Not used |
| SW | — | Not used |
| +5V | 5V | Power |
| GND | GND | Ground |

**Power**

* **Battery (2×18650 = 7.4 V)** → TB6612 **VM**
* **Arduino Nano** → powered via **USB** or **5 V pin**
* **All GNDs must be common** (Arduino, TB6612, and battery)

// TB6612FNG Joystick Controlled 4-Motor Robot

// Front/Back = Joystick1 (A0)

// Left/Right = Joystick2 (A1)

#define AIN1 4

#define AIN2 5

#define PWMA 6

#define BIN1 7

#define BIN2 8

#define PWMB 9

#define JOY\_FB A0 // Forward/Backward joystick (VRx)

#define JOY\_LR A1 // Left/Right joystick (VRx)

// Deadzone range (to avoid jitter)

#define DEADZONE 100

void setup() {

pinMode(AIN1, OUTPUT);

pinMode(AIN2, OUTPUT);

pinMode(BIN1, OUTPUT);

pinMode(BIN2, OUTPUT);

pinMode(PWMA, OUTPUT);

pinMode(PWMB, OUTPUT);

Serial.begin(9600);

}

void loop() {

int fb = analogRead(JOY\_FB) - 512; // Forward/Backward

int lr = analogRead(JOY\_LR) - 512; // Left/Right

// Apply deadzone

if (abs(fb) < DEADZONE) fb = 0;

if (abs(lr) < DEADZONE) lr = 0;

// Combine joystick input to get motor speed

int leftMotorSpeed = fb + lr;

int rightMotorSpeed = fb - lr;

// Constrain PWM values

leftMotorSpeed = constrain(leftMotorSpeed, -512, 512);

rightMotorSpeed = constrain(rightMotorSpeed, -512, 512);

// Drive left motors

if (leftMotorSpeed > 0) {

digitalWrite(AIN1, HIGH);

digitalWrite(AIN2, LOW);

} else if (leftMotorSpeed < 0) {

digitalWrite(AIN1, LOW);

digitalWrite(AIN2, HIGH);

} else {

digitalWrite(AIN1, LOW);

digitalWrite(AIN2, LOW);

}

analogWrite(PWMA, abs(leftMotorSpeed) / 2);

// Drive right motors

if (rightMotorSpeed > 0) {

digitalWrite(BIN1, HIGH);

digitalWrite(BIN2, LOW);

} else if (rightMotorSpeed < 0) {

digitalWrite(BIN1, LOW);

digitalWrite(BIN2, HIGH);

} else {

digitalWrite(BIN1, LOW);

digitalWrite(BIN2, LOW);

}

analogWrite(PWMB, abs(rightMotorSpeed) / 2);

delay(10);

}