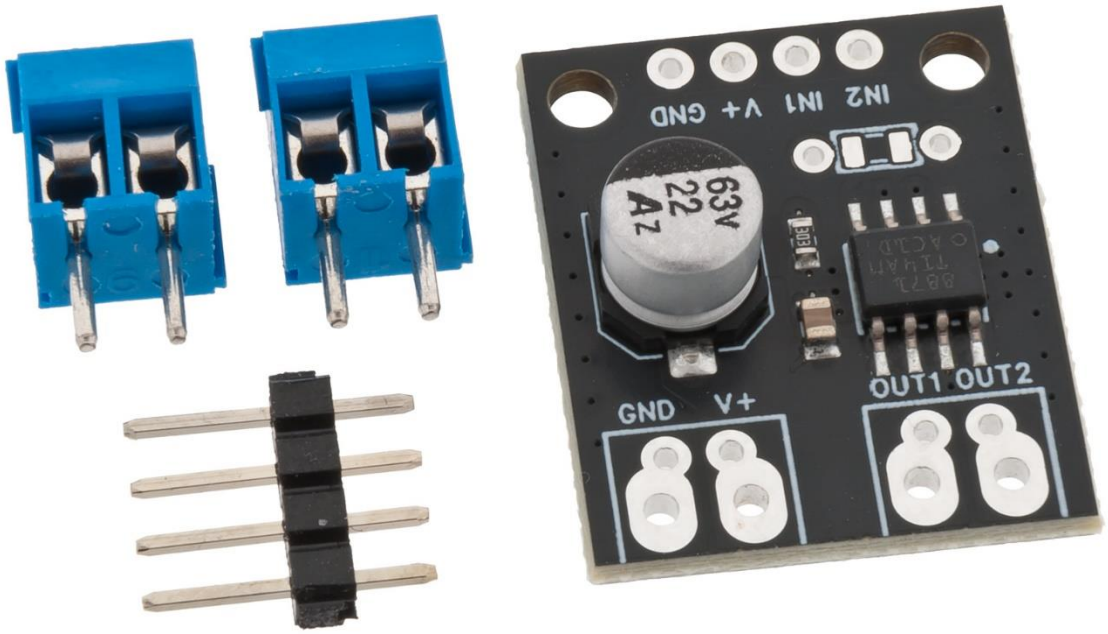
DRV8871 DC Motor Driver

Part.no. 41024460



Small breakout board for Texas Instruments DRV8871 Single channel DC motor driver. The driver is controlled with two PWM signals and will drive a connected DC motor in both directions. The inputs will accept both a static DC voltage to run the motor at 100% or PWM signals to vary the speed. When both inputs are low, the motor is disconnected from the driver outputs and is free to spin (coast). When both inputs are high, the motor will be braked and resist rotation.

The driver also contains a clever current limiting that can be programmed with a single resistor. Other protection features are undervoltage lockout, overcurrent protection and thermal shutdown.

In short, a very capable, low-cost, easy-to-use and reliable motor controller for all small brushed DC motors.

Included hardware:

- 2x 3.5mm 2-pin screw terminals
- 1x 2.54mm 4-pin header

Functions

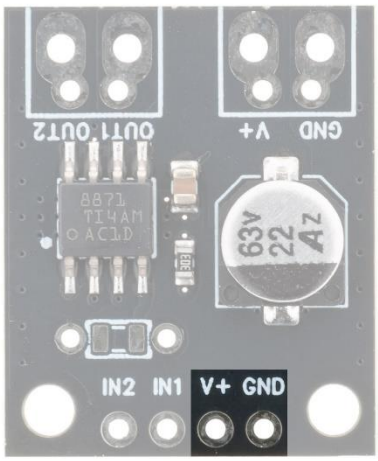
- Simple bidirectional PWM motor control
- Programmable current limiting
- Thermal protection
- Wide motor voltage support

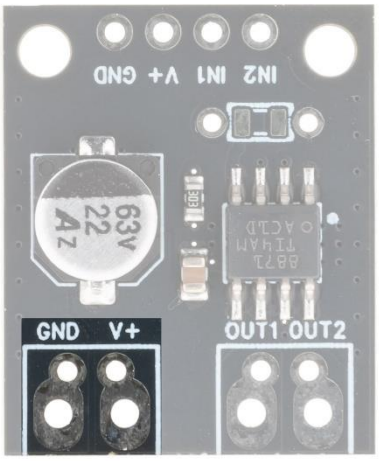
Specifications

- Supply voltage: 6.5 - 45 VDC
- Output current: max 3.6 A
- RDSon: 565 mOhm
- PWM frequency: max 200 kHz
- Logic level: min 1.5 V
- Dimensions: 24 x 20 mm
- Mounting holes: c-c 15 mm / ø2.5 mm

Connections

Supply voltage:



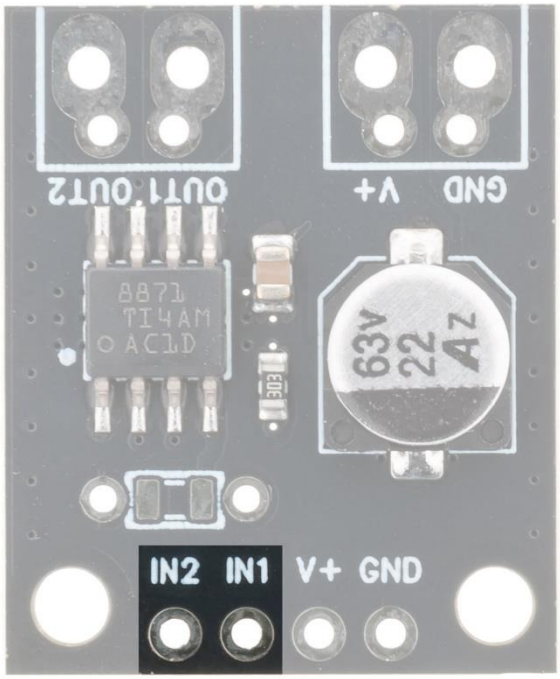


The supply voltage range is 6.5 - 45 VDC and should be connected to EITHER the pins adjacent to IN2/IN1 OR via the screw terminals on the output side of the board. Please note that the GND pins must be connected to both the microcontroller and external power supply to establish a common ground for the driver and logic signals.

The power supply should be able to provide enough current even during peaks when the motor is stalled. Check the motor specifications to determine peak (or stall) current.

The driver has an undervoltage lockout and will be disabled when the supply voltage drops below 6.5V.

Control inputs:



Using DRV8871 is very simple and requires just two control pins to make the motor go forward, backward, break or coast.

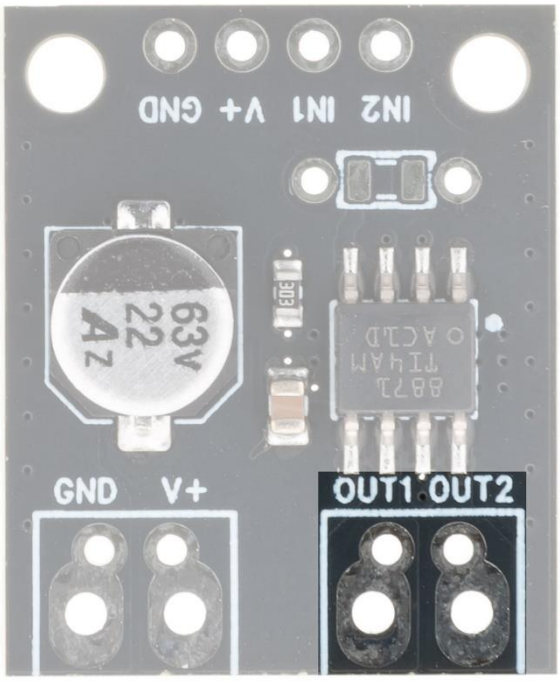
The logic signals can be as low as 1.5V and will work with both 3.3V and 5V systems.

Both inputs are active-high and have internal pulldown resistors to GND.

Don't forget to connect GND to the microcontroller!

IN1	IN2	OUT1	OUT2	DESCRIPTION
0	0	High-Z	High-Z	Coast; H-bridge disabled to High-Z (sleep entered after 1 ms)
0	1	L	H	Reverse (Current OUT2 → OUT1)
1	0	H	L	Forward (Current OUT1 → OUT2)
1	1	L	L	Brake; low-side slow decay

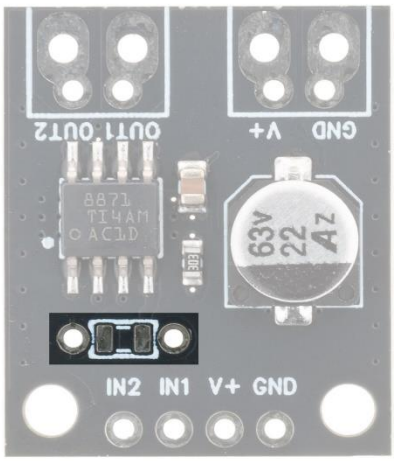
Motor outputs:

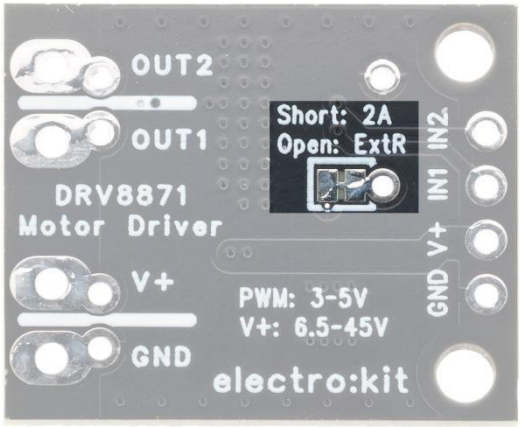


The motor outputs have dual patterns for either 2.54mm headers or 3.5mm screw terminals (included).

If vertical space is limited, wires can also be soldered directly to the board.

Current limiting:





DRV8871 features a simple current limiting using just one resistor. The breakout board comes with a resistor pre-installed for a limit of 2A. The default current limit is usually enough to drive most small motors. It can be changed to allow the full 3.5A the driver is capable of, or to limit the current further to protect very small motors.

To change the current limit, first break the solder jumper on the back. The jumper can be severed using a sharp knife. Measure continuity between the two pads with a multimeter to ensure the jumper is fully open. The second step is to determine the new current limit and which resistor to install. Minimum allowed resistor is 15kohm (4.2A limit).

The resistor can be calculated using the following formula:

I_{TRIP} (A) = 64 / R_{ILIM} (kΩ)

The default resistor is 30k and thus:

I_{TRIP} (A) = 64 / 30 (kΩ)= 2.13A

For a current limit of ~1A:

I_{TRIP} (A) = 64 / 62 (kΩ) = 1.03A

The breakout board has patterns for both an SMD 0805 resistor and a PTH 1/8W resistor.

Arduino example code

```
/*

Example code for EKM023 DRV8871 Motor Driver Breakout.

Connect IN1 and IN2 to arduino pins 9 and 10.
Connect GND to the Arduino, and if not using a separate power supply for the motor, V+ to
5V.
NOTE! If using a separate motor power supply, leave V+ UNCONNECTED.

This example sketch will showcase how to drive a connected DC motor both forward and
reverse.
The sketch will:
 - Slowly ramp up the speed in one direction
 - Hold at 100% for a set time (holdDuration)
 - Ramp back down to 0%
 - Hold at 0% for a set time (holdDuration)

The sketch will also showcase the difference between breaking and coasting.
Change the variable "breakMode" from true to false to observe the difference.
"Break" will power both IN1 and IN2 to energize the motor coils, effectively applying
breaks.
"Coast" will instead keep both inputs low and leave the motor unpowered.

*/

// Motor pins
const int IN1 = 9;      // PWM
const int IN2 = 10;     // PWM
bool brakeMode = true;  // true = brake, false = coast

// Parameters
const int maxPWM = 255;
const int rampDuration = 2000;  // Total time (ms) for one ramp up or down
const int holdDuration = rampDuration; // Time to stay in 0% and 100% to demonstrate
difference between break and coast

const int numSteps = maxPWM + 1;
const int stepDelay = rampDuration / numSteps;

void setup() {
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  Serial.begin(9600);
  Serial.println(" ");
  Serial.println("------ ------ ------ ------ ------ ------ ");
  Serial.println("=== DRV8871 Motor Demo ===");
  Serial.print("Brake mode: ");
  Serial.println(brakeMode ? "Braking" : "Coasting");
  Serial.print("Ramp duration: ");
  Serial.print(rampDuration);
  Serial.println(" ms");
  Serial.print("Hold duration: ");
  Serial.print(holdDuration);
  Serial.println(" ms");
  Serial.println("------ ------ ------ ------ ------ ------ ");
}

// Set motor speed (-255 to 255)
void setMotorSpeed(int speed) {
  int pwm = abs(speed);

  if (speed > 0) {
    analogWrite(IN1, pwm);
```

```
    if (brakeMode) {
      digitalWrite(IN2, LOW);
    } else {
      analogWrite(IN2, 0);
    }
  } else if (speed < 0) {
    analogWrite(IN2, pwm);
    if (brakeMode) {
      digitalWrite(IN1, LOW);
    } else {
      analogWrite(IN1, 0);
    }
  } else {
    if (brakeMode) {
      digitalWrite(IN1, HIGH);
      digitalWrite(IN2, HIGH);
    } else {
      digitalWrite(IN1, LOW);
      digitalWrite(IN2, LOW);
    }
  }

  //Serial.print("Speed: ");
  //Serial.print(speed);
  //Serial.print(" | PWM: ");
  //Serial.println(pwm);
}

void rampRun(int direction) {
  const char* dirName = (direction > 0) ? "Forward" : "Reverse";

  // Ramp up
  Serial.print(dirName); Serial.println(" ramp up");
  for (int pwm = 0; pwm <= maxPWM; pwm++) {
    setMotorSpeed(direction * pwm);
    delay(stepDelay);
  }

  // Hold at full speed
  Serial.print(dirName); Serial.println(" hold at full speed");
  setMotorSpeed(direction * maxPWM);
  delay(holdDuration);

  // Ramp down
  Serial.print(dirName); Serial.println(" ramp down");
  for (int pwm = maxPWM; pwm >= 0; pwm--) {
    setMotorSpeed(direction * pwm);
    delay(stepDelay);
  }

  // Hold at stop
  Serial.print("Hold at zero (");
  Serial.print(brakeMode ? "Braking" : "Coasting");
  Serial.println(")");
  setMotorSpeed(0);
  delay(holdDuration);
}

void loop() {
  rampRun(1);   // Forward
  rampRun(-1);  // Reverse
}
```

Additional resources

- DRV8871 Datasheet @ Texas Instruments

Mechanical dimensions