SPEED AND DIRECTION CONTROL FOR SMALL BRUSHED MOTORS

A PSUBS.ORG Community Project
submitted by
Jonathan Wallace
July 9, 2021

1) Overview

This document describes the use of inexpensive and readily available electronic devices to control the speed and direction of small motors that may serve a useful functional purpose on submarines. Small motors in this context are considered those that require 5-24 volts and draw less than 10 amps of current.

An assumption is made the user is familiar with the Arduino IDE or other tools for software installation onto a microprocessor. See references on the last page for links to tutorials and information if you are not familiar with this procedure.

The most basic parts of this project are shown in Figure 1 and include a microprocessor, a small motor controller, and a variable voltage input device such as a joystick or potentiometer. Depending upon the application you may also require a good buck converter to power the microprocessor, most of which accept a maximum voltage of 5vdc.

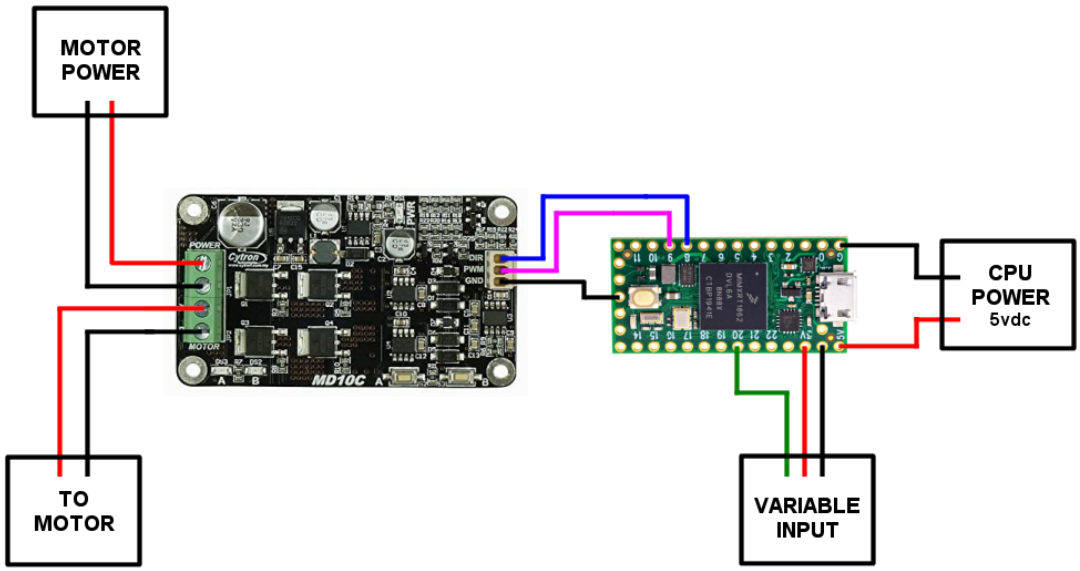


Figure 1.

This circuit illustrates a single input for directional control (clockwise-counterclockwise) of a motor and assumes a constant motor speed. If desired, motor speed (RPM) can be dynamically modified by simply adding another input for it or using software to adjust RPM based upon the single input as shown. For example, assuming the input is a joystick, perhaps when the joystick is pushed to the right a quarter of it's travel the motor turns at 25% it's maximum speed and when the joystick is pushed halfway of it's travel the motor turns at 50% it's maximum speed. Whether by additional inputs or by software, many configuration options are possible.

2) Motor Controller

The Cytron MD10C can control a single motor with a voltage of 5-24 vdc and a current draw up to 10 amps. It is an H-Bridge that provides direction and speed control of the motor via TTL inputs.

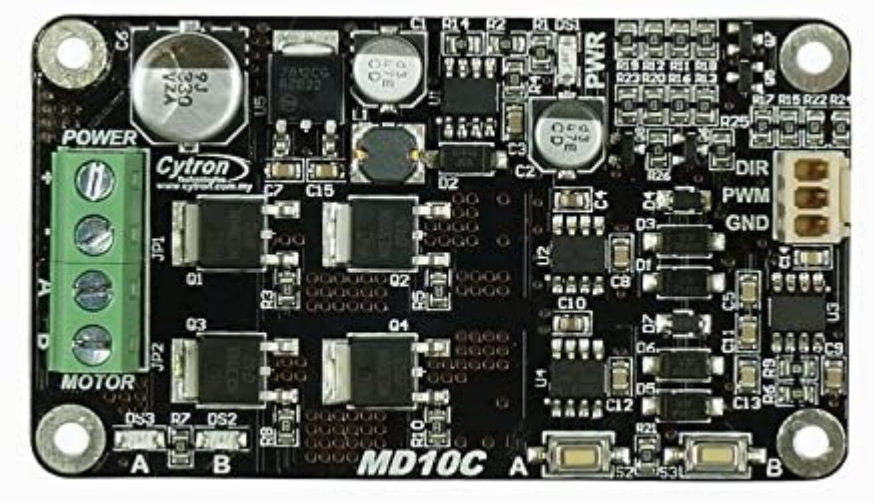


Figure 2.

Supply power for the motor is connected to the "+" and "-" terminals of the green connector shown on the left of the MD10C circuit board. This power must be the same as required by the motor (5-24vdc). The motor itself is connected to the "A" and "B" terminals. If the motor does not rotate in the desired direction to your input control when first connected, simply swap the motor wires between the "A" and "B" terminals.

TTL Inputs

Direction and speed for the motor is controlled by TTL inputs from a microprocessor which are supplied to the MD10C via the terminal on the right side of the circuit board. The DIR terminal takes a digital input of 0 or 1 and controls whether the motor rotates clockwise or counter-clockwise. The PWM terminal takes digital input in the form of a Pulse-Width Modulation signal to control the speed of the motor. The GND terminal is connected to the ground of the microprocessor electrical circuit.

3) Microprocessor

A TEENSY 4.0 microprocessor (Figure 3) is illustrated however an Arduino is also compatible and can be substituted if desired. In fact, any microprocessor that can accept analog input and supply both a digital and PWM signal can be used for this project however the provided software code shown will only work with TEENSY or Arduino and require porting if some other microprocessor is selected.

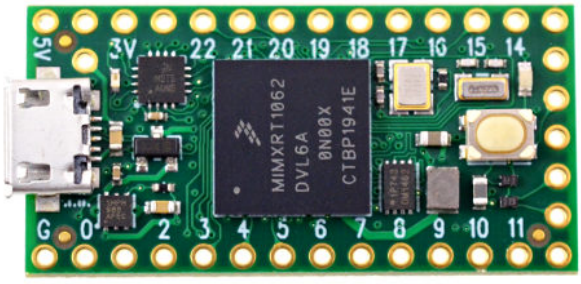
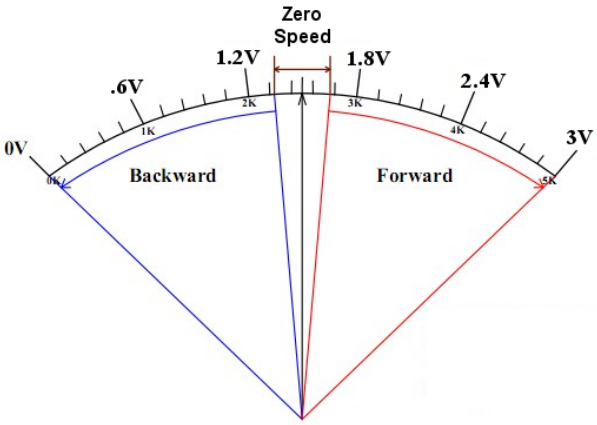


Figure 3.

4) Joystick

This document assumes use of a potentiometer based joystick as the input device to control left/right rotational direction of the motor. A Hall Effect joystick can be substituted if desired.

A "deadzone" is expected when the joystick is at the center position such that no motor rotation occurs. However, as can be seen by the "joystick deadzone" graphic in Figure 4 a positive voltage will still be present when the joystick is at its center position, thereby causing the motor to turn.



Joystick Deadzone

Figure 4.

This issue is resolved by creating a "virtual" joystick deadzone in software, and the supplied software program has such functionality within it. While the existing deadzone values in the software program should be sufficient for any joystick, changing these values can provide more or less joystick deadzone as desired. To modify the deadzone span, change the values of "DZL" (dead zone left) and "DZR" (dead zone right) in the software program. For DZL, any value between 0-450 is appropriate. For DZR, any value between 550-1023 is appropriate.

NOTE:
Before connecting the potentiometer output to the microprocessor, use a multimeter to ensure that you have wired the potentiometer such that pushing the joystick to the right increases voltage output, and pushing it left decreases voltage output.

5) Electrical Connections

Note that all electrical connections reference the TEENSY 4.0 microprocessor. If you substitute an Arduino or other microprocessor you will need to identify appropriate digital output, pwm output, and analog input pins for that particular processor and modify the supplied software code appropriately. Refer to Figure 1 for the following connections:

a) Microprocessor to MD10C controller.

i. Connect pin 8 on the TEENSY 4.0 microprocessor to the DIR terminal on the MD10C controller (purple wire).

ii. Connect pin 9 on the TEENSY 4.0 microprocessor to the PWM terminal on the MD10C controller (pink wire)

iii. Connect electrical ground from the TEENSY 4.0 microprocessor (see Figure 5) OR the overall electrical circuit, to the GND terminal on the MD10C.

NOTE: the electrical ground between the power supply for the motor and the TEENSY 4.0 microprocessor need to be from the same electrical circuit OR connected together if two separate power supplies are used (black wires).

iv. Connect the power supply for the motor to the green terminals on the MD10C controller marked “+” and “-”.

v. Connect the motor to the green terminals on the MD10C controller marked “A” and “B”.

NOTE: If the motor turns opposite of what you expected, reverse these wires.

b) Microprocessor to Joystick.

i. Connect pin “3v” on the TEENSY 4.0 microprocessor to the “+” voltage pin of your input device (red wire).

ii. Connect electrical ground from the TEENSY 4.0 microprocessor (see Figure 5) to the “-” voltage pin of your input device (black wire).

iii. Connect pin 20 on the TEENSY 4.0 microprocessor to the signal output pin of your input device (green wire).

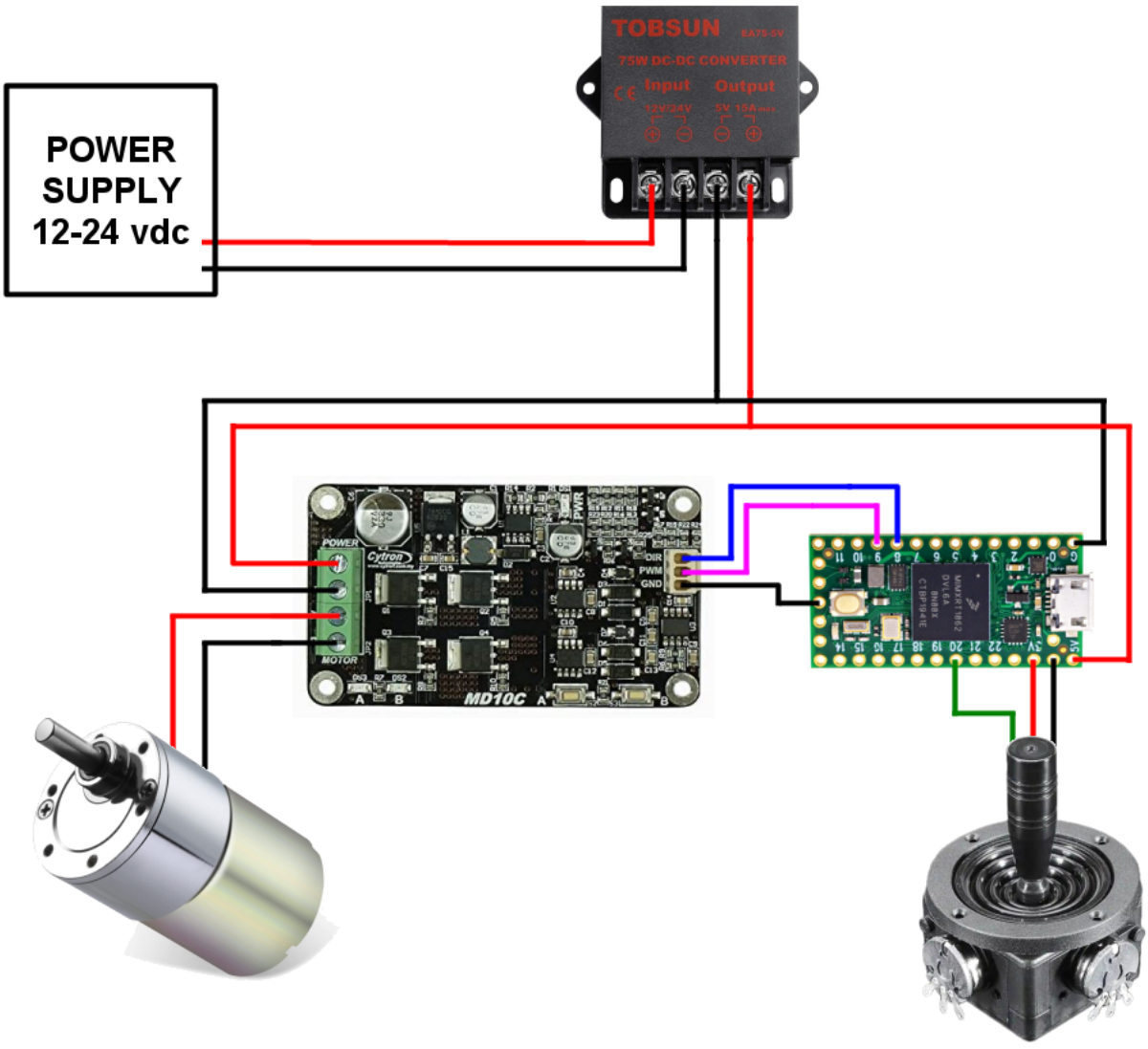
6) Software Program

The provided software will run without modification on a TEENSY 4.0 microprocessor. It is also compatible with any Arduino microprocessor except that the specific pins used to interface with the MD10C controller and Joystick will need to be modified before compiling and uploading. The software can be used as a guide for all other microprocessors. The most current version of this software can be found in Arduino IDE format at PSUBS.ORG under the Community Projects page.

```
// Change next three definitions to taste
#define DZL           300   // Dead Zone Left         (0-450)
#define DZR           700   // Dead Zone Right        (550-1023)
#define SPEED         120   // Motor Rotational Speed (0-255) 0=stopped, 255=full speed

// Change next three definintions based upon the pins
// you use on your microprocessor
#define JOYSTICK 20     // Analog input pin for Joystick
#define DIR           8       // Digital Output Pin
#define PWM           9       // PWM Output Pin


// ////////////////////////////////////////////////////////////////////////////
// ////////////////////////////////////////////////////////////////////////////
// Do not change anything else in the program from this point, forward.
// ////////////////////////////////////////////////////////////////////////////
// ////////////////////////////////////////////////////////////////////////////
#define PORT     0          // Rotate motor left
#define STBD     1           // Rotate motor right

int adc = 0;


void setup(void)
{
  pinMode(JOYSTICK, INPUT);
  pinMode(DIR, OUTPUT);
  pinMode(PWM, OUTPUT);
}


void loop(void)
{

  // Read joystick input
  adc = analogRead(JOYSTICK);

  if ( adc >= DZR ) {
     // Rotation right
     digitalWrite(DIR,STBD);
     analogWrite(PWM,SPEED);
  } else if ( adc <= DZL ) {
     // Rotation left
     digitalWrite(DIR,PORT);
     analogWrite(PWM,SPEED);
  } else {
     // Motor stopped
     analogWrite(PWM,0);
  }

}
```

7) Practical Layout

A practical layout of this motor controller is shown below including a TOBSUN “buck” converter which transforms an input of 12-24 vdc down to 5 vdc for both the motor and the microprocessor.



8) References.

Cytron MD10C motor controller
https://www.cytron.io/p-10amp-5v-30v-dc-motor-driver

TEENSY 4.0 Microprocessor
https://www.pjrc.com/store/teensy40.html

JH-D202X-R4 Mini Joystick
https://www.adafruit.com/product/3102

Arduino IDE Tutorials (software installation)
https://www.arduino.cc/en/Tutorial/HomePage