Caleb Hubbell
5/2/2018
Team 9

Motor Block Artifacts

Order:
Schematic
Block Diagram
Interfaces and Properties
Bill of Materials
Testing Procedure
Testing Results
Code

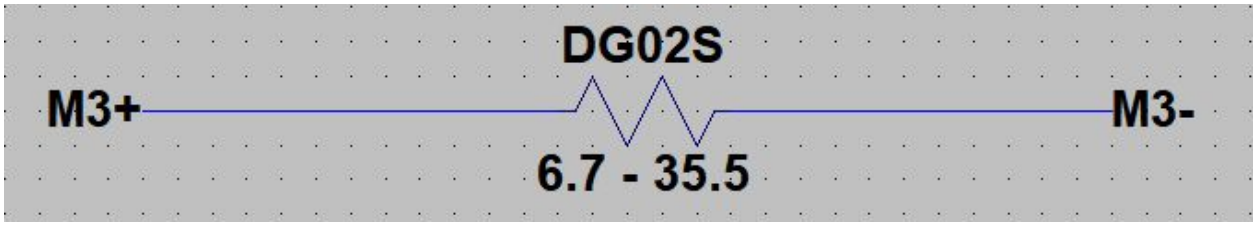

Figure 1: Wiring Schematic

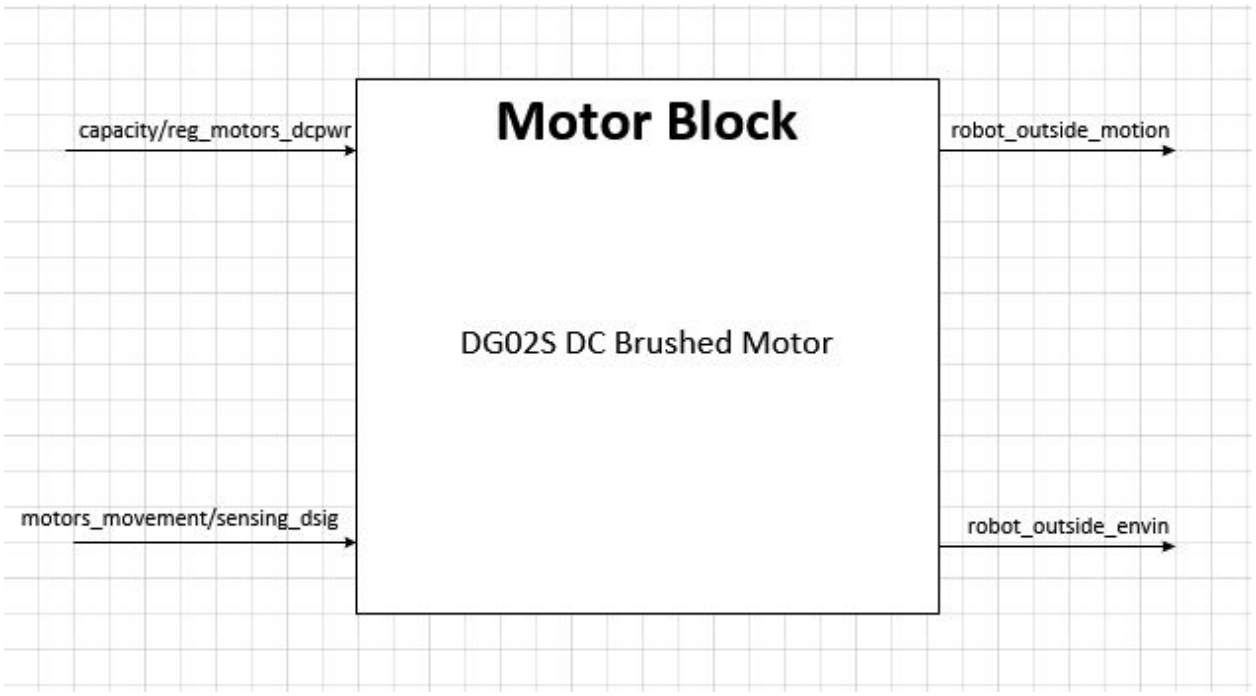

Figure 2: Block Diagram

Interface	Properties

Caleb Hubbell
5/2/2018
Team 9

robot_outside_envin	• Surfaces include: tile, short carpet, concrete • Obstacles include: walls, chair legs, cords, clothing
robot_outside_motion	• Speed on all surfaces must be greater than 0.3 m/s • Max speed > 0.3 m/s • 10 min or greater time in motion
capacity/reg_motors_dcpwr	• 3-12 V DC • 0 - 0.7 A • 10% tolerance on input parameters
motors_movement/sensing_dsig	• PWM DC signal powering 2 motors • PWM frequency of 1.6KHz. • Draw of 0 - 0.45A • Duty cycle of 0-40% • 2 active high inputs; 5V high, 0V low

Table 1: Block Interfaces and Properties

Item	Cost	Quantity	Link
2 DAGU DG02S	$9.90	1	https://www.ebay.com/i/280921404028?chn=ps

Table 2: Bill of Materials

Testing Procedure:

The following two testing procedures should be followed to verify the block speed in both a loaded and an unloaded environment.

A. Top Speed No Load Test

Because the block must meet the criterion of "Max speed > 0.3 m/s", this test will measure the linear velocity of the motor under no load.

a. Upload file "Motor_Test" to the arduino Uno.
b. Make sure the motor shield is connected to the arduino Uno.
c. Connect the motor to port "M3" of the shield.
d. Plug the arduino Uno into a DC power supply (PC power cannot handle the current draw of the motors).
e. Set a timer for 10 seconds.
f. As the timer counts down, count the rotations of the wheel.
g. Multiply this value by 6. This is the RPM of the motor.
h. Compute: 0.059 * RPM * 0.10472. This is your linear velocity in m/s.

PASS: This test passes if the calculated linear velocity is greater than .3 m/s.

B. Top Speed Max Current Draw Test

Because the block must meet the criterion of "Max speed > 0.3 m/s", this test will measure the linear velocity of the motor under a simulated maximum load.

a. Upload file "Motor_Test" to the arduino Uno.
b. Make sure the motor shield is connected to the arduino Uno.
c. Connect one wire of the motor to port "M3" of the shield.
d. Connect the other wire to a multimeter.
e. Connect the other end of the multimeter to the other "M3" port.
f. Plug the arduino Uno into a DC power supply (PC power cannot handle the current draw of the motors).
g. Apply increasing resistance to the wheel until your multimeter reads the max allowed current (0.45A). Continue holding at this pressure.
h. Set timer for 10 seconds.
i. As the timer counts down, continue to apply resistance as you count the rotations of the wheel.
j. Multiply this value by 6. This is the RPM of the motor.
k. Compute: 0.059 * RPM * 0.10472. This is your linear velocity in m/s.

PASS: This test passes if the calculated linear velocity is greater than .3 m/s.

If the block passes all of the listed tests, all interface properties have been verified and the block is ready for inclusion into the system.

Testing Results

A. V = 3.53V. I = 0.1A. R = 35.5Ω. RPM = 150.

Linear Velocity = 0.93m/s

B. V = 3.35V. I= 0.5A. R = 6.7Ω. RPM = 54.

Linear Velocity = 0.33m/s

Arduino Code:

```
// Written by Caleb Hubbell for ECE 342

//=============================================================
====
//     Declare Global Variables & Header Files
//=============================================================
====
```

Caleb Hubbell
5/2/2018
Team 9

```
// Including Arduino's I2C library, as well as the motorshield,
and LCD libraries.
#include <Wire.h>
#include <Adafruit_MotorShield.h>

int LSpeed = 80;
int RSpeed = 80;

// Create the motor shield object with the default I2C address
Adafruit_MotorShield motorShield = Adafruit_MotorShield();

// Select which 'port' for each motor.
Adafruit_DCMotor *leftMotor = motorShield.getMotor(3);
Adafruit_DCMotor *rightMotor = motorShield.getMotor(4);

void setup()
{
  // The motor shield and the serial monitor are initialized
  motorShield.begin();
  Serial.begin(9600);

  // Motors started
  leftMotor->run(RELEASE);
  rightMotor->run(RELEASE);
}

// the loop routine runs over and over again forever:
void loop()
{
    // go forward
    leftMotor->run(FORWARD);
    leftMotor->setSpeed(LSpeed);
    rightMotor->run(FORWARD);
    rightMotor->setSpeed(RSpeed);
}
```