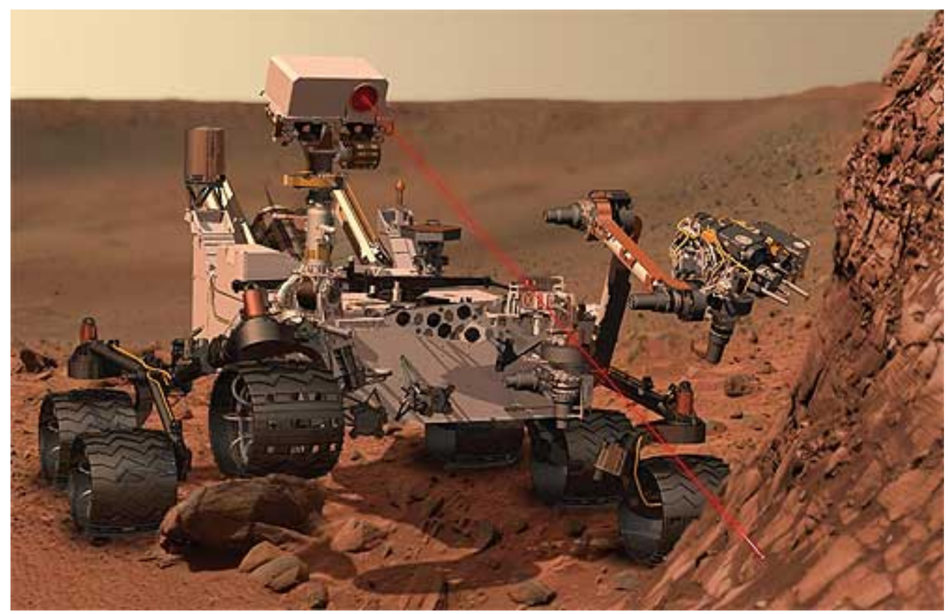
DC Motor and Pulse Width Modulation

Introduction to Mobile Robotics

DC Motor

- **Direct Current (DC) motor**
  - Converts electrical energy into rotational mechanical energy
  - Earliest form of motor and it is easy to control.
  - High torque and good speed controllability
  - Typically used in robotic manipulators and mobile robots.
  - Considered as torque generator



DC Motor

- **DC Motor Fundamentals**
  - current flowing through the motor is proportional to the generated torque

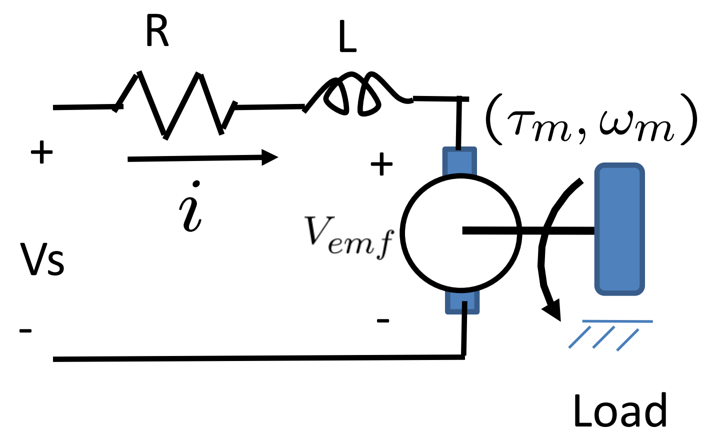
$$\tau_m = K_m i$$

  - angular rotation generates back-electromotive force

$$V_{emf} = K_v \omega_m$$

  - operation:
    - Vs is applied and current is generated which translates to torque generation

Motor Model



$$V_s = iR + L\frac{di}{dt} + V_{emf}$$

$$V_{emf} = K_v \omega_m$$

$$\tau_m = K_m i$$

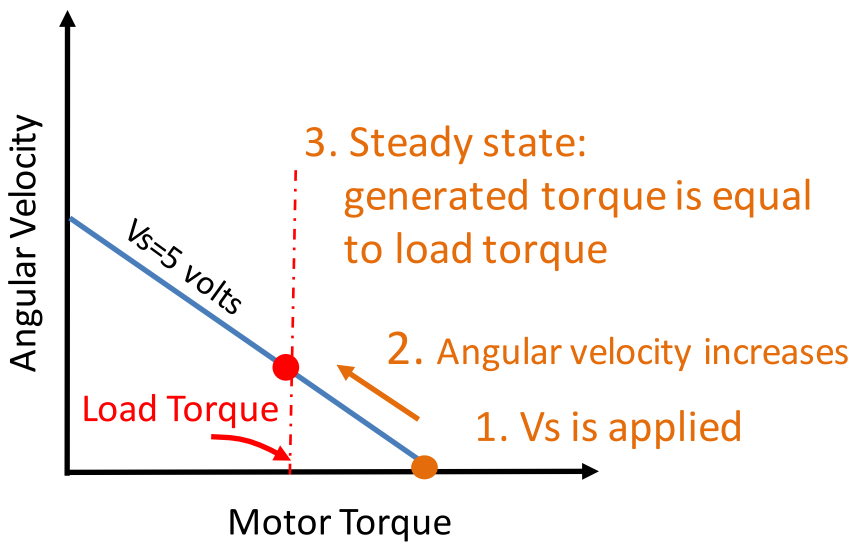
R armature resistance

L armature inductance

Kv voltage constant V/rpm

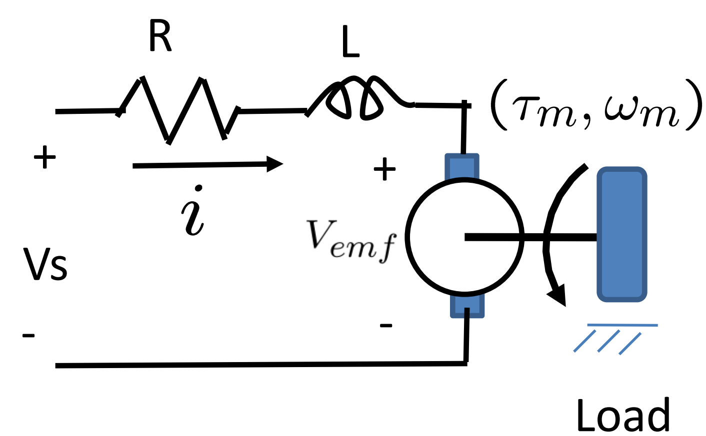
Km torque constant

DC Motor



For constant load torque, as the input voltage Vs is increased, the angular velocity ____________.

Motor Model



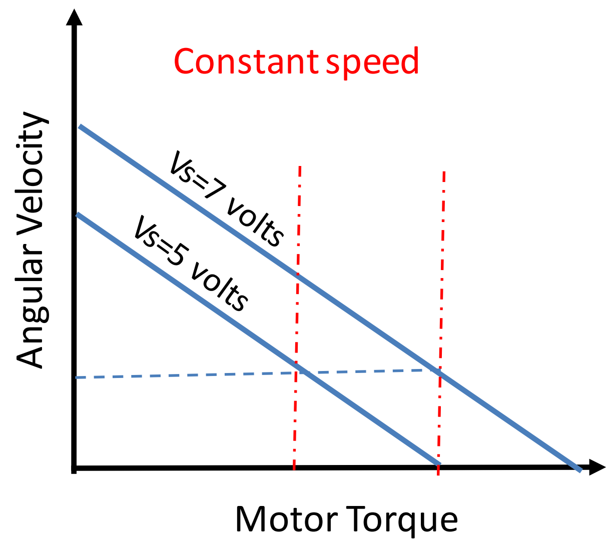
$$V_s = iR + L\frac{di}{dt} + V_{emf}$$

$$V_{emf} = K_v \omega_m$$

$$\tau_m = K_m i$$

R armature resistance
L armature inductance
Kv voltage constant V/rpm
Km torque constant

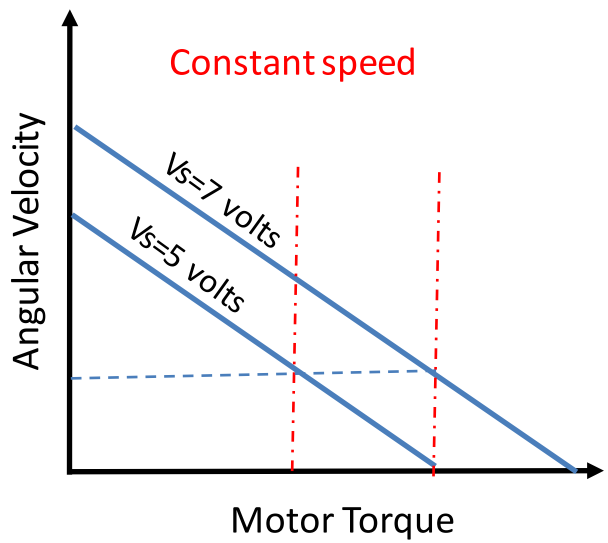
DC Motor

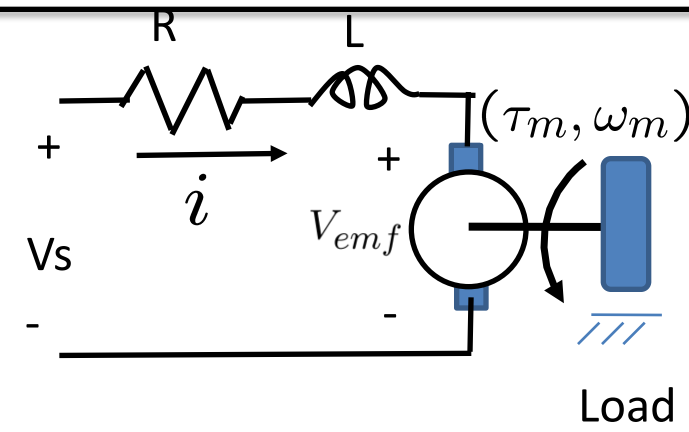


Constant angular velocity with varying load

What variable needs to be varied to achieve constant velocity?

DC Motor Control





$$V_s = iR + L\frac{di}{dt} + V_{emf}$$

$$V_{emf} = K_v \omega_m$$

$$\tau_m = K_m i$$

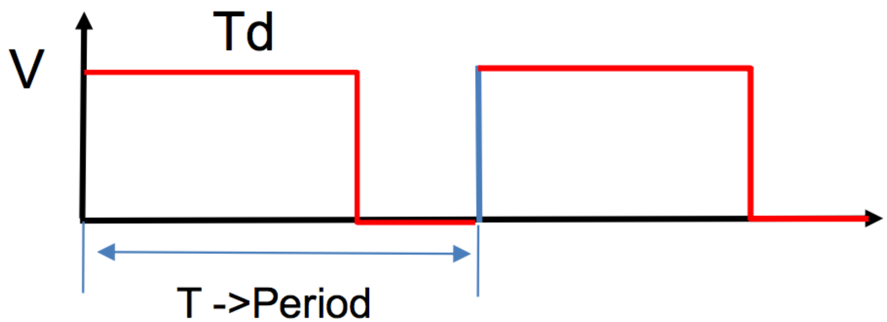
How can we generate a varying Vs from a constant voltage source (e.g., battery)?

Introduction

- **Pulse Width Modulation**
  - an efficient way of controlling motors
  - DC or average value is changed by varying the duty cycle

Introduction

- **Pulse Width Modulation**
  - Typical PWM freq. = 1/T is 100 Hz to 10 kHz.



$$V_{dc} = \frac{1}{T}\int_0^T v(t)\,dt$$

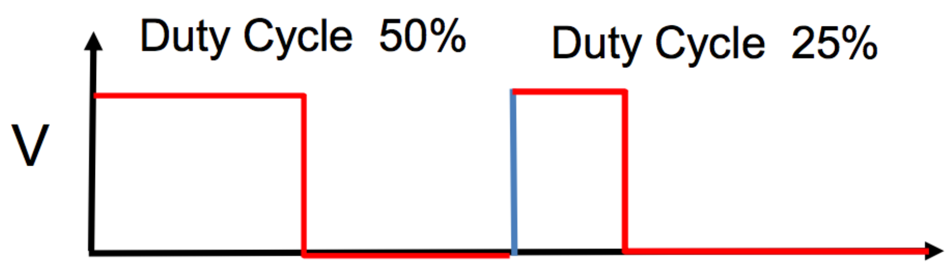
$$V_{dc} = \frac{1}{T}\,Area$$

PWM

- **Duty Cycle**

$$\text{Duty Cycle} = \frac{\text{Pulse is high (duty)}}{\text{Period}} \times 100\%$$

$$\text{Duty Cycle} = \frac{Td}{T} \times 100\%$$

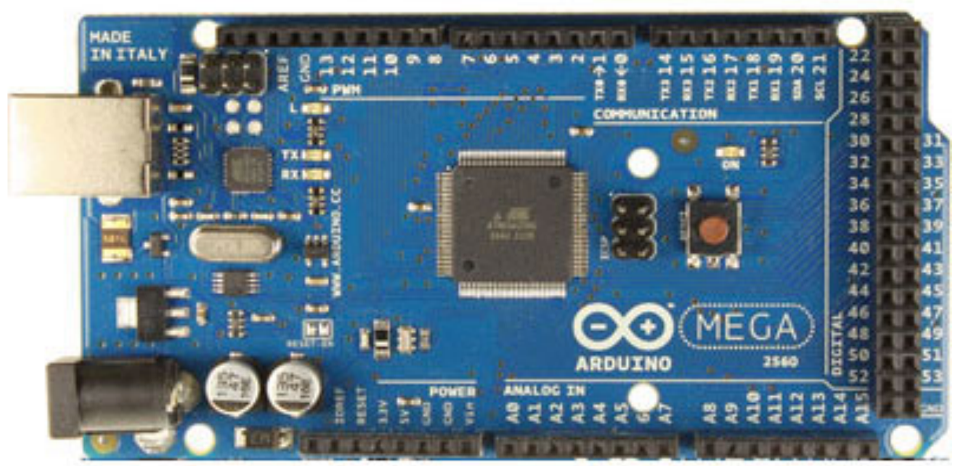


Pulse Width Modulation

- Draw a PWM signal with 75% duty cycle

PWM and Microcontroller Bit Representation

- **Computer (Microcontroller)**
  - Arduino Mega 8 bit microcontroller will be used to control the mobile base



PWM and Microcontroller Bit Representation

- **BIT or bit**
  - smallest memory unit
- **N bit system (microcontroller)**
  - Represents N bit data size, register, data bus, and address bus.
  - For example, an 8 bit system implies that the registers are 8 bit.
- **N bit PWM**
  - Implies that 0-100% duty cycle is represented by 0 – $(2^N - 1)$.
  - For example an 8 bit PWM has the mapping below

duty cycle (%)	
0	0
50	127
100	255

PWM and Bit Representation

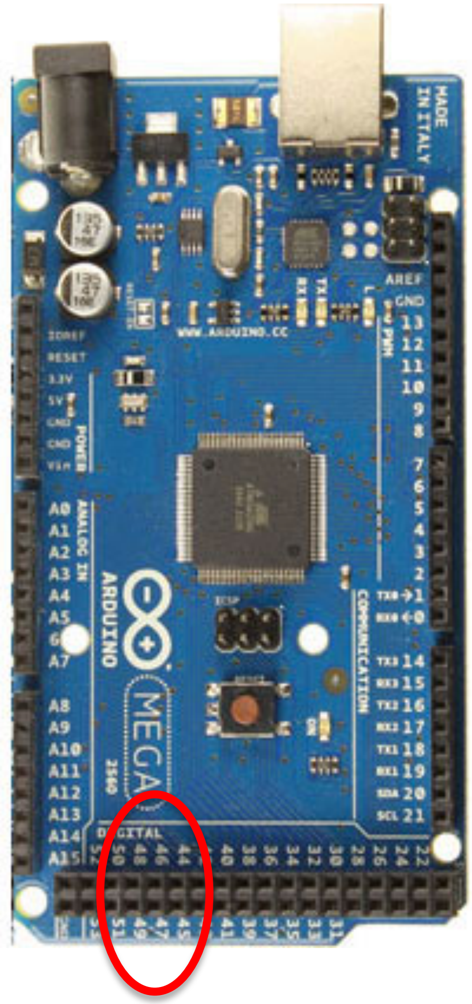
- In terms of minimum change of output voltage (output voltage resolution), what does a high number of bits imply?

PWM on ATmega2560

- **Atmega2560 (Microcontroller)**
  - Used by Arduino Mega
  - Note: Built in Arduino function for PWM has a frequency of 500 Hz, which is intended RC servo motors and other devices. Some applications require different freq. or higher frequency. Hence, we will not use Arduino built in function.

- **In the given sample code, PWM generation will be based on TIMER 5**
  - Pin 45 for motor 0
  - Pin 46 for motor 1

PWM Implementation Using Arduino Mega



Code Initialization

`pwm_init();`

```
void pwm_init(void){
    pinMode(45, OUTPUT);
    pinMode(46, OUTPUT);
    TCCR5A = _BV(COM5A1) | _BV(COM5B1) | _BV(WGM52) |
    _BV(WGM50);

    TCCR5B = _BV(CS51) | _BV(CS50);   //set prescaler to 64
    OCR5A  =  0;     OCR5B  =  0;
}
```

The above code (pwm_init) initializes timer 5 as PWM generators(2 PWMs). The PWMs are 8 bits at 500 Hz.

TCCR5A, TCCR5B, OCR5A, and OCR5B are microcontroller registers and already defined in the ARDUINO environment.

PWM Duty Cycle

- **OCR5A (Output Compare of TIMER 5)**
  - Will set the duty of channel/motor 0. The maximum value of duty is 255.

255	->	100%	duty cycle
127	->	50%	duty cycle
0	->	0%	duty cycle

PWM Duty Cycle

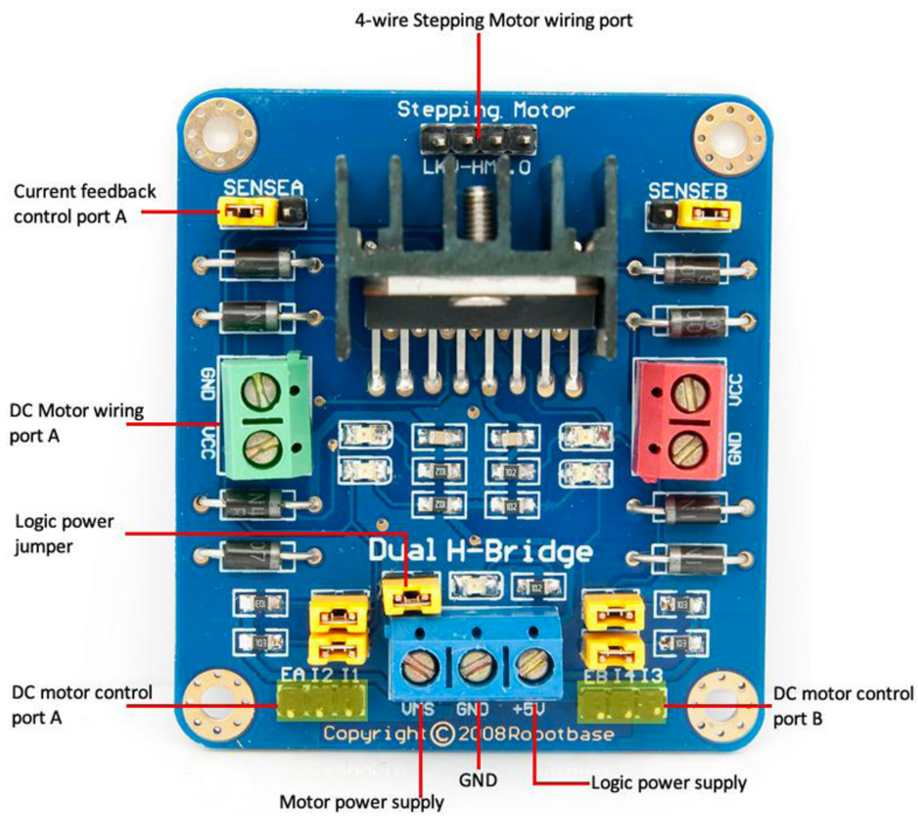
- **OCR5B**
  - Will set the duty of channel/motor 1. The maximum value of duty is 255.

255	->	100%	duty cycle
127	->	50%	duty cycle
0	->	0%	duty cycle

Motor Driver

Motor Driver

- **Current amplifier**
  - Allows low power signal from microcontroller to drive DC motors
  - EA will be connected to PWM signal for Motor 0
  - I1 and I2 are used for direction control for Motor 0
  - EB will be connected to PWM signal for Motor 1
  - I3 and I4 are used for direction control for Motor 1.



EA	I1	I2	Motor A status
》0	0	1	Clockwise rotation
》0	1	0	Anticlockwise rotation

Motor Driver

- **Motor Control Summary**
  - Aside from PWM signal, a motor needs 2 digital pins for direction.
  - For a differentially steered (2 wheels) mobile robot
    - 2 motors – 2 PWM signals and 4 direction pins

Motor Functions

```
void motor_init(void);

void set_motor_duty(int channel, int duty);
```

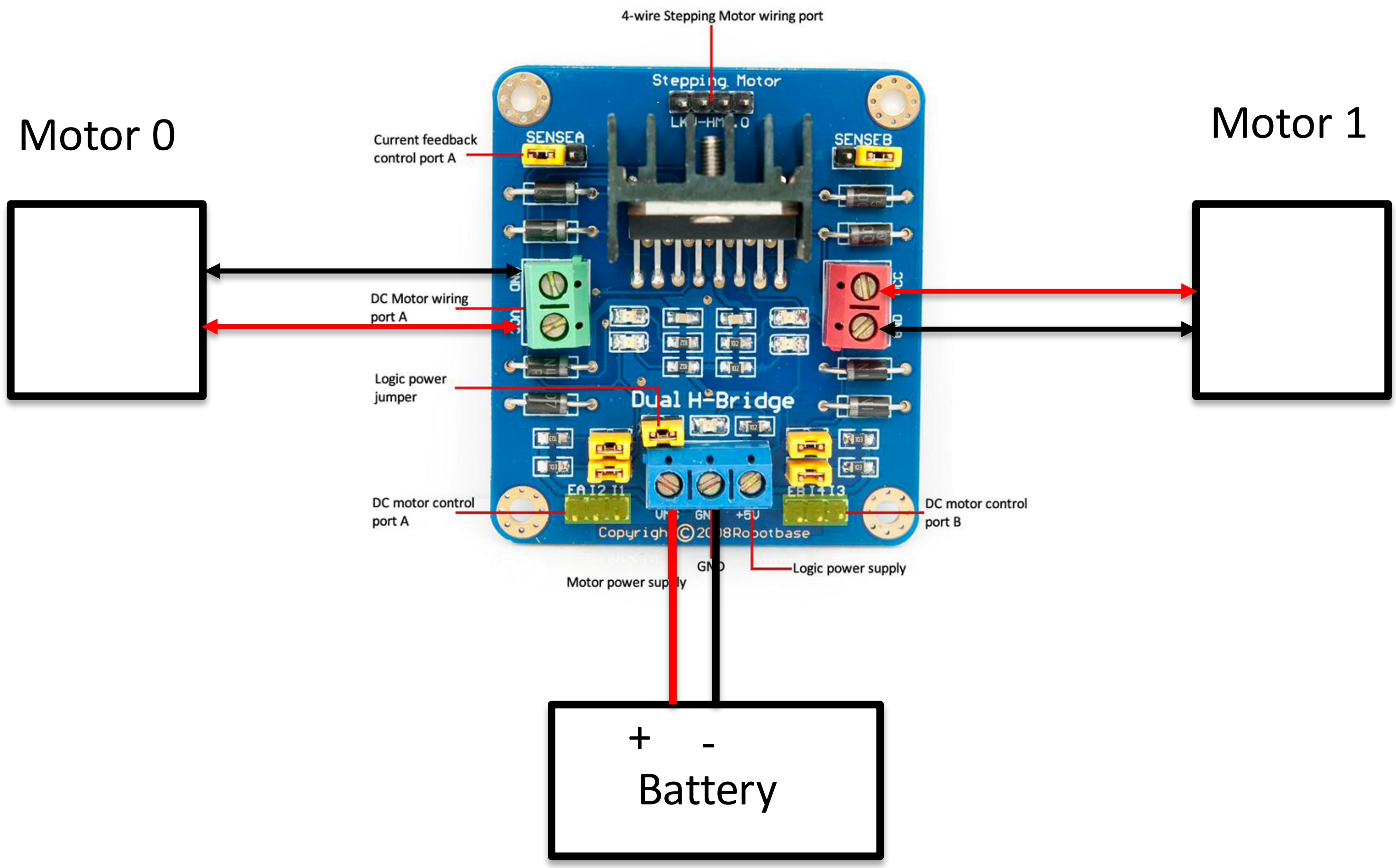
motor_init() initializes the PWM and direction pins of the motors

set_motor_speed(channel, speed)
channel = 0 implies motor 0 and channel = 1 -> motor 1
speed = 127 implies 50% duty cycle moving forward
speed = -127 implies 50% duty cycle moving backward

Trouble Shooting

set_motor_speed(0,100) should yield a forward motion. If it is backward, you can reverse the motor wiring (+ -> -) or change the direction code.

Diagram



Motor 0
Motor 1
4-wire Stepping Motor wiring port
Current feedback control port A
DC Motor wiring port A
Logic power jumper
DC motor control port A
DC motor control port B
Motor power supply
GND
Logic power supply
Stepping Motor
SENSEA
SENSEB
Dual H-Bridge
+
-
Battery

Pins

Arduino		Motor Driver
45	PWM	EA
50	Direction	I1
51	Direction	I2
46	PWM	EB
52	Direction	I3
53	Direction	I4
GND		GND
+5V		+5V

Sample Code

```
#include "mrobot.h”
void setup()
{
   motor_init();
}
void loop() {
   set_motor_speed(0,100);
   set_motor_speed(1,100);
}
```

Summary

- The provided motor functions can change the voltage and thereby change the speed of the motor.

- The main drawback is that the implementation is open loop, meaning for a constant duty or voltage, the speed will change if the load varies.