## Introduction

In the presentation, you would have learned how a DC motor and motor control circuit work. In this activity, you will develop a circuit capable of controlling the speed of a DC motor.

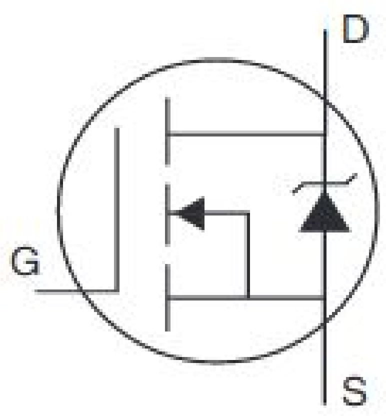
## Objective

We will be assembling a one-directional motor control circuit. A single transistor will control whether the motor is turned on or off and we will use our Arduino to control that transistor. Varying a speed value in the code will cause a Pulse Width Modulation signal to change its duty cycle, changing the motor's speed of rotation.

## Background

### MOSFETs

The primary component to be used in this lab is a Metal-Oxide Semiconductor Field-effect Transistor (MOSFET). In particular, it is the IRF530n.



MOSFETs are devices created by junctions of metal and a semiconducting material (typically Silicon). As seen above, a MOSFET is a three-terminal device, with a Source, Drain, and Gate. A key property of a MOSFET is its ability to control the current flow from the Drain to the Source based on the voltage applied to the Gate terminal. Many MOSFET circuits are designed to operate in an "Ohmic" mode, such that the MOSFET actually acts as a resistor that can vary between a value as small as 90 milliohms, or in the ballpark of Megaohms.

We can take advantage of this property by using the MOSFET as an electronically controllable, high-speed switch. The MOSFET can be considered "on" if its Gate-Source Voltage is above a certain “Threshold Voltage,” usually specified in a datasheet.  Then, it acts as a short circuit (where current travels with little resistance) between the Source and Drain.  Alternatively, if the Gate-Source voltage is low, it acts as an infinitely large resistor, and can be treated as an open circuit (no current flowing) between Source and Drain.  By controlling the voltage at the gate pin, we can control the flow of current between the MOSFET’s source and drain terminals.

Two types of MOSFET exist:  n-type and p-type.  N-type MOSFETs are capable of sinking current (can only be used to connect a node to a lower voltage), whereas P-type MOSFETs are capable of sourcing current (can only be used to connect a node to a higher voltage).  In this lab we will be using an N-type MOSFET as a switch to connect one leg of the motor to ground, thus completing the circuit and allowing current to turn the motor.

## DC Motor Circuits

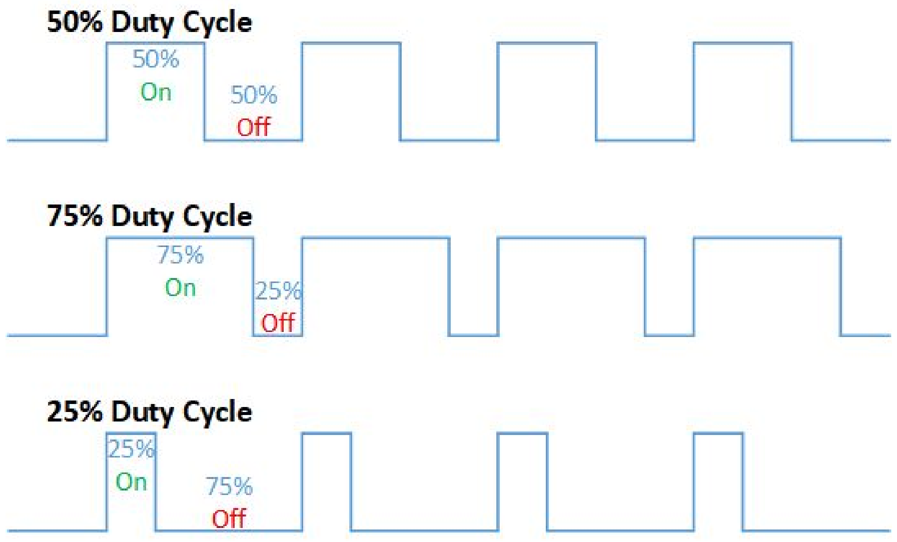
In this lab, a MOSFET will be used to control the speed of a Brushed DC motor.  DC motors are electromechanical devices that convert electrical energy (current flow) into rotation by means of a magnetic field.

Applying a potential across the two terminals excites a coil in the motor’s rotor (the part of the motor that moves), causing it to generate a magnetic field.  This magnetic field will want to align with a magnetic field applied externally by means of a permanent magnet or other coil attached to the stator (the part of the motor that is stationary).  This results in a torque that moves the rotor.  The motor continues to rotate by means of a commutator, which changes the polarity of the magnetic field as the motor rotates.  This ensures that the rotor and stator fields will never get stuck aligned, and a magnetic field will always exert a torque on the motor’s rotor.

The speed of the DC motor is controlled by the voltage applied across its terminals, while the torque applied by the motor’s rotor is controlled by the current flowing through the motor.  The actual dynamics of the motor are slightly complicated, but these simple relationships are sufficient for the application of this circuit.

## Pulse Width Modulation

Pulse Width Modulation (PWM) is a process by which one can control the average power of a signal. This is done by quickly activating a switch between the power and load, turning it on and off at a high rate.
PWM signals have a “duty cycle,” representing what percentage of the signal’s period is spent with the signal at its high value. Varying the duty cycle varies the average power. For example, a 50% duty cycle would appear to only be at half voltage, while a 100% duty cycle would deliver the full voltage.

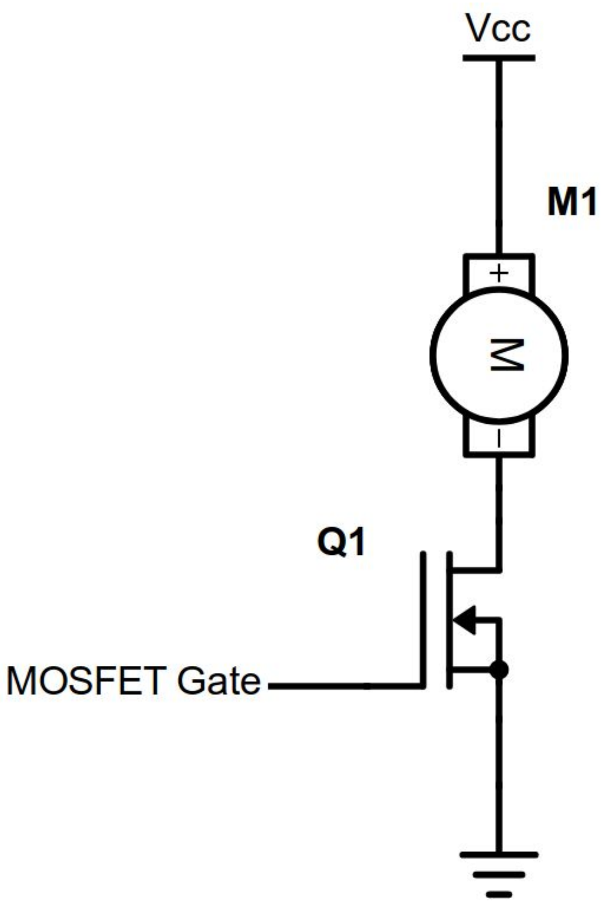


This process is useful in motor control because by pulsing the motor at various duty cycles, we can cause it to run at reduced speed. The motor rotation remains smooth so long as the pulse frequency is high, because the inductance of the motor tends to cause it to respond to the average of the signal rather than its instantaneous value.

On the Arduino, a PWM Signal can be generated on any of the ~ marked pins using the function analogWrite(). This PWM signal is generated off an 8 bit timer, so the value of $t_{high}$ is expressed as a number between 0 and 255 $(2^8)$. The duty cycle is then $t_{high}/255$.

## Instructions

1. Assemble the circuit shown below on a breadboard. The positive motor lead is connected to our 5V Supply, and the negative motor lead is connected to the NFET, which will connect to ground when the correct gate signal is applied. Pay attention to the NFET's pinouts as shown in the datasheet here (found on page 8).



2. Connect the MOSFET gate to a 5V source using a jumper wire. Verify that the motor starts rotating.
3. Unplug the gate of the MOSFET from the 5V source.
4. Connect the MOSFET gate to a PWM output of the microcontroller. These can be identified by a (~) marking next to the pin.
5. Complete the following tasks (Ranked in order of difficulty):
    a. Make the motor rotate at 50% speed by applying a 50% duty cycle signal.
    b. Repeat above for a variety of speeds (25%, 50%, 75%, 100%). Note the performance of the motor at each speed.

c. Have the motor rotate at two different speeds (selected by the position of the switch)
d. Have the motor sweep speeds, starting from the minimum speed and increasing to the maximum, before decreasing back to the minimum.

## Code Functions

- analogWrite(pin, value)
    - Outputs an approximated analog voltage through Pulse Width Modulation
        - In this lab, we take advantage of the fact that it is a pulsing approximation. Writing a true analog voltage is not desired in this lab as it leads to more power loss across the transistor.
    - Arduino Website Documentation
- digitalRead(pin)
    - Returns true if the selected pin is high, or false otherwise.
        - Used to read the position of the switch