HERIOT - WATT UNIVERSITY

School of Engineering & Physical Sciences

Mechanical Engineering Laboratory

Arduino Experiments

Pulse Width Modulation: DC Motor Control

Introduction:

Motors are used for numerous things in our daily lives, and the Arduino can control them. Here we'll use pulse-width modulation (PWM) to vary the speed of a motor.

PWM

Microcontrollers like the Arduino are often used to control devices such as motors and light sources. Often this just involves switching these devices on and off, but in some instances they are used to control the speed of a motor or the intensity of a light source.

If we take controlling the speed of a motor as an example, then a commonly used method of controlling the motor speed involves keeping the motor continuously on, but varying the power that drives it. Arduino does not have a true analogue voltage output, however by adding a digital-to-analogue converter we can then then use the analogue signal to vary the power.

An alternative mode of controlling the speed of the motor is Pulse Width Modulation (PWM). Essentially it works by only involving “on” and “off” states. PWM varies the amount of time that the blinking pin or a motor spends HIGH vs. the time it spends LOW. If it spends most of its time LOW, the LED will look dim. Because the pin is blinking much faster than your eye can detect, the Arduino creates the illusion of a “true” analogue output.

The ratio of “on” time to the total time is called the duty cycle, and variation of the duty cycle can be used to control the motor speed.

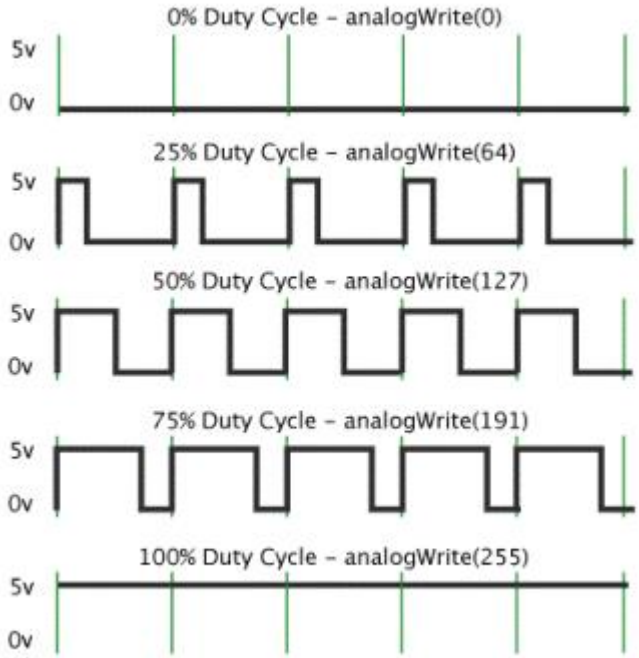

Figure 1: Pulse Width Modulation

Required Components:

- Arduino UNO
- Potentiometer
- Transistor (P2N2222A or similar)
- Diode (1N4148 or similar)
- DC motor
- 330Ω resistor
- Breadboard
- Jumper cables

Procedure:

The Arduino pins are strong enough to light small LEDs (up to 40 milliAmps), but they're not strong enough to run motors and other power-hungry parts. (This motor needs 50-100mA). Because the motor needs more current than an Arduino pin can provide, we'll use a transistor to do the heavy lifting. A transistor is a solid-state switch. When we give it a small amount of current, it can switch a much larger current.

You can turn a transistor on and off using the digitalWrite() function, but you can also use the analogWrite() function to vary the speed of the motor. The analogWrite() function pulses a pin, varying the width of the pulse from 0% to 100%. This is the same technique used in the RGB LED example for "Pulse-Width Modulation".

One thing to keep in mind is that when you lower the speed of a motor using PWM, you're also reducing the torque (strength) of the motor. For PWM values below 50 or so, the motor won't have enough torque to start spinning. It will start spinning when you raise the speed a bit.

Flyback diode

When the motor is spinning and suddenly turned off, the magnetic field inside it collapses, generating a voltage spike. This can damage the transistor. To prevent this, we use a "flyback diode", which diverts the voltage spike away from the transistor.

Connect the side of the diode with the band (cathode) to 5V. Connect the other side of the diode (anode) to the black wire on the motor.

Task 1:

Turn the motor on and off using the digitalWrite(); command.

Task 2:

Try slowly accelerating the motor to full speed, then back down to zero.

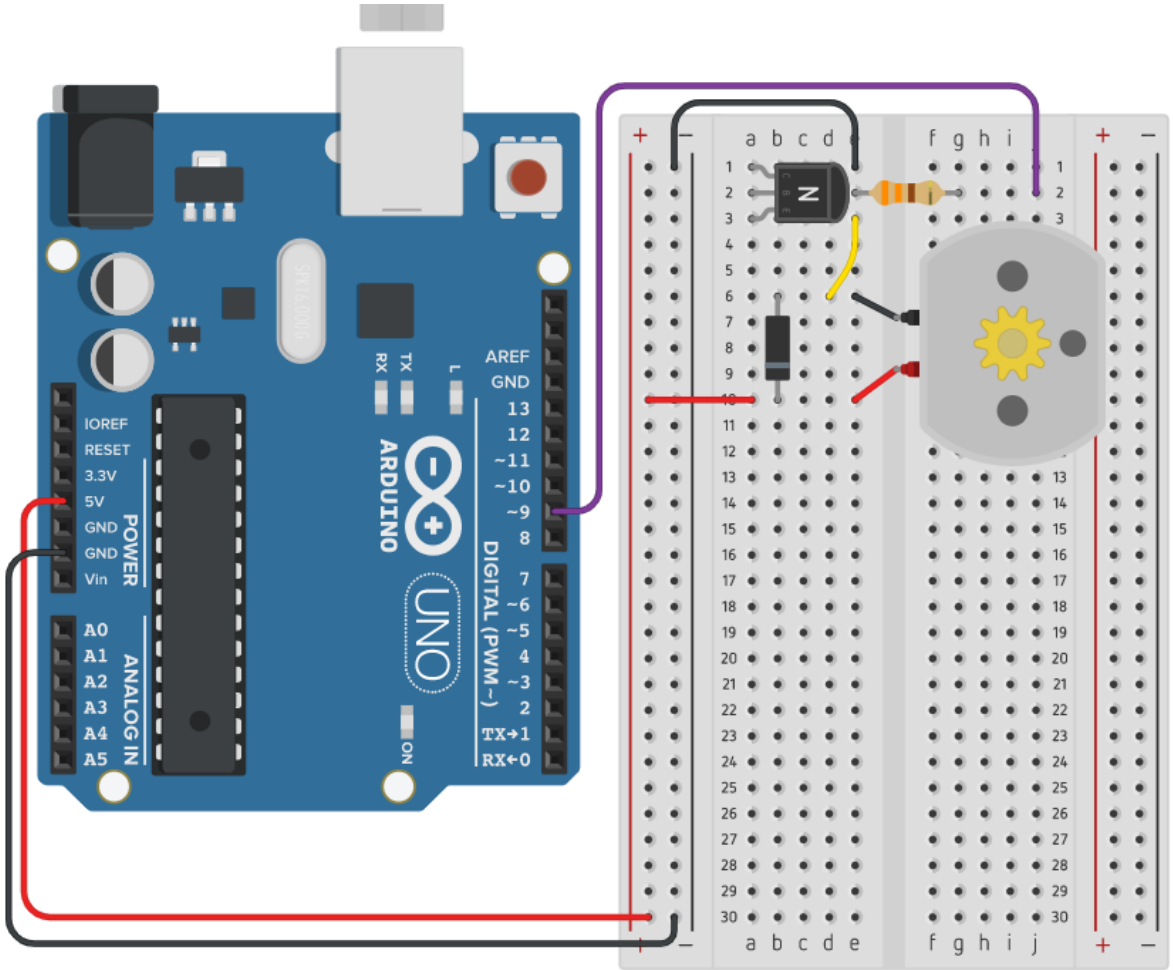

Figure 2: Circuit for Tasks 1 and 2

Task 3:

Using a DC motor driver and power supply, create a circuit and program that can read the position of a potentiometer and based on this adjust the speed of a motor.

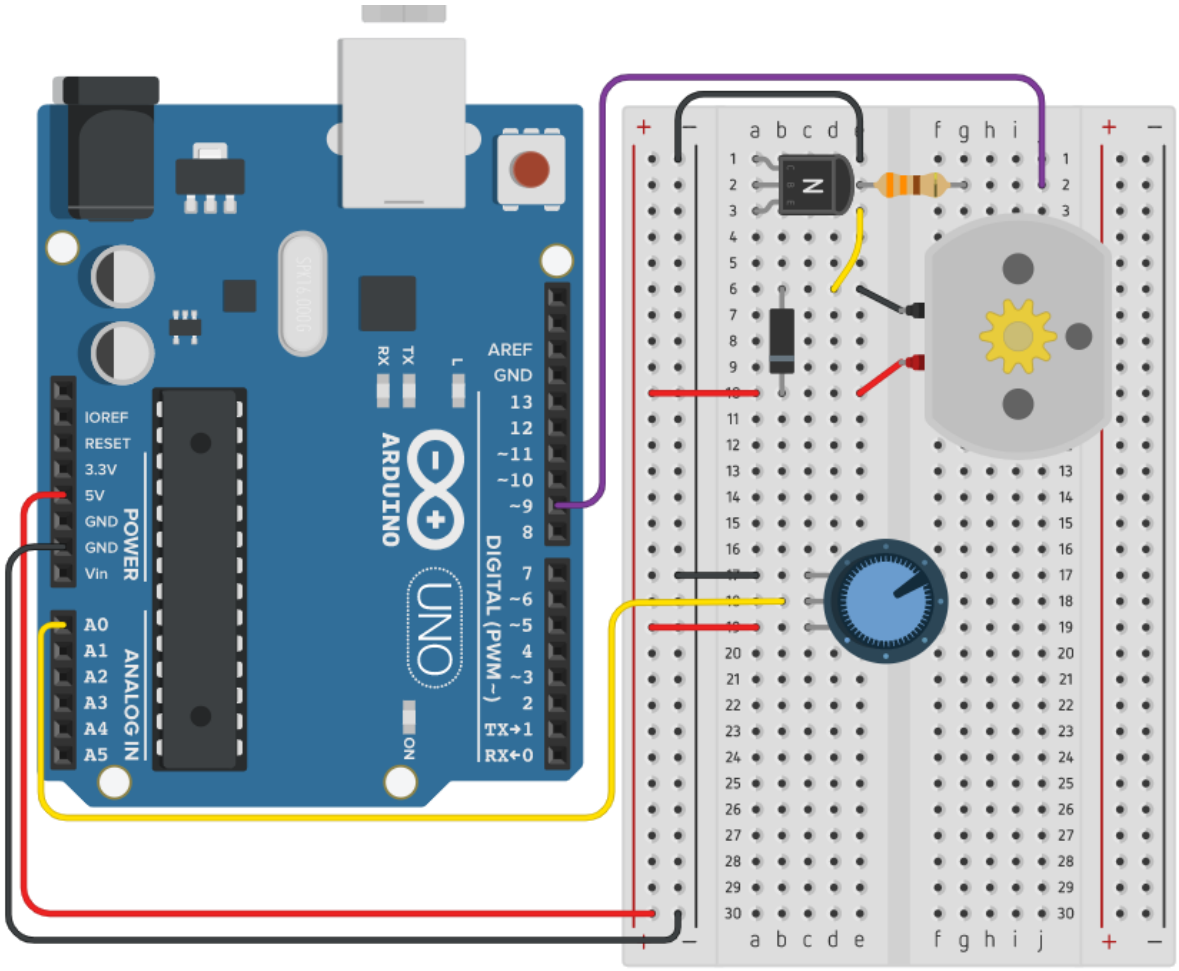

Figure 3: Circuit for Task 3