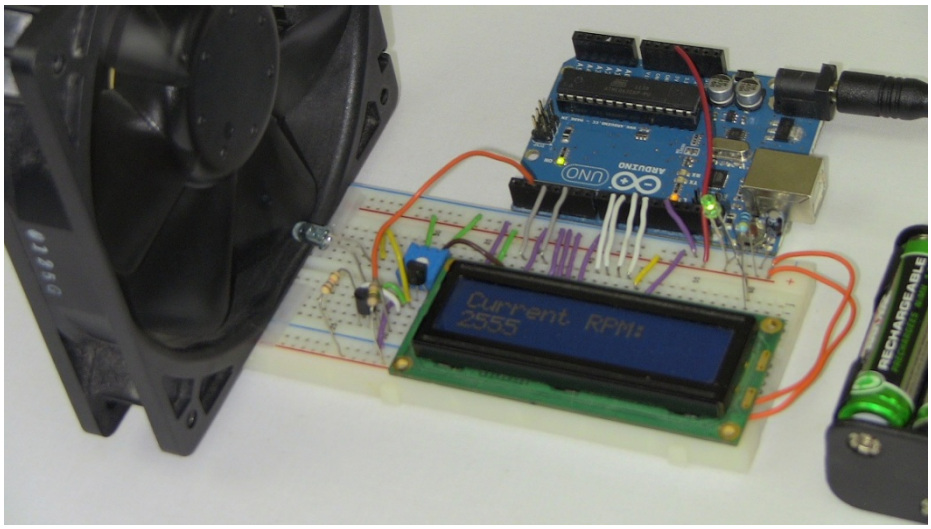
CENG314 EMBEDDED COMPUTER SYSTEMS

LABORATORY WORK 3

INTRODUCTION

Pulse Width Modulation (or PWM) is a technique for controlling power. We use it here to control the amount of power going to the motor and hence how fast it spins.



In this lab session, we will control the rotation speed of a fan via PWM. In addition, we will observe, on an LCD screen, that how the fan speed changes as we apply different duty cycle PWM signals.

MATERIALS

1x Arduino board
1x Breadboard
1x HD44780 compatible 2x16 character LCD
1x infrared led (transmitter)
1x infrared phototransistor (receiver/detector)
1x 100k ohm resistor
1x 330 ohm resistor
1x10k ohm potentiometer
1x TIP 121 transistor
1x 1N4001 diode
1x 2.7k resistor
1x 12V computer cooling fan (80mm)

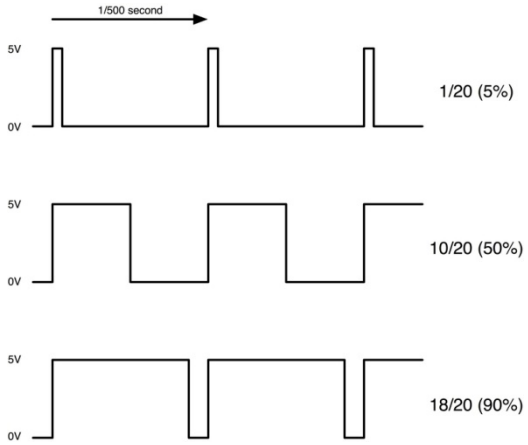
PRE-LAB

Please bring your ideas or solutions on the matters below in the beginning on the lab session in written form.
1) Discuss why would we need a driver circuit to power the fan, instead of directly connecting it to Arduino.
2) What kind of relationship do you expect to be between the duty cycle of PWM signal and the fan RPM? Linear?

LABORATORY WORK

Pulse-width modulation is a digital technique for varying the amount of power delivered to an electronic component. By adjusting the amount of power delivered to a motor or LED, the speed or brightness (respectively) can be controlled.

A microcontroller-based PWM solution uses fewer components and has the flexibility of varying the duty cycle and frequency on-the-fly through software. We use it here to control the amount of power going to the motor and hence how fast it spins. The diagram below shows the signal from the PWM pin of Arduino.

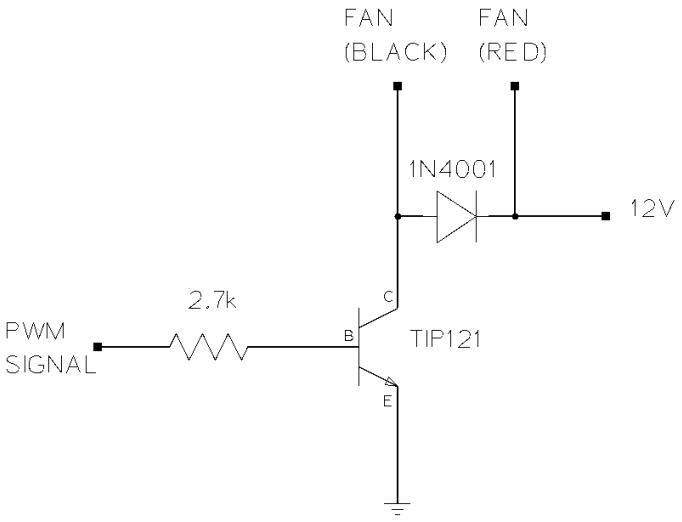


Every 1/500 of a second, the PWM output will produce a pulse. The length of this pulse controls the amount of energy that the motor gets. No pulse at all and the motor will not turn, a short pulse and it will turn slowly. If the pulse is active for half the time, then the motor will receive half the power it would if the pulse stayed high until the next pulse came along.

Pulse-width modulation is difficult below a certain duty cycle for motors because they don't gain the same rotational inertia in comparison to the static resistances of the grease, gearing, and gaps between commutators.

On an Arduino Uno, PWM output is possible on digital I/O pins 3, 5, 6, 9, 10 and 11. On these pins the analogWrite function is used to set the duty cycle of a PWM pulse train that operates at approximately 500 Hz.

We will connect the motor to Arduino using a motor driver circuit.



Follow the steps below to complete your goal for this lab session.

1. Setup your circuit by making necessary additions to last week's lab work . Make all the connections carefully.
2. Check and make sure that you are driving the fan through TIP121, not from the Arduino directly.
3. Check again that you are not driving the fan directly through Arduino. This is serious, check it again and again. You can permanently damage the Arduino board, or your computer.
4. Write a program that generates a PWM signal. Display the RPM value and duty cycle on the LCD.
5. Change the duty cycle of the PWM signal starting from 10% with 10% increments and note the fan RPM for each value. Observe whether duty cycle-RPM relationship is what you have predicted in the pre-lab. Keep your data for later. You will be including a duty cycle/RPM graph in your final report.
6. Demonstrate your sensor output signal, Arduino sketch, and running circuit to the lab assistant.