Introduction to Arduino II

Last week you downloaded the Arduino application software and learned about the general concepts of the hardware of Arduino Uno R3 as shown below.

Last week you tried the simple “Blink” sketch. Today we will look into a more advanced example in which a more complicated circuit must be used. As a reminder, the basic programming concepts are copied below.

Comments: In the comments section you describe what the code is supposed to do.
Libraries: You need to select which library folder you intend to use.
Variable Declarations: Specify the variables you intend to use and their initial values.
Setup: Here you define how the Arduino will be used and define/setup pins and communications. The word void appears before Setup. The code is written after the open curly bracket { and it is always concluded by closing the bracket }.
Loop: This is where you place the main instructions for your code. The loop is executed after setup. Any code between the curly brackets { } is processed sequentially and then repeated again for as many times as the user wishes.
User-defined functions: These are created by the user who wishes them to perform certain specific tasks.

Note that after the code is uploaded into Arduino, it may not be retrieved. Therefore, it is important to be organized and keep a record of your codes and sketches.
Motor Sketch

This is an interesting sketch that you will use to become familiar with applications that require a large current required by motors, speakers and similar devices. This sketch is similar to the Blink sketch, except here we use a motor instead of an LED and we use transistor switch to help us deliver a large current to the motor. You can obtain the “Motor” sketch from the following site, which is the same as parts of chapter 8 of “Arduino for Dummies” regarding a DC motor. The book also has chapters on stepper motors which we will not discuss here.


There is one important difference between this sketch and the Blink sketch that you did last week. Here you will use a transistor as a switch so you can deliver the necessary current to the DC motor. Note that the Arduino board only delivers a limited amount of output current, which is sufficient for turning on an LED as you saw last week.

In this experiment, you will use the small output current of Arduino to the base of a 2N2222 transistor to switch it “on”. When switched on, this transistor can deliver the necessary large current from the collector to the emitter of the transistor. This current will turn on the motor. You can also manipulate the code for this sketch to drive the motor at different speeds or ramp the speed up or down. Note that a rectifying diode is placed in parallel to the motor. This diode is placed there to protect the motor against a build-up of reverse voltage. It is important to place the diode in the circuit with the right polarity. Inserting the diode with the wrong polarity will put your Arduino and computer at risk. **Check your circuit carefully before powering it up!**

In the figure below pin 9 has been chosen as the analog output of the Arduino board, as indicated by PWM (pulse width modulation). Six PWM output pins have been indicated on this board. Any one of them could have been used for this purpose.

*You may need to use a smaller resistor than 2.2 k for motors that are not very “sticky” or require larger current!
The simple idea behind this sketch is to turn the motor on and off repeatedly. According to the code below we have chosen 1000 ms (or 1 s) for the period of on and off times. Build the circuit using the components in your tool box. Connect your Arduino board and your circuit to your laptop. Type the code below. Practice writing comments which include some points that you wish to be remembered about this sketch. Compile the code and make sure there are no error messages. Some common mistakes involve syntax errors which include typos, semicolons, and case sensitivity. Next, upload the sketch and observe the motor running and going through cycle of off and on. Save the sketch by giving it a name of your choice.

First we declare digital pin 9, and then we define pin 9 to be an output. In the loop command, we are instructing the board to turn the output high for 1000 ms and low for 1000 ms.

```cpp
int motorPin = 9;
void setup() {
  pinMode(motorPin, OUTPUT);
}
void loop() {
  digitalWrite(motorPin, HIGH);
  delay(1000);
  digitalWrite(motorPin, LOW);
  delay(1000);
}
```

Troubleshooting

In the event that the motor does not respond as expected, review the following:
1. Are you using an npn transistor? pnp does not work!
2. Are you correctly inserting the transistor in the circuit according to its pin diagram (CBE)?
3. Is the polarity of your power connection correct?
4. Is the diode inserted correctly in the circuit?
5. Are you using the correct Arduino pins?
6. Finally, your motor may require larger amount of current to start spinning. Reduce the value of the resistor connected to the base of the transistor (as low as 200 Ω). A larger base current will result in a larger collector current.
7. In the event that you are using a large motor, use an external power supply capable of providing large current. You may have to crank up the power supply voltage a little beyond 5 volts. Note that your transistor should be able to handle the current needed by the motor. Check your transistor’s specifications.

You can easily change the on and off times and observe the motor turning on and off accordingly. For a variation of the above sketch visit the Arduino site https://www.arduino.cc/en/Tutorial/TransistorMotorControl.

Changing Motor Speed

Suppose you wish to change the speed of the motor by gradually ramping up and down. In the previous off and on case you switched back and forth the output voltage of pin 9 between 0 and 5 V. Here you increase the output voltage gradually from 0 to 5 in predetermined increments. You would then gradually decrease the output voltage from 5 back to 0. The 0 – 5 V interval is divided into 256 segments ($2^8 = 256$ for an 8 bit processor). Starting with 0, the highest value
would be 255. We choose an incremental increase or decrease of 5. Therefore, the motor will experience 51 steps of incremental voltage increase and decrease. Here is the code:

```cpp
int motorPin = 9;

void setup() {
  pinMode(motorPin, OUTPUT);
}

void loop() {
  for(int motorValue = 0, motorValue <= 255; motorValue +=5) {
    analogWrite(motorPin, motorValue);
    delay(30);
  }
}
```

Press the Compile button after typing the code and saving the sketch. If compiled correctly, upload the sketch. You should be able to observe the motor to slowly ramp up its speed to the maximum value and then ramp down to zero. It is normal for the motor to hum at lower speeds.

**Controlling Motor Speed**

You can make this sketch a little more exciting by adding a potentiometer (variable resistor) to your circuit so you can select the speed of the motor by dialing the knob of your potentiometer. **In order to do this you could connect a 10 k potentiometer between the 5 V power pin and the ground (GND).** The rest of the circuit would be the same as before. This potentiometer draws some current away from the motor. The current through the potentiometer would be larger for lower resistance values. Therefore, the motor runs slower for smaller potentiometer resistance values. Recall that you will need to use the middle pin and one of the outside pins of the potentiometer. You can connect the potentiometer and observe the corresponding change of

You can display the potentiometer value using the serial monitor button at the top right of your Arduino IDE window. However, you must include some additional lines of code to the sketch if you wish to observe the serial monitor function. Learning the code for printing values in the serial monitor window will be quite useful for many interesting experiments. One example would be to monitor the analog sensor values (temperature, light, pressure, etc.).

For a detailed description of the codes for serial monitoring of the potentiometer values, see page 134 of the book “Arduino for Dummies”), or search online for similar sketches. Finally, note that the above sketches are for DC motors which simply spin at speeds proportional to the current that flows through them. There are, however, more complicated motors that you might want to learn about. On page 136 of the book, you will find an introduction to servo motors and the codes for a servo motor sketch. I believe that the kits you purchased at the beginning of the semester should include a servo motor. Enjoy experimenting with Arduino!