Lab 1 - Getting Started with Arduino

ME 351 - Introduction to Instrumentation and Measurement Systems, Winter 2022
Lab Days: 2

What we'll learn:

- How to write and upload simple programs to an Arduino-style board.
- How to use these core Arduino functions:
  `digitalRead()`, `digitalWrite()`
- How to put together a simple circuit.

Tips & Tricks

Carefully read the lab. Skipping ahead and trying to piece things together in post is a good way to give yourself a headache. *If you ask a question that is already covered clearly in the lab, we will tell you to return to the document and read it.*

Some quick wording explanations:

- **Arduino**: a whole ecosystem of electronics that is open-source. This includes physical board designs, a programming environment, and many other resources.
- **Arduino Uno**: one of the most popular Arduino development boards. Since it is open-source, brands other than the official Arduino can make and sell them.
- **Arduino IDE**: the Arduino integrated development environment. This is an application on your computer that you can use to program and interact with Arduino boards.

Part 1: Connecting to the Arduino

Tasks

1. Download and install the Arduino IDE.
   a. Go to the Downloads section of the official Arduino website (arduino.cc). (arduino.cc/en/software)
   b. Download and install the appropriate version.
      i. On a **personal computer**, using the Windows, Mac, or Linux installers is easiest.

ii. On a **lab computer**, choose “Windows ZIP file for non admin install”. Then, right-click on the zip file and unzip it. From here, you can start the Arduino software by clicking on the arduino.exe file inside the unzipped folder.

c. Start the Arduino IDE.

2. Try uploading a simple example program, Blink, to your Arduino board. If you need additional guidance, this guide may help you: digikey.com/en/maker/blogs/2018/how-to-get-started-with-arduino.

   a. Connect your Arduino UNO to your computer with the USB cable. The Arduino is powered by the USB cable as well as connected for serial communication. You do not need to use the barrel plug unless you are untethering your board.

   b. In the Arduino IDE, open [File] → [Examples] → [01.Basics] → [Blink].

   c. Select [Tools] → [Board] → [Arduino/Genuino Uno].

   d. Select [Tools] → [Port] and select the “(Arduino/Genuino Uno)” port.

      i. There are some port name possibilities, please select the one that includes “Arduino” or, if none do, try each one available.

      ii. If none are available, the old unplug-replug usually does the trick.

   e. Select the round button with an arrow to the right (upload button).

We’ve succeeded when the Arduino IDE says “Done Uploading” in the blue bar towards the bottom. You should see the orange light on the Arduino Uno blink on and off every second. **Common issues:**

- The wrong port is selected. Double-check your port or select another port option.
- The wrong board is selected. Ensure that [Tools] → [Board] is set as above.
- The USB cable isn't all the way in. The big USB-B plug can be a bit stiff.

Quick tips:

- Turn on line numbers! This helps a lot with questions and debugging. Go into [File] → [Preferences] and check the "Display line numbers" box.
- Press [Ctrl] + [R] to compile the program. This is useful to quickly check if there are bugs without uploading the program to the board, which can take more time.
- Press [Ctrl] + [U] to upload the program. The Arduino IDE will always compile before uploading, so you do not need to press compile beforehand.
- Press [Ctrl] + [T] to automatically format your code nicely. This can be very useful for reading and debugging, please press it often!

Part 2: Dabbling in Circuits

Before we put any circuits together, let's make sure we're clear on the dangers. While the voltage and current levels available from the electronic components are very low (the most dangerous thing you could do is lick the 9V battery), a misstep can still damage the electronic components. For instance, you could damage sensors, blow up LEDs, or make the Arduino into an expensive paperweight (also known as "bricking").

The main danger is "**short-circuiting**", when two points in a circuit at different voltages connect with no resistance in between. From Ohm's Law (V = IR), a nonzero voltage difference and resistance that approaches zero will lead to current that approaches infinity. In practice, this essentially means that a lot of current runs through the circuit, burning things along the way. The most common way to do this is to connect power (nonzero voltage relative to ground) and ground (zero voltage reference point).

Discussion Question 1

Take a look at the Arduino Uno. Which two pins would be “shorted” (short-circuited) if you connected them together directly? There are multiple correct answers, so please include just two pins.

Description, Continued

For this class, we’ll be using solderless breadboards and jumper wires in place of soldered or twist connections. These are very good to quickly prototype circuits, though less permanent and with some other limitations. If this is the first time you’ve used a breadboard, please take a look at this brief introduction to breadboards: wiring.org.co/learning/tutorials/breadboard/

The kit also includes jumper/header/DuPont wires: individual wires that can easily connect and disconnect components thanks to the connector on each end. The bundle contains male-to-male wires: good for bridging between the Arduino Uno pins and the breadboard, as well as bridging breadboard rows. Slide them out of the bundle to keep them from getting too messy. The wires in a neat row are male-to-female wires: good for connecting to things like sensors. You can peel them apart when you need them.

Discussion Question 2

There are many useful tutorials on the official Arduino website (arduino.cc, hover over [Documentation] and select [Tutorials]). We will use the Blink tutorial which is a built-in example. The Arduino documentation (docs) is a good place to get more details about the built-in examples (arduino.cc/en/Tutorial/BuiltInExamples/Blink).

Based on the **Blink example documentation**, what range of resistor values are acceptable for lighting an LED with the 5V power source we use in this lab? Which resistor do they suggest you use in the Blink example?

Discussion Question 3

The resistors in your kit have labels on the yellow paper that holds them together to indicate their resistor value. However, in the wild you may need to identify resistors through their color bands, so knowing where to learn to do that will be helpful. Find an online link to a resistor color chart or calculator.

Tasks

1. Set up a simple circuit to flash an **external** LED in addition to the orange, on-board LED on the Arduino Uno.
    a. Pull out some jumper wires, your breadboard, an LED and a resistor at a reasonable value, according to your answer to Discussion Question 2.
        - The example shows the resistor and LED plugged directly into the Arduino pins. **DO NOT** do this; practice using your breadboard.
    b. Closely read the "Circuit" section of the tutorial page for the Blink example and create the circuit according to the diagram and schematic provided.
    c. Be **very careful** not to connect the LED between power (non-ground voltage) and ground without a resistor. It is very easy to burn out or explode LEDs this way!

We're done with this section when the orange on-board LED and the external LED we set up on the breadboard are blinking together. **Common issues:**

- The LED is backwards. Many electrical components have polarity, LEDs are one of them. The long end should be connected to the higher voltage and the short end to the lower voltage. Try flipping it.
- The circuit is not complete - that is, there is not a connection between power and ground at some point. This connection allows current to flow. Double-check your circuit with the diagram.

Part 3: Dabbling in Code

For those who are new to programming in C or C++ (or in general), these tutorials may help you: cplusplus.com/doc/tutorial/, particularly the following sections:

- Structure of a Program
- Variables and Types
- Operators
- Control Structures

Arduino also has great reference pages for basic programming and Arduino-specific programming available on their website (arduino.cc/reference/en/).

We recommend continuing the lab until you find a word you're not familiar with.

Discussion Question 4

In the code for the Blink example, what function is called within the loop is used to turn on the built-in LED? What are the two arguments the function takes, and what do they

represent? There are two possible values for the second argument; what are they, and what voltages do they lead to for the Arduino Uno?

Tasks

1. Change your code and circuit so that the LED on the breadboard:
    a. Is connected to pin 7. This allows you to practice setting up different pins in your circuit and your code. Make sure to include a new `pinMode()` command for your new pin.
    b. Blinks on and off every half-second. This will require you changing the `delay()` at the end of the loop, which is a valuable command we will use throughout this class.
2. **Demonstrate your working system to a teacher for sign-off 1.**
    a. In lab sections or office hours, show your system to a teacher. They will record your sign-off or give you feedback on what additional steps you need to take to have an acceptable sign-off.
3. **Alternatively, make a brief video of your working system for sign-off 1.**
    a. If you cannot demonstrate your system in lab sections or office hours, you may submit a video instead. We will evaluate your video when we grade the lab report, so you will not receive feedback or confirmation that your sign-off is acceptable.
    b. Save the video as an .avi, .mp4, .mov, or .mkv file and title it "Firstname_Lastname-L1_S1", substituting in your own name.
    c. The file must be smaller than 25 MB and shorter than 30 seconds.
    d. Email the file to Lab_1_S.zc23in85jwsgr37d@u.box.com. You should receive a confirmation email that the file was uploaded successfully.
    e. **Common issues:** Your video may be too big. If so, you can adjust your phone's resolution settings or compress/trim your video on your computer. Your sign-off will not upload successfully if you try to send more than 25 MB at once, or if the file is linked via Google Drive and not actually attached to the email. If you do not get a confirmation email that the file was uploaded successfully to Box soon after sending it, it is reasonable to assume the upload failed and you will need to try again.
4. **Save a copy of your code in a .ino file for submission with your lab report.**
5. **Keep your LED circuit on the board for the following tasks.**

Part 4: Getting Inputs

So far we have demonstrated electrical outputs by flashing an LED, now let's look at electrical inputs by reading the voltage present at a pin using a button.

Tasks

1. Set up the Button example to control your **external** LED.
    a. Find the Button example documentation on the Built-In Examples page.
    b. Create the example circuit on your breadboard **separate** from your LED circuit from Part 2 and 3. **Keep your external LED circuit intact.**
    c. Open the Button example in the Arduino IDE and upload it.

Discussion Question 5

In the original Button example circuit, when the pushbutton is pressed, what will we read at the pin: `HIGH` or `LOW`? You can determine this by seeing what the resistor is connected to in your circuit (ground or power), or by looking at the logic in the code. Briefly explain your reasoning.

Tasks, Continued

2. Change your **circuit** (changes to the code will **not** count for a sign-off) so that the external LED on the breadboard:
    a. Turns **on** when the button **is not pressed**
    b. Turns **off** when the button **is pressed**
    c. **Side note**: This will require you changing what the resistor is connected to (ground or power). The original configuration in the example is a pull-down resistor, which pulls the level of the pin downwards by default (when the button is not pressed). The new configuration is called a pull-up resistor, which does the opposite.
3. **Demonstrate your working system to a teacher for sign-off 2.**
    a. In lab sections or office hours, demonstrate your system to a teacher. They will record your sign-off or give you feedback on what additional steps you need to take to have an acceptable sign-off.
4. **Alternatively, make a brief video of your working system for sign-off 2, making sure to clearly show the circuit.**
    a. If you cannot demonstrate your system in lab sections or office hours, you may submit a video instead. We will evaluate your video when we grade

the lab report, so you will not receive feedback or confirmation that your sign-off is acceptable.

   b. Save the video as an .avi, .mp4, .mov, or .mkv file and title it “Firstname_Lastname-L1_S2”, substituting in your own name.
   c. The file must be smaller than 25 MB and shorter than 30 seconds.
   d. Email the file to Lab_1_S.zc23in85jwsgr37d@u.box.com.

5. **Save a copy of your code in a .ino file for submission with your lab report.**
6. **Preserve the circuit on the board for Lab 2.**

Post-Lab Questions

1. Grounding

a. What does ground on the Arduino represent in terms of voltage?
b. In electronics, what is a common ground? When and why is it necessary?
c. Is the Arduino ground or the common ground the same as Earth ground? Why or why not?

2. Polarity

a. What is electronic polarity? What do we have to be careful about for polarized components?
b. What component(s) from this lab are polarized?

3. Digital Reading and Writing

a. Which pins on the Arduino Uno can you use `digitalRead()` and `digitalWrite()` on? Please give your answer in terms of pin labels (e.g. A0, GND, 2). **Hint:** Analog pins can also be used as digital pins.
b. When you use `digitalRead()` on the Arduino Uno, above what voltage do readings end up being evaluated as `HIGH` and below what voltage are they `LOW`? **Hint:** The answers are not 0 or 5V. In reality, there are a range of voltages that are mapped to these readings.

Lab 1 Sign-offs Page

You do not need to write anything on this page, but it is a good reference for the sign-offs you should have completed during the lab. Before you demonstrate or email your sign-off, please make sure your recording is **school-appropriate**.

If you choose to submit your sign-off via email, you will automatically receive a confirmation that the file has been uploaded. **If you do not receive confirmation, try again, and then contact the teaching team if it fails.**

Please note that with a large online class, it is extremely important to **format and name files as noted below**. If there are errors in formatting, file-naming, or file submission we will apply **large formatting deductions**. We do this to ensure we can spend most of our time teaching you, rather than searching for your responses/files/etc.

Sign-offs for Lab 1:

1. _____ Demonstrating your custom blink program.
    - Show your system to a teacher, or:
    - Submit a video of your system with the following requirements:
        - An .avi, .mp4, .mov, or .mkv file
        - Smaller than 25 MB and shorter than 30 seconds
        - Titled “Firstname_Lastname-L1_S1”, using your name
        - Email your video to **Lab_1_S.zc23in85jwsgr37d@u.box.com**

2. _____ Demonstrating your off button.
    - Show your system to a teacher, or:
    - Submit a video of your system with the follow requirements:
        - An .avi, .mp4, .mov, or .mkv file
        - Smaller than 25 MB and shorter than 30 seconds
        - Titled “Firstname_Lastname-L1_S2”, using your name
        - Email your video to **Lab_1_S.zc23in85jwsgr37d@u.box.com**