CPE 301 1104 - Lab Report 1

Lena Kemmelmeier, Lab Partner: Emma Cornia

February 15, 2024

Emma and I built our circuits alongside each other and decided to write separate lab reports.

Abstract

This lab allowed us to familiarize ourselves with the basics of Arduino programming. Here, the task was to make a simple LED light activate and create output in the Serial monitor when a button is pressed. This was our first experience with wiring with the Arduino AT Mega 2560 and the Arduino IDE and it allowed us to apply what we learned about pull-down resistors in the lecture.

1 Introduction

This lab was our first experience with coding the Arduino. We applied our understanding of the setup function, which runs when one presses reset or powers the board, and the loop function, which runs over and over again while the board is active. Moreover, we gained proficiency utilizing the Arduino *Wired* library in C++, which will serve as a conceptual basis for future programming in just ANSI C. For this lab in particular, we learned about setting the mode of pins (input/output), were introduced to the serial monitor and its baud rate, and became more familiar with Serial monitor I/O. Additionally, we integrated a pull-down circuit into our design.

2 Answers to Questions

2.1 What is a pin on the Arduino?

We set digital pins on the Arduino to be inputs or outputs using functions in Arduino programming. Analog pins read analog voltage levels. Pins allow us to connect and control different components (e.g., LEDs).

2.2 What is the Serial port in reference to the Arduino ATMega 2560?

The serial port on the Arduino ATMega 2560 is how we interface with the microcontroller/circuit and other devices. The default serial port for the Arduino ATMega 2560 is Serial, which is seen in our code.

2.3 How is analog data different from digital data?

Digital data is discrete. With the Arduino, it only has two states, high (1) or low (0). Here, it is the voltage levels that are being interpreted as either high or low. Analog data, on the other hand, is continuous. Just like with digital data, analog data also represents voltage levels with the Arduino.

3 Experimental Design

In our experimental design, pin 4 served as an output to the LED while pin 2 served as our input from the button. We used a pull-down resistor for the button: When it was pressed, our pin (and therefore our LED) changed from a low state to a high state (see Figure 1). In our code's setup function, we

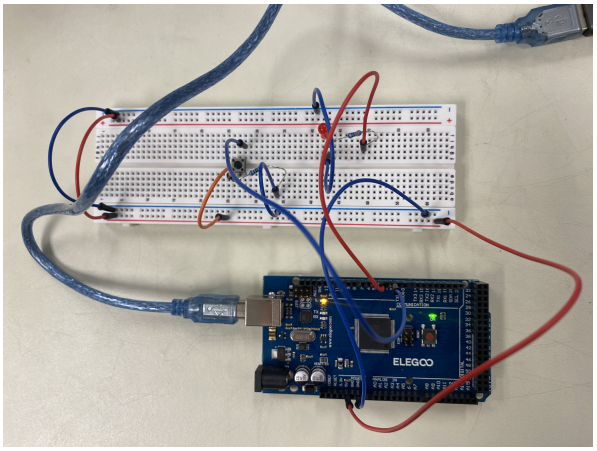

Figure 1: The breadboard set-up.

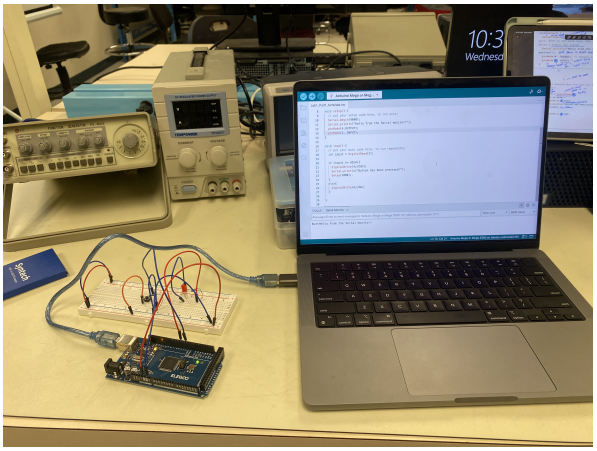

Figure 2: Results when the button was not pressed.

set the serial monitor with its baud rate (9600 bits/s), the speed at which a signal can travel. We also printed a message to the Serial monitor and established our inputs/outputs here. In the loop function, which was continually run, we checked for our button input. If the input was high (the button was pressed), the LED would become high, and another message to the Serial monitor would display, indicating the button had been pressed. This was followed by a 1000 ms delay. If the input was not high (the button had not been pressed), then the LED would become low.

4 Results

When the button was not pressed, the LED light would be inactive, and no new message would display on the Serial monitor (see Figure 2). When the button was pressed, the LED light would be active and the message, "Button has been pressed!" would display on the Serial monitor (see Figure 3).

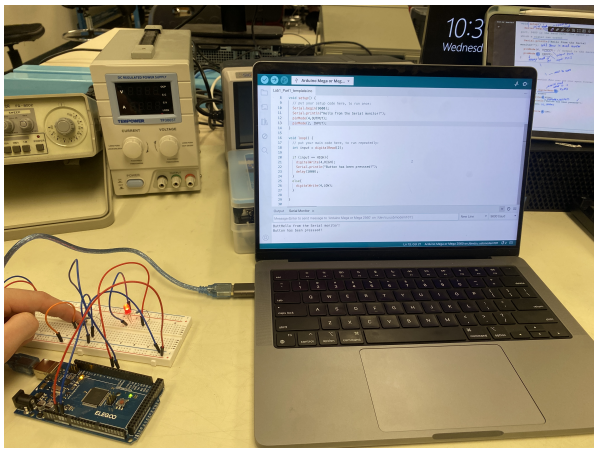

Figure 3: Results when the button was pressed