CPE 301 1104 - Lab Report 2

Lena Kemmelmeier, Lab Partner: Emma Cornia

February 22, 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 buses. Specifically, we applied our knowledge of decoders and memory management. Since this lab was more complicated than the previous one, it also allowed us more experience with debugging.

1 Introduction

In this lab, we reviewed the concept of decoders and memory-mapped IO. Decoders map n input lines to a maximum of 2^n outputs, where only one output line is 1. RAM chips are connected to address and data buses. The two highest address bits are used to select the memory chips via the decoder, and each output pin of the decoder can select one RAM chip. Each RAM chip is mapped to a range of memory. Basically, we use a chip select by segmenting the address bus, which forms a control bus that shares the address space. In this lab, we mimicked this by creating a makeshift chip select decoding circuit on the Arduino.

2 Answers to Questions

2.1 What is a Data bus, an Address bus, and a control bus? How do they differ?

A bus is a set of conductive lines. A data bus carries data either to or from external systems. An address bus carries the address to be recalled or written. A control bus carries signals like the clock and read/write line. Each has a distinct purpose by carrying different things.

2.2 Why is a decoder needed in most peripheral device configurations?

A decoder is needed in most peripheral device configurations because when we have multiple devices it can select the correct address for each device for communication. This is beneficial because it allows us to use multiple peripheral devices without extra data and control lines, creating a more manageable system.

3 Experimental Design

We used an Arduino Atmega 2560 connected to a computer with the Arduino IDE. Altogether, we utilized a solderless breadboard, two DIP Push buttons, four LEDs, four 330 ohm SIP Resistors (for LEDs), and two 2K ohm SIP Resistors (for buttons), and a jumper kit.

Our breadboard setup can be seen in Figure 1. In our experimental design, Pin 3 (our right button) served as the input for the LSB. Pin 4 (our left button) served as the input for the MSB. Our outputs were pins 5-8, all connected on their own LEDs. Pin 5 represents D0, Pin 6 D1, Pin 7 D2, and Pin 8 D3.

We verified our inputs and outputs with the decoder truth table provided in the lab description. Here, Line 1 was represented by our right button (button 1), and Line 2 by our left button (button 2).

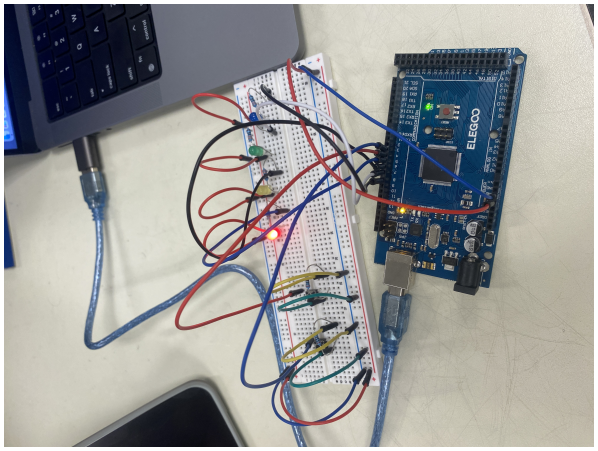

Figure 1: The breadboard set-up.

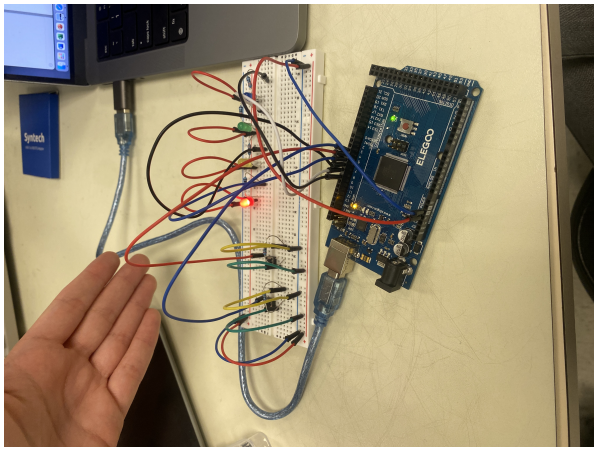

Figure 2: Results when button 1 (right) was pressed

The state of D0 was represented by the red LED (LED 1), the state of D1 by the yellow LED (LED 2), the state of D2 by the green LED (LED 3), and the state of D3 by the blue LED (LED 4).

4 Results

When no buttons were pressed (both Line 1 and Line 2 were low), LED 1 was on (D0 was high) (See Figure 2). When button 1 was pressed (Line 2 was high), LED 2 was on (D1 was high) (See Figure 3). When button 2 was pressed (Line 1 was high), LED 3 was on (D2 was high) (See Figure 4). When buttons 1 and 2 were pressed (Line 1 and Line 2 were high), LED 4 was on (D3 was high) (See Figure 5).

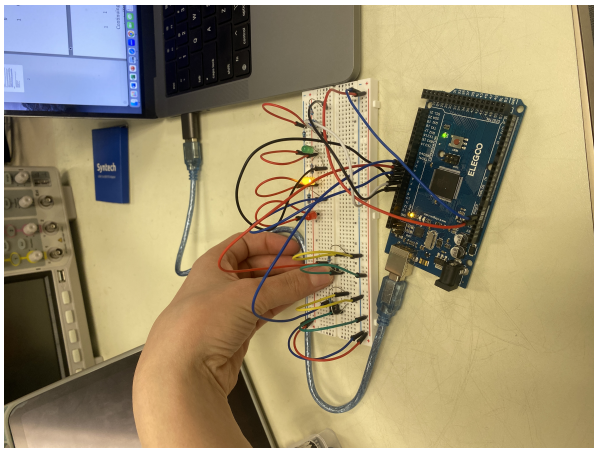

Figure 3: Results when button 1 (right) was pressed

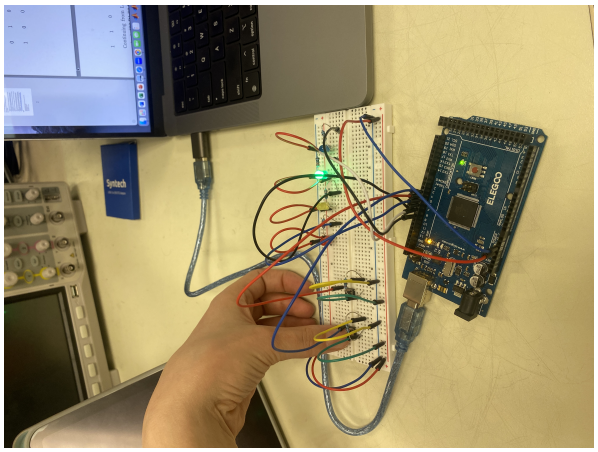

Figure 4: Results when button 2 (left) was pressed

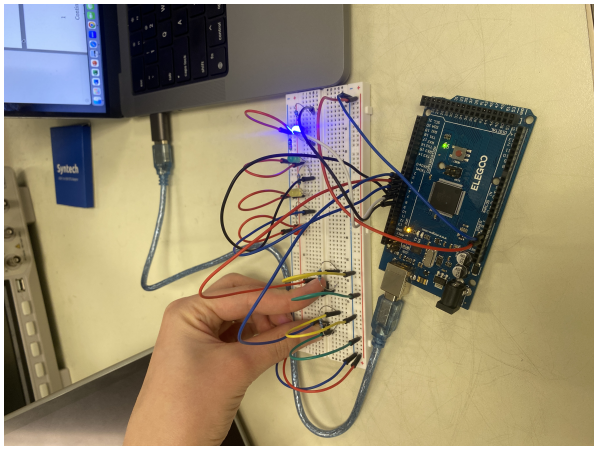

Figure 5: Results when both buttons were pressed.