SACRAMENTO
STATE

MARIO PALACIOS
LAB COURSE (CpE 64) SECTION #20
WEDNESDAY
EMMANUEL DUPART

Part 1.Breadboarding

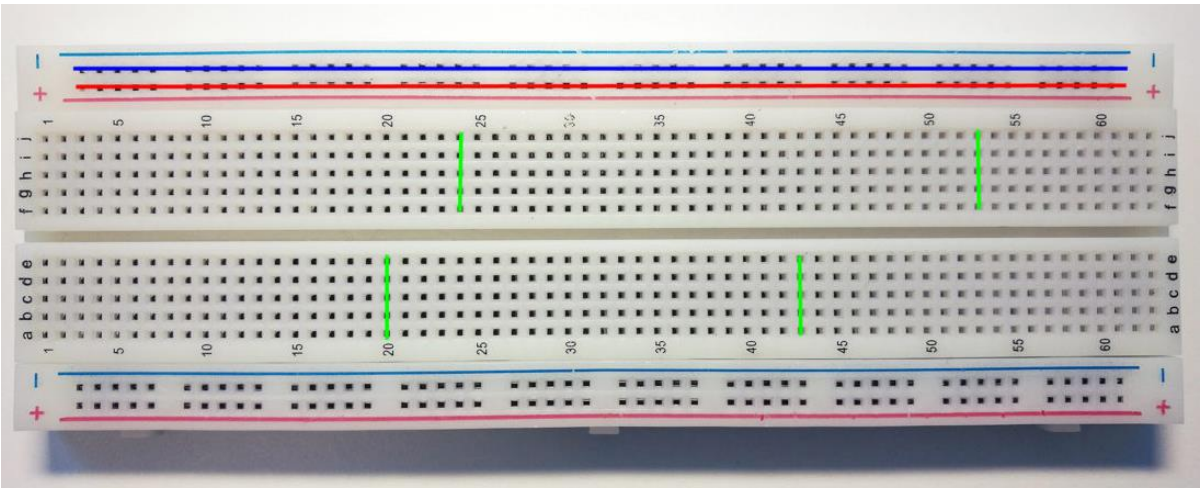
Description:

The objective this part of the lab is to get familiarization with the breadboard and the lines of continuity.

Problem Definition:

Examine the breadboard, figure out where the line of continuity are (what lines of holes are connected to other lines of holes). A Digital Multimeter set to a low resistance setting can be used to test where the lines of continuity are located. Knowing where the lines of continuity are crucial to testing circuit designs. The breadboard can be set up using switches for the inputs, along with Light Emitting Diodes and LED's for its outputs.

Engineering Data:



After testing the breadboard's continuity, I found out that buses (red and blue lines) have continuity and are best used to supply power to the breadboard. The bus strip is separated into groups of five and after 4 groups there is a gap, which means there is a break and the continuity stops. On the terminal strip the continuity is shown to be in vertical manner (green lines), and does not contain any break.

Conclusion:

After completing this part of the lab, I am to say that I understand how my breadboard works. It took me a while to see how the strips are connected, along with the terminology but now I have a better understanding. I will be using this knowledge in future labs. This part of the lab took me about 10 minutes to complete.

Part 2. Resistors Parallel – Series, and Bussed Resistors

Description:

The objective of this part is to learn resistor color codes, and compare them to resistor values. While reviewing how resistors in series and parallel are connected.

Problem Definition:

Choose three different resistors, measure their resistance using the color code and then a Digital Multimeter. To get the resistor's given measurement use the first two color band and multiply by ten to the power of the third color band. Compare both values and after connect them in series and parallel, following the figure. Measure the total resistance of circuit using values given by the color codes, and compare to the measured valued given by the Digital Multimeter.

Engineering Data:

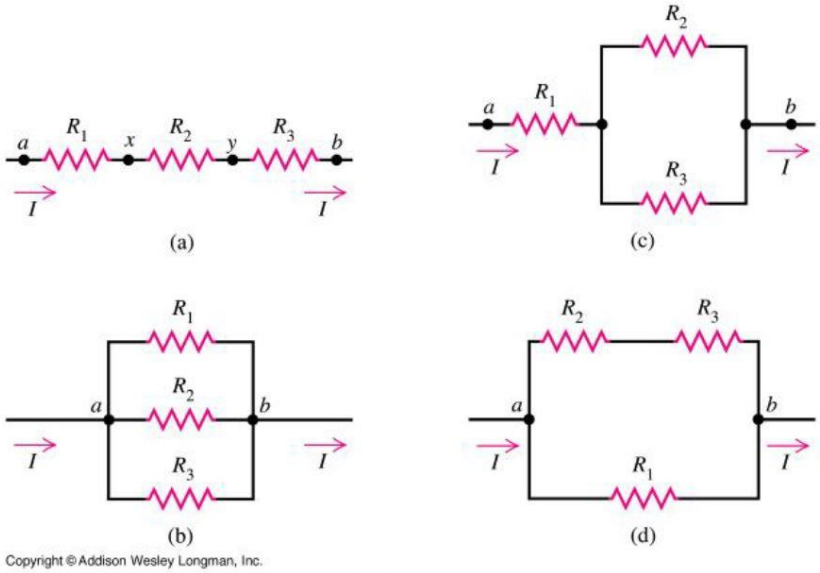


Figure showing (a) series resistors (b)parallel resistors (c) series R1 in parallel with R2 and R3 (d) R1 in parallel with R2 and R3 in series

Resistor	Color Code Value	Measured Value (Ohmmeter)
(R1) Blue, Grey, Orange	68,000 Ω	67,200 Ω
(R2) Brown, Black, Orange	10,000 Ω	9,790 Ω
(R3) Brown, Black, Yellow	100,000 Ω	97,600 Ω

Table 1. Calculated and Measured Values of Resistors

Resistor Configuration	Calculated	Measured Value (Ohmmeter)
(a) Series Resistors	178,000 Ω	173,900 Ω
(b) Parallel Resistors	8018.87 Ω	7,860 Ω
(c) Series R1 in Parallel with R2 and R3	77,090.9 Ω	76,100 Ω
(d) R1 in parallel with R2 and R3 in Series	42,022.4 Ω	41,300 Ω

Table 2. Calculated and Measured Values of Circuit Designs

The tested values can be found on tables 1 and 2, where the calculated and measured values were recorded. In Table1 R1's given value is 68,000 Ω but it was measured at 67,200 Ω, giving it a percent error of 1.18%. The percent error of R2 is 2.1% and R3 is at 2.4%. Table 2 calculated value for Resistor Configuration (a) was found by adding (R1 + R2 + R3) and after measuring it with the Digital Multimeter (DMM) the percent found was 2.3%. Resistor Configuration (b) was calculated as $(1/R1 + 1/R2 + 1/R3)^{-1}$ and the percent error would be 1.98%. Resistor Configuration for (c) was calculated as $[(1/R2 + 1/R3)^{-1} + R1]$ and the percent error was 1.29%. The last Resistor Configuration (d) was calculated as $[R2 + R3 + (1/R1)^{-1}]$ and the percent error was 1.72%.

Conclusion:

It is important to measure the actual resistance in each resistor because the given value from the color bands will differ from the value given from a DMM. This can be found by looking over the tables. The bus resistors is a group of resistors put in series with an output that has no resistance. This output is noted with either a triangle or a dot. They can come in forms of Single In-line Package (SIP), single row of connection tips, or Dual In-line Package (DIP), two rows of connection tips.

Part 3. LED and bar LED's

Description:

Get comfortable creating a circuit that uses switches as inputs and light emitting diodes (LEDs) as outputs and use resistors for LED current limiting.

Problem Description:

Connect a current limiting resistor of about 330 Ω value in series with the Bar LED. Then connect the two switches to the Bar LED. Power the breadboard and verify the LEDs lights when the switches are turned off and on.

Engineering Data:

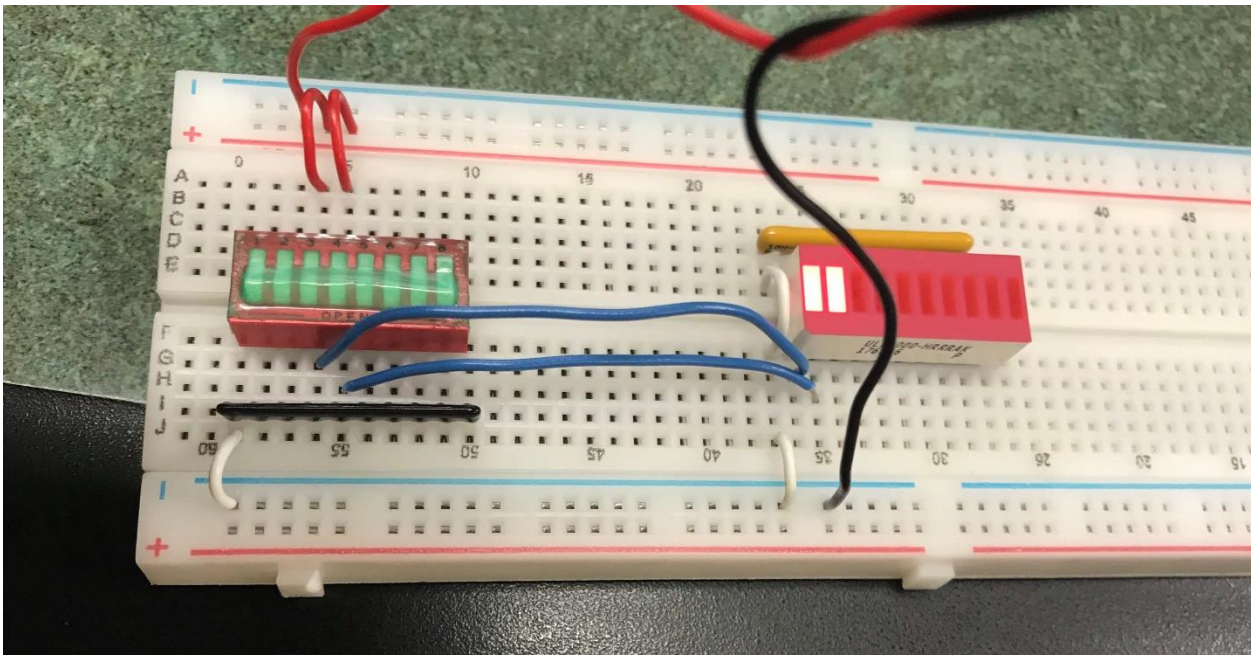


Figure 2: Bar LED Circuit Design with Switches Turned On

After connecting the bus resistors in series with both the switches and Bar LED, the lights were able to be turned on, as seen in figure 2. There had to be ground connections made in order to pass current to the breadboard, it can be seen by the white wires.

Conclusion:

Overall building this circuit is a combination of parts two and one, because without that knowledge a circuit could not be built. The circuit design consisted of two bus resistors that are in series, and the output that has not resistance is used as the ground, allowing for current to flow through the resistors and into the switches and Bar LED.

Part 4. Analog Discovery

Description:

The goal of this part of the lab is to get familiarization with Analog Discovery Kit.

Problem Description:

Use the video links provided to get set up and knowledge on how to use the Analog Discover Kit.

Quickstart #1: Getting Started http://www.youtube.com/watch?v=aYgFKIsrOYQ

Quickstart #2: Voltage Tool http://www.youtube.com/watch?v=Gdl7hPFaPWI

Quickstart #3: Voltmeter Tool http://www.youtube.com/watch?v=Va1lURqbmew

Engineering Data:

Overall the Analog Discovery Tool the Voltage Tool allowed for the circuit design made in part 3 to be powered on. The Voltmeter Tool was used to determine the amount of volts passing through the circuit, which was 4.27V with a percent error of 14.6%. I was not able to find any photos on this part of the lab.

Conclusion:

The Analog Discovery Kit is a device that can be used when there is no oscilloscope, voltage source, or even a DMM. This device allows students to do labs at home properly. This part of the lab took me the longest because there was only a few available to use.

Part 5. MultiSim Schematic and Breadboard

Description:

In this part of the lab we learn how to create schematics on a software called MultiSim.

Problem Description:

Download the template MultiSim file from moodle and open it. Use the schematic and replace the logic gate used to a 7400 NAND gate.

Engineering Data:

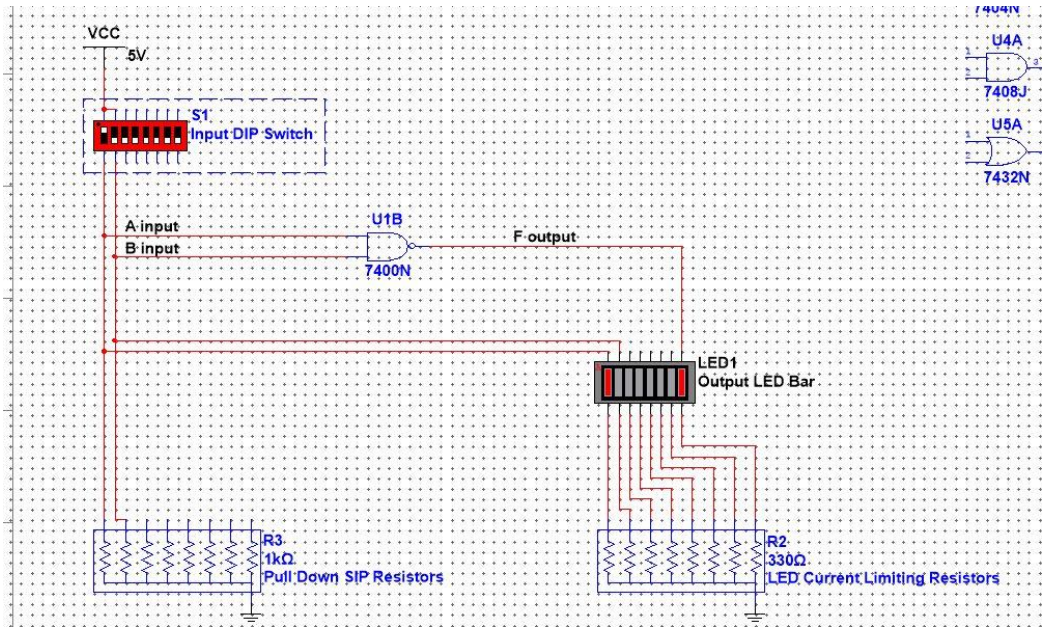


Figure 3. MultiSim Schematic with 7400 NAND Gate

Figure 3 shows the schematic from MultiSim that shows the 7400 NAND gate. When running the simulation when both switches are off then the light at the end is still on but when both switches are on then the end light is off.

Conclusion:

After completing this part, I was able to understand how to add and replace a logic gate on an already existing schematic. I also learned that the gate put in is an AND gate where a light at the end would be on if either light one or two would be on. However if both lights were on then the light at the end would be off.

Part 6. Fundamental Programming Refresher

Description:

The objective of this part is to review the fundamental programming concepts.

Problem Description:

Create the following programs in C language using any IDE or complier.

- Write a program that will display “Hello World.”
- Write a program that adds two numbers
- Write a program that will take a number input between 0-9, and displays a message whether the number is less than or greater than 5
- Write a program that will print your name 10 times using a “for” loop. Repeat for while, do-while loops
- Write a program that will take a number input between 0-4, and using switch case statement. The program will display a different message for each input

Engineering Data:

```
#include <stdio.h>
#include <stdlib.h>
int main(void){

        //Print Statement
    printf ("Hello World\n");

        //Sum of two numbers
    int a = 5;
    int b = 10;
    int c = a + b;
    printf ("Sum of a and b = %d\n", c);

        //greater/less than 5
        int num;
    printf ("Pick a number from 0-9\n");
    scanf ("%d", &num);
    if (num < 5) {
        printf ("less than 5\n");
    } else {
        printf ("greater than 5\n");
    }

    //Fruity LOOPS
        /*FOR LOOP*/
    for (int i = 1; i <= 10; i++) {
        printf ("Mario Palacios %d\n", i);
    }
    int j = 5;
        /*WHILE LOOP*/
    while (j <= 10) {
        printf ("Mario Palacios %d\n", j);
        j++;
    }
```

```
Hello World
Sum of a and b = 15
Pick a number from 0-9
12
greater than 5
Mario Palacios 1
Mario Palacios 2
Mario Palacios 3
Mario Palacios 4
Mario Palacios 5
Mario Palacios 6
Mario Palacios 7
Mario Palacios 8
Mario Palacios 9
Mario Palacios 10
Mario Palacios 1
Mario Palacios 2
Mario Palacios 3
Mario Palacios 4
Mario Palacios 5
Mario Palacios 6
Mario Palacios 7
Mario Palacios 8
Mario Palacios 9
Mario Palacios 10
```

Figure 4. Code From “Hello World” to While Loop with Outputs

```
        int k = 1;
        /*DO-WHILE*/
    do {
        printf ("Mario Palacios %d\n", k);
        i++;
    } while (k <= 10);

        //Switch Statements
    int choice;
    printf ("Choose a number from 0 - 4\n");
    scanf ("%d", &choice);
    switch(choice) {
        case 0: printf ("Mac n' Cheese");
        break;
        case 1: printf ("Chicken Nuggets");
        break;
        case 2: printf ("Hot Wings");
        break;
        case 3: printf ("Hamburger");
        break;
        case 4: printf ("Pizza");
        break;
        default: printf ("Try again, choose a number from 0 - 4");
        break;
    }

    return 0;
}
```

```
Mario Palacios 1
Mario Palacios 2
Mario Palacios 3
Mario Palacios 4
Mario Palacios 5
Mario Palacios 6
Mario Palacios 7
Mario Palacios 8
Mario Palacios 9
Mario Palacios 10
Choose a number from 0 - 4
1
Chicken Nuggets
```

Figure 5. Code from Do-While Loop to Switch Case Statements with Outputs

Conclusion:

After completing this part I was able to get a refresher on the key bits of programming code that will be needed for the rest of the course. This was the first writing in C but it is very similar to Java except some syntax is different.