

Raspberry Pi Based Mainframe Simulator

Brent Seidel
Phoenix, AZ

November 22, 2021

This document is ©2021 Brent Seidel. All rights reserved.

Note that this is a draft version and not the final version for publication.

Contents

1	Introduction	1
1.1	Parts	1
1.1.1	3D Printing	1
1.1.2	Electronic Parts	1
1.1.3	Mechanical Hardware	2
2	Hardware	3
2.1	Electronics	3
2.1.1	Processor	3
2.1.2	Discrete I/O	3
2.2	3D Printed Parts	3
2.2.1	Racks	3
2.2.2	Processor Tray	3
2.2.3	Discrete I/O Tray	4
2.2.4	Address/Data Panels	4
2.2.5	Control Panel	4
2.2.6	Mode Panel	6
2.3	Assembly	6
2.3.1	Planning	6
2.3.2	Discrete I/O Boards	6
2.3.3	Panel I/O Tray	10
2.3.4	I2C Bus Cable	13
2.3.5	Panel LEDs	15
2.3.6	Panel Switches	19
2.4	Power	20
3	Software	21
3.1	Overview	21
3.2	Simulation	21
3.3	Web Server	22
3.4	Other Software	22

List of Figures

2.1	GPIO Board with Socket and Resistors	8
2.2	GPIO Board with Socket, Resistors, and Fixed Wires	8
2.3	GPIO Board with Headers	9
2.4	Panel I/O Tray with Connectors	10
2.5	Panel I/O Tray with GPIO Boards	11
2.6	Panel I/O Tray with GPIO Boards with Address and Reset Jumpers Installed. These are Control LED and Control Switch Boards	11
2.7	Panel I/O Tray with GPIO Boards with Power, I2C, and MPC23017s Installed	11
2.8	Panel I/O Tray with LSB LED and Switch Connectors Wired	12
2.9	Panel I/O Tray with MSB LED and Switch Connectors Wired	12
2.10	Back Side of Panel I/O Tray Showing Connector Wiring	12
2.11	The I2C Bus Cable	13
2.12	An I2C Bus Cable Connector	14
2.13	LED Panel Connector with Ground Wire	16
2.14	Attach the Cable to the Panel	16
2.15	Solder the First LED	17
2.16	Solder the Second LED and Add a Cable Tie	17
2.17	Solder the Remaining LEDs on That Side and Add Cable Ties	18
2.18	The Completed Panel	18
2.19	Control Panel Switches Showing Switches Installed	19
2.20	Control Panel Switches Showing Wiring	19
2.21	Control Panel with Switches and LEDs Installed and Wired	20

List of Tables

2.1 Example 16 Bit Processor Layout	7
2.2 Improved 16 Bit Processor Layout	7
3.1 Simulations Available	21

Chapter 1

Introduction

A rather simplistic view is that this provides a way to get a Raspberry Pi to blink some LEDs and read some switches. However, this hardware could be modified for a number of other applications that require lots of discrete I/O, such as burglar alarms or sprinkler controls.

1.1 Parts

1.1.1 3D Printing

A fair bit of 3D printing is required, so having lots of filament is a good idea. Labels on the panels are done by (on my printer with a 0.15mm layer height) printing 2 layers of the panel color, then 3 layers of a contrasting color for the text, and then the rest in the panel color. The panels are printed with the text faced down. You may wish to tinker with this for your printer. The actual parts are:

- Four or six racks depending on whether you want a 16 or 32 bit system.
- One processor tray.
- Two or three discrete I/O trays.
- One control panel
- One modes panel
- Two or four address/data panels (starting at 0, 8, 16, and 24).

1.1.2 Electronic Parts

Note that soldering will be required so a soldering station with appropriate tools and solder will be required in addition to the following.

- One Raspberry Pi 3B computer (available from many places).

- Four or six half-size perma-proto breadboards (p/n 1609 for qty 1 or p/n 571 for qty 3 from Adafruit)¹.
- One quarter-size perma-proto breadboard (p/n 1608 from Adafruit).
- At least four or up to eight MCP23017 I2C 16 bit input/output port expander chips (p/n 732 from Adafruit).
- One 28-pin DIP socket per MCP23017. (eg p/n A120353-ND from Digi-Key)
- Recommended one LTC4311 I2C Extender/Active Terminator (p/n 4756 from Adafruit).
- One set of Female/Male jumper wires (p/n 1954 from Adafruit). These are used to connect from the Raspberry Pi to the rest of the circuitry since female connectors are needed on the Raspberry Pi end).
- Hookup wire, preferably multi-colored (available from many places).
- 36 pin 0.1" female header (5 pack p/n 598 at Adafruit). You may need a couple packs.
- 23 to 39 Mini panel mount SPDT toggle switches (p/n 3221 from Adafruit). Feel free to use other switches if you like, though you may need to adjust the hole sizes in the panels.
- 11 or 16 female and male DB-9 connectors (eg p/n AE10972-ND (male) and p/n AE11063-ND (female) from Digi-Key (buy a pack of 25 for a price discount)).
- Lots of 5mm oval LEDs (I used red, but you can use other colors) (eg p/n C5SMF-RJF-CT0W0BB1-ND from Digi-Key (buy a pack of 100 for a price discount, you can always use the excess in other projects)).
- Lots of 120Ω resistors (eg p/n CF14JT120RCT-ND (lots of other options) from Digi-Key). These are used for current limiting the LEDs from 3.3V drive. If you're driving the LEDs with 5V, you'll need a bit more resistance. Since the Raspberry Pi uses 3.3V, these will work for this project.
- A couple meters of CAT-5 cable. This is for the cables to connect the various assemblies (available from many places).

1.1.3 Mechanical Hardware

I have a Phillips Panhead Electronic Hardware Kit from Digi-Key (p/n 1602-KIT-ND) that has an assortment of machine screws and nuts. I used these because they were handy.

You will need many cable ties. I have a container of assorted cable ties from Costco. These can be obtained from many other sources.

¹Adafruit generally throws one of these in with your order if your order is over \$100

Chapter 2

Hardware

2.1 Electronics

2.1.1 Processor

I had a Raspberry Pi 3 board laying around so I figured that I would use it. Interestingly, this is a more powerful computer than many of the processors it might be used to simulate.

2.1.2 Discrete I/O

Each of these boards is built on a 1/2 size perma-proto board from Adafruit. It has one MCP23017 16 bit I/O expander chip with I2C interface in a socket. Three pins are provided to configure the address, so the system can have up to 8 of these chips or 128 discrete I/O pins. There is space on the board for 8 current limiting resistors for LED drive.

2.2 3D Printed Parts

2.2.1 Racks

Each rack is a bbs_rack2(10, 6, 4). The size is basically constrained by what my 3D printer can do. Each of these requires about 15 hours to print on my printer. Two of these can be bolted together side-by-side and fit in a 19 inch rack. This is the approximate size of the classic computer equipment. Multiple racks can be bolted together vertically as needed to provide panel space. Each rack has slots for two vertically oriented trays.

2.2.2 Processor Tray

The processor tray has mounting holes for the Raspberry Pi processor board. It also has a 1/4 size perma-proto board from Adafruit (this may get changed to a half sized board) to provide space for miscellaneous wiring. Also on the tray is a LTC4311 I2C Extender/Active Terminator (p/n 4756 from Adafruit). This may not strictly be necessary, but since the I2C bus will be running off to

several other boards, I thought that it might be a good idea. There is also a DB-9 connector that carries power and the I2C bus to the other trays.

2.2.3 Discrete I/O Tray

Each of these trays has one or two MCP23017 16 bit I/O expander chips with I2C interface. These trays would typically be mounted in the slot next to the panel. The tray has 4 DB-9 connections and can interface with up to 16 LEDs and 16 switches. One additional DB-9 connection is provided to connect the the power and I2C bus from the processor.

2.2.4 Address/Data Panels

The address/data panels provide 8 switches and LEDs numbered consecutively from right to left. The starting number is selectable in the model. Each panel has two cables (one for LEDs and one for switches) that attach to connectors on the discrete I/O boards. Enough of these need to be provided to account for all the address and data lines in the simulated processor. A system would typically have two or four of these panels for 16 or 32 bit systems. Note that there are many cases where the number of address lines does not match the number of data lines. In these cases panels would need to be provided for whichever number is greater.

2.2.5 Control Panel

At this point, without an actual CPU simulation, this is mostly a way to organize what needs to be in a relatively generic control panel. The following functions are needed:

- Read from a memory location.
- Write to a memory location.
- Stop/pause processor execution.
- Start processor execution at a specified address.
- Continue processor execution.
- Reset the processor.

The following switches are defined on the panel:

- “Run”/“Pause” - The “Run” position allows program execution. The “Pause” position temporarily suspends execution. Moving the switch back to the “Run” position will continue where the program left off.
- “Start”/“Stop” - Moving the switch to the “Start” position loads a starting address to begin program execution. The “Stop” position halts a program and a new starting address will need to be provided.
- “Auto”/“Man” - The “Auto” position allows for remote control of the simulator via a web interface. The “Man” position disables remote control.
- “Addr”/“Data” - Selects display and entry of memory addresses or data.

- “Dep” deposits the switch value. If the “Addr”/“Data” switch is in the “Addr” position, a memory address is loaded. If the “Addr”/“Data” switch is in the “Data” position, the switch value is deposited in memory and the address incremented.
- “Exam” is used to examine memory at the current address. The address is incremented each time the switch is moved to the “Exam” position.
- Spare - an empty position reserved for future expansion.
- “Power”/“Off” - This will eventually be wired up to control power to the Raspberry Pi computer.

To Specify an Address

Note that if the word size is greater than 8 bits, some of the lower order address bits may be ignored.

- Toggle the desired address using the address/data switches.
- Move the “Addr”/“Data” switch to the “Addr” position.
- Move the “Dep” (deposit) switch to the “Dep” position.
- Once the LED for the “Dep” switch illuminates, move the switch to the unlabeled position.

To Read Memory

- Specify the starting address as described above.
- Move the “Addr”/“Data” switch to the “Data” position.
- Move the “Exam” switch to the “Exam” position.
- The contents of memory at the specified address will be displayed in the Address/Data LEDs.
- The “Addr”/“Data” switch can be used to toggle between display of the address and the data.
- Move the “Exam” switch to the unlabeled position. This increments the internal address pointer so that by toggling this switch a block of memory can be modified.

To Write Memory

- Specify the starting address as described above.
- Move the “Addr”/“Data” switch to the “Data” position.
- Toggle the desired data using the address/data switches.
- Move the “Dep” switch to the “Dep” position.
- The contents of memory at the specified address will be updated and displayed in the Address/Data LEDs.

- The “Addr”/“Data” switch can be used to toggle between display of the address and the data.
- Move the “Dep” switch to the unlabeled position. This increments the internal address pointer so that by toggling this switch a block of memory can be examined.

To Start Execution at a Specific Address

- Move the “Start”/“Stop” switch to the “Stop” position.
- Move the “Addr”/“Data” switch to the “Addr” position.
- Toggle the address using the address/data switches.
- Move the “Start”/“Stop” switch to the “Start” position.
- If the “Run”/“Pause” switch is in the “Pause” position, move it to the “Run” position.

2.2.6 Mode Panel

This is a display only panel to show the processor and memory access modes. It has seven labeled positions in two groups separated by a spare. The first group is the processor mode. This indicates which of four possible modes (User, Supervisor, Executive, or Kernel) the processor is in. The second group indicates the type of memory access. There are three possibilities: Instruction (read-only), Data (read/write), and Interrupt (read-only).

2.3 Assembly

2.3.1 Planning

The first decision is if you want to simulate a 16 bit or 32 bit (or something else) system. Each panel can hold up to 8 LEDs and switches. Each MCP23017 can control 16 LEDs or switches, so 2 MCP23017s can control 2 panels. Each rack has space for 1 panel and two circuit trays. If you attach two racks side by side, they will approximately fit in a 19 inch rack (as I don’t have a standard 19 inch rack, you will probably have to design and print some sort of adapter).

Once you’ve decided on your system size, then you need to figure out how to arrange things. See Table 2.3.1 for an example arrangement. Upon assembling this system, I discovered a drawback—the top of the Control Panel Controller interferes with the cable connecting the controllers to the processor. An improved arrangement is in Table 2.3.1.

Once your arrangement is finished, then you can go ahead and figure out how long your cables need to be. After you have the cables from the panels to the controllers finished, it becomes much harder to move things around. Ask me how I know.

2.3.2 Discrete I/O Boards

The discrete I/O boards offer 16 bits of GPIO. Each can be configured as an input or an output. There is not enough space on the breadboard for 16 current limiting resistors for LEDs. So, the design has 8 resistors on each board. Two boards are used on a panel I/O tray. One board drives

Table 2.1: Example 16 Bit Processor Layout

Processor	empty
Address Data Panel Controller	empty
Address/Data 8-15 Panel	Address/Data 0-7 Panel
empty	empty
Control Panel Controller	empty
Mode Panel	Control Panel

Table 2.2: Improved 16 Bit Processor Layout

Processor	empty
empty	Address Data Panel Controller
Address/Data 8-15 Panel	Address/Data 0-7 Panel
empty	empty
Control Panel Controller	empty
Mode Panel	Control Panel

the LEDs and one reads the switches. The LED board uses the 8 resistors on itself as well at the 8 resistors on the switch board.

- First, solder the resistors and socket. See Figure 2.1.
- Then add some fixed wires. See Figure 2.2.
- Solder in the female header strips. Some of these can be omitted and the wires in the next step soldered in directly, but having these will make reconfiguration easier. See Figure 2.3.
- Add local wires to the header strips. You may wish to wait until the board is installed in the Panel I/O Tray before doing this.

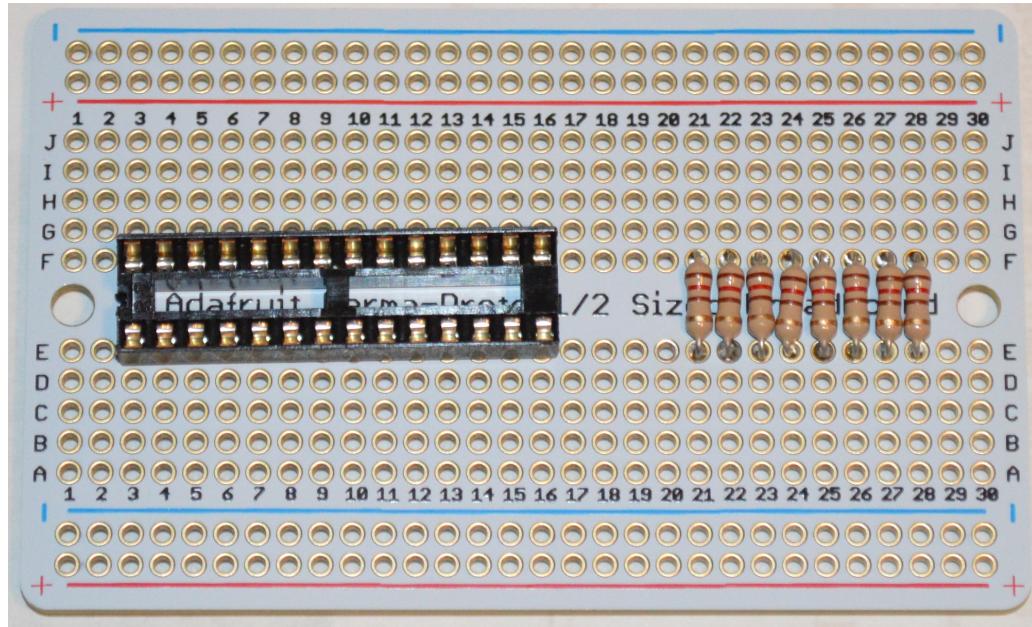


Figure 2.1: GPIO Board with Socket and Resistors

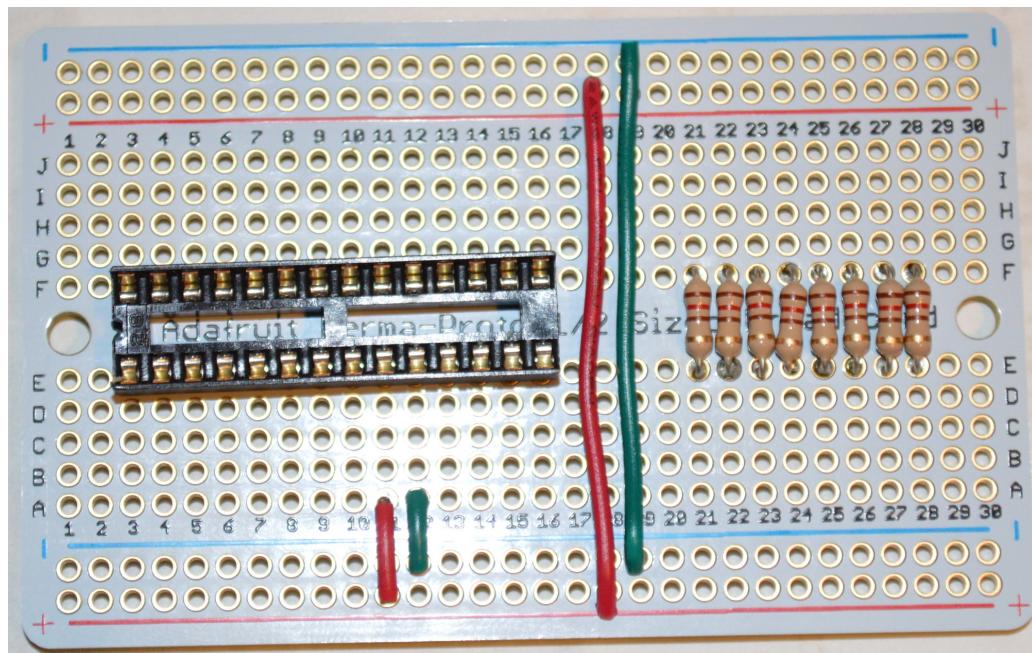


Figure 2.2: GPIO Board with Socket, Resistors, and Fixed Wires

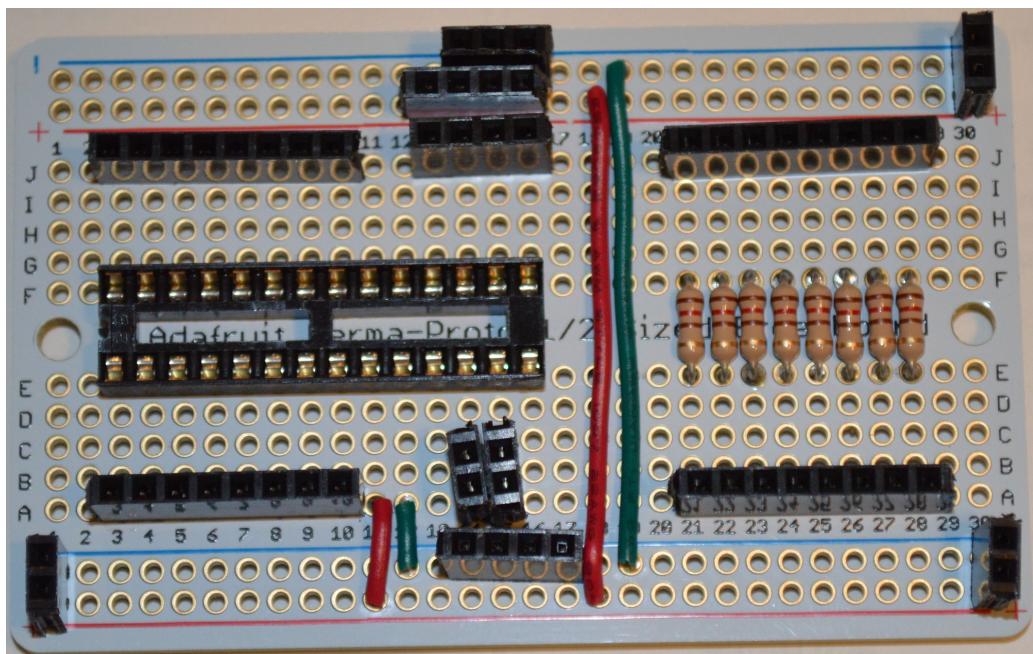


Figure 2.3: GPIO Board with Headers

2.3.3 Panel I/O Tray

There are two discrete I/O boards on this tray. Together they can drive up to 16 LEDs and read 16 switches. The left board drives the LEDs and the right board reads the switches. The connectors from left to right are: MSB LEDs, MSB switches, LSD switches, and LSB LEDs.

It turns out that 8 of the 16 GPIO pins on the MCP23017 are numbered ascending down one side of the chip and the other 8 are numbered descending down the other side. To account for this, you can either have the jumpers cross over for two of the panel connectors (I did this in the first version), or simply rotate two of the connectors by 180°, thus putting the twist into the cable.

- Attach the connectors to the tray. See Figure 2.4.
- Attach the two Discrete I/O Boards. See Figure 2.5.
- Add wiring for power, I2C, and board configuration (Figure 2.6 shows the address and reset jumpers installed. From the address jumpers, these are the control LED and control switch boards) and install the MCP23017s. See Figure 2.7. If the processor and bus cable are ready, this can be hooked up to see that the processor detects the MCP23017s.
- Add wiring to the connectors. See Figures 2.8, 2.9, and 2.10. Also refer to subsection 2.3.5 to make sure that the connector pinouts match those of the panel being connected.

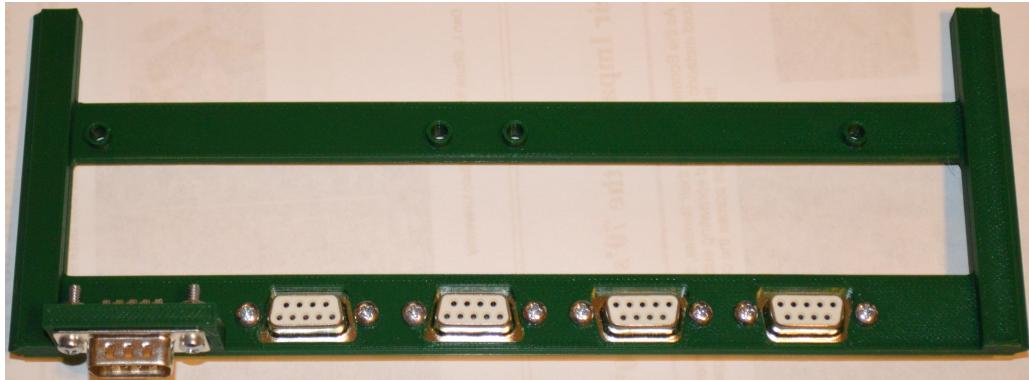


Figure 2.4: Panel I/O Tray with Connectors

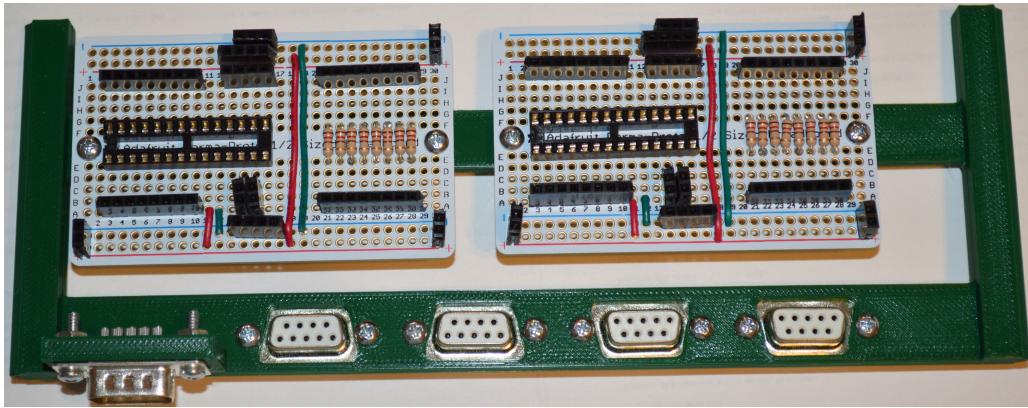


Figure 2.5: Panel I/O Tray with GPIO Boards

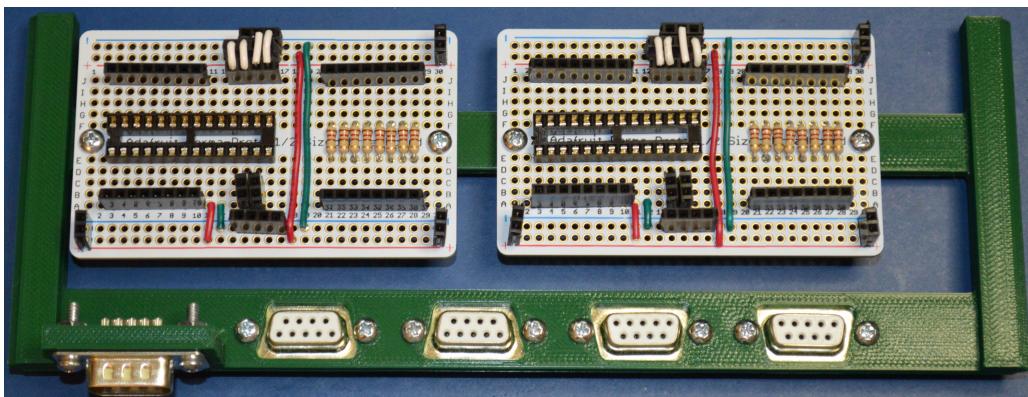


Figure 2.6: Panel I/O Tray with GPIO Boards with Address and Reset Jumpers Installed. These are Control LED and Control Switch Boards

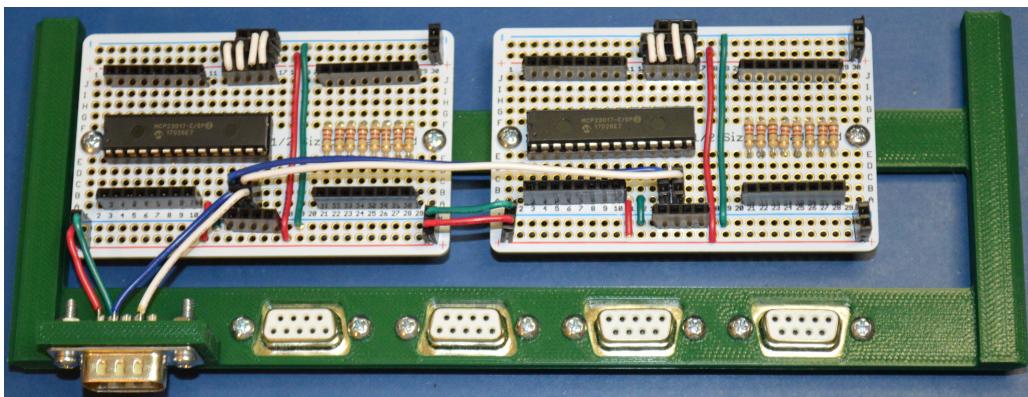


Figure 2.7: Panel I/O Tray with GPIO Boards with Power, I2C, and MPC23017s Installed

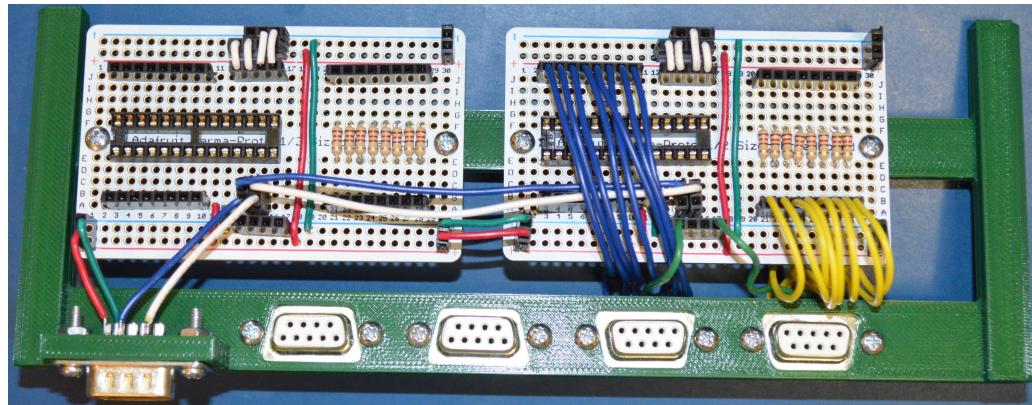


Figure 2.8: Panel I/O Tray with LSB LED and Switch Connectors Wired

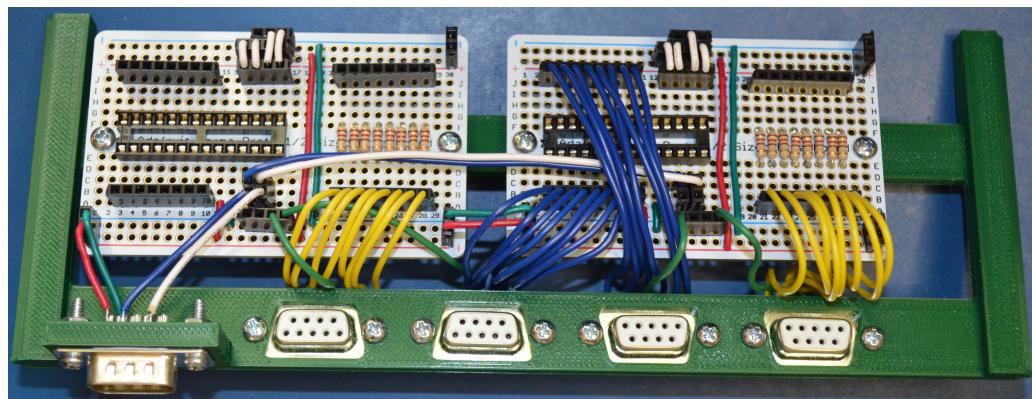


Figure 2.9: Panel I/O Tray with MSB LED and Switch Connectors Wired

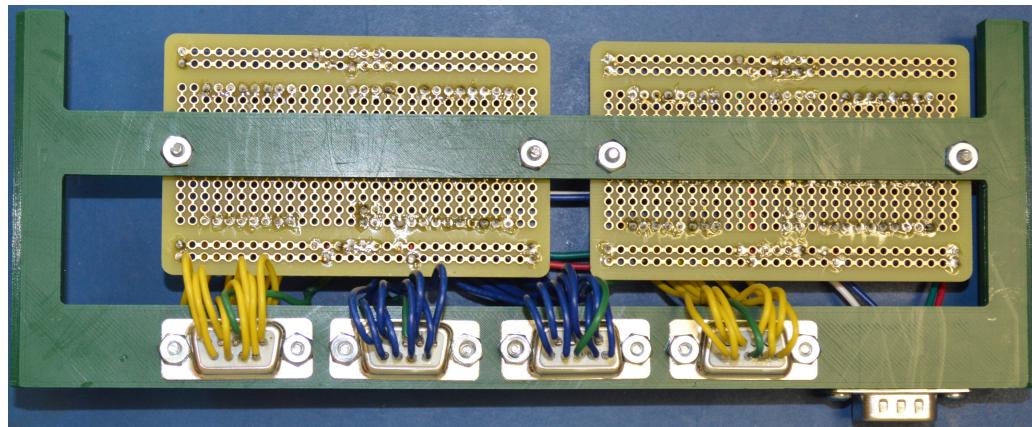


Figure 2.10: Back Side of Panel I/O Tray Showing Connector Wiring

2.3.4 I2C Bus Cable

The I2C bus cable is probably the most difficult bit of soldering as you will have to add two wires to each contact in a connector. Multiple times. Which pairs are used for which pins isn't particularly important, but I wire pins 2 & 6, 2 & 7, 4 & 8, and 5 & 9 as pairs. Pin 3 is left unconnected. It is obviously important to have the same wire go to the same pin on each connector on the bus. Figure 2.11 shows an example cable. Figure 2.12 is a close-up of one of the connectors. It is a little easier to decide ahead of time how many connectors you want on the bus. Soldering a cable segment to an existing connector can be done, but is more difficult.

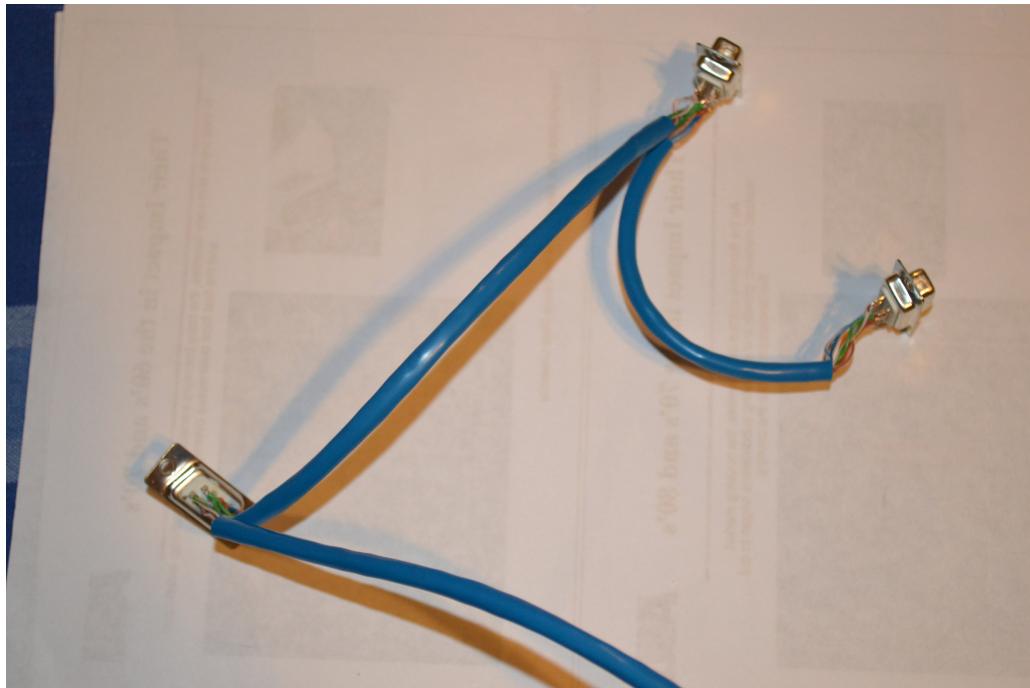


Figure 2.11: The I2C Bus Cable

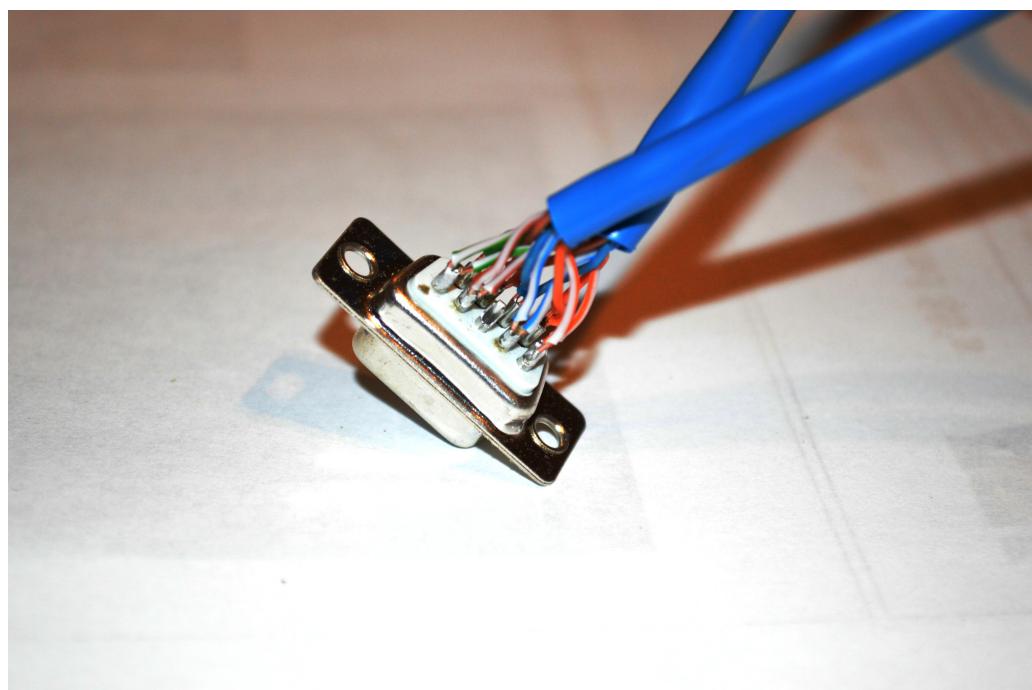


Figure 2.12: An I₂C Bus Cable Connector

2.3.5 Panel LEDs

The wiring for the panel LEDs is similar for all panels. The Modes Panel is shown as an example. The LEDs on the other panels would be wired the same way. Note the cable ties used to attach the wiring to the panel. This also helps to keep the LEDs in place so that they don't push out.

You can choose whatever pinout you want for the connectors, however it helps if there is some reason and consistency behind it. What I have done is to look at the connector from the back with the wide side up and assign the pins in the same order as the LEDs or switches, with the very middle pin assigned to ground (as shown in Figure 2.13). That is, the left-most pin corresponds to the left-most LED or switch, and so on.

- Since CAT-5 cables are only 8 conductor, a separate ground/return wire needs to be added. As a note of caution, I find that the orange and brown pairs are difficult to distinguish. Be careful, otherwise you may wind up having to resolder the connector. See Figure 2.13.
- First using a cable tie, attach the cable to the panel. This helps to keep the cable from moving around while the LEDs are soldered. There is also a bare copper wire used as a ground/return wire for the LEDs. See Figure 2.14.
- Solder the first LED to one side of the cable. This one is the hardest as there is only a short length of wire to work with. See Figure 2.15.
- Solder the next LED and add a cable tie between them. See Figure 2.16.
- Solder the remaining LEDs on that side and add cable ties between them. See Figure 2.17.
- Find a place to solder the ground/return wire from the connector to the bare LED ground/return wire. See Figure 2.18.
- Complete the other side of the panel. See Figure 2.18.
- At some point, you may wish to trim the leads on the LEDs.

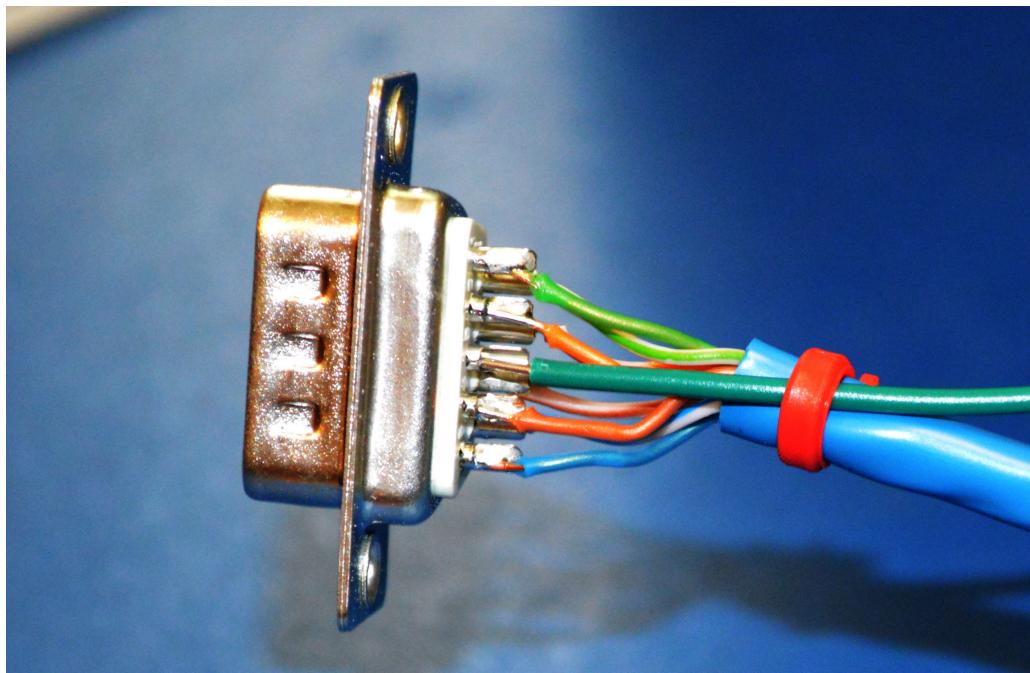


Figure 2.13: LED Panel Connector with Ground Wire

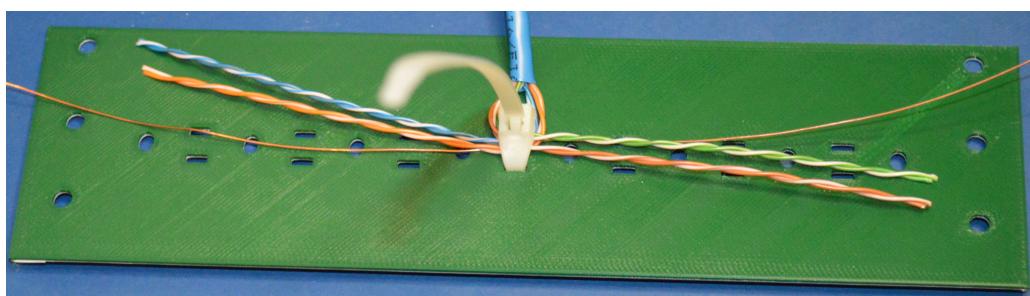


Figure 2.14: Attach the Cable to the Panel

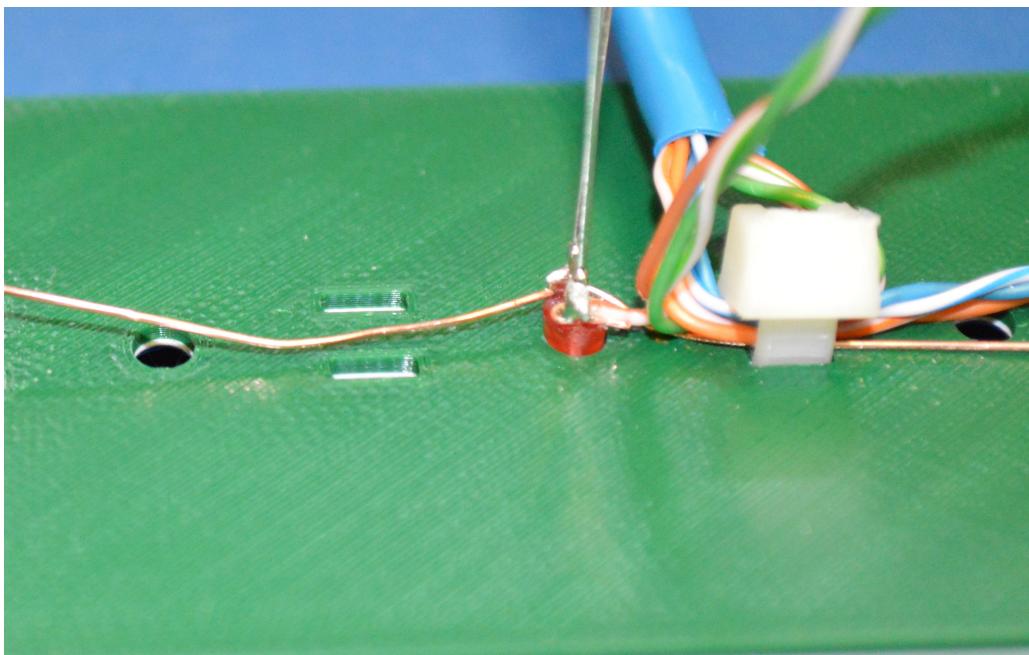


Figure 2.15: Solder the First LED

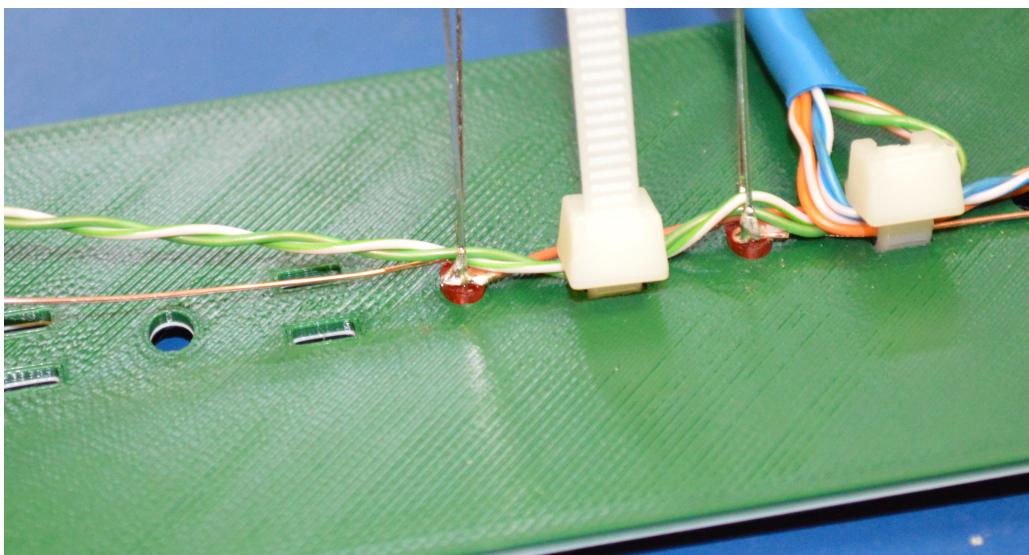


Figure 2.16: Solder the Second LED and Add a Cable Tie

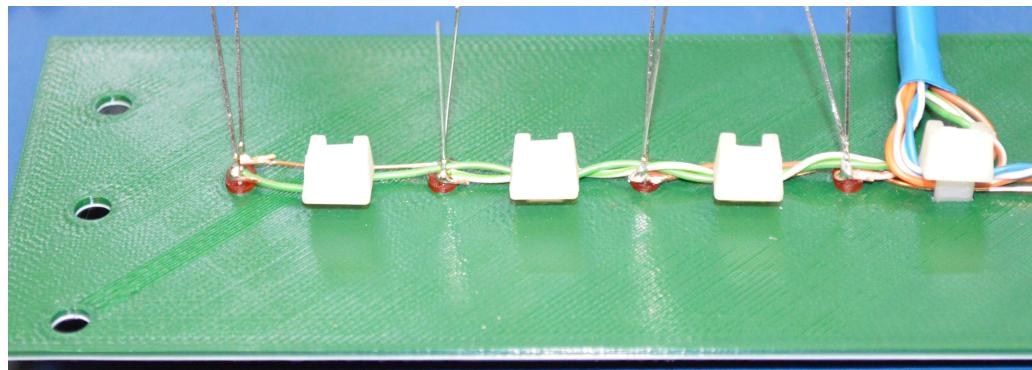


Figure 2.17: Solder the Remaining LEDs on That Side and Add Cable Ties

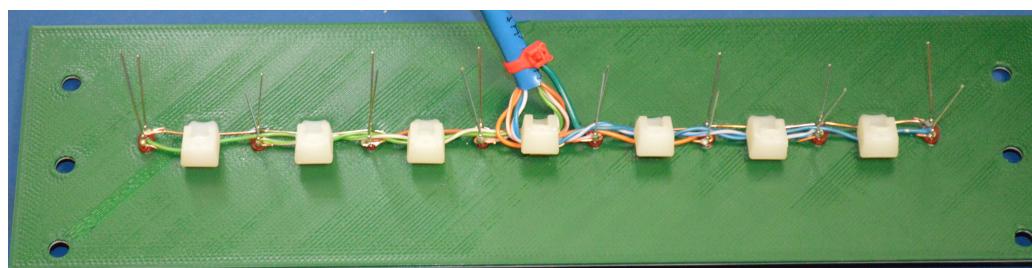


Figure 2.18: The Completed Panel

2.3.6 Panel Switches

My implementation uses SPST switches where the center connector is common. When the switch is in the up position, the center and lower connector are connected. When the switch is in the down position, the center and upper connector are connected. The MCP23017s have an optional pullup for each input, thus the switches can be used in an open-ground configuration. The top connectors are tied together and connected to ground. The center connectors are connected to the individual input pins.

- Install the switches as shown in Figure 2.19. This example shows a control panel that only has six switches.
- Install the wiring for the switches as shown in Figure 2.20. Since the control panel has only six switches, one of the cable pairs is unused.
- Finally, install the LEDs in the panel as described in Section 2.3.5 and shown in Figure 2.21.

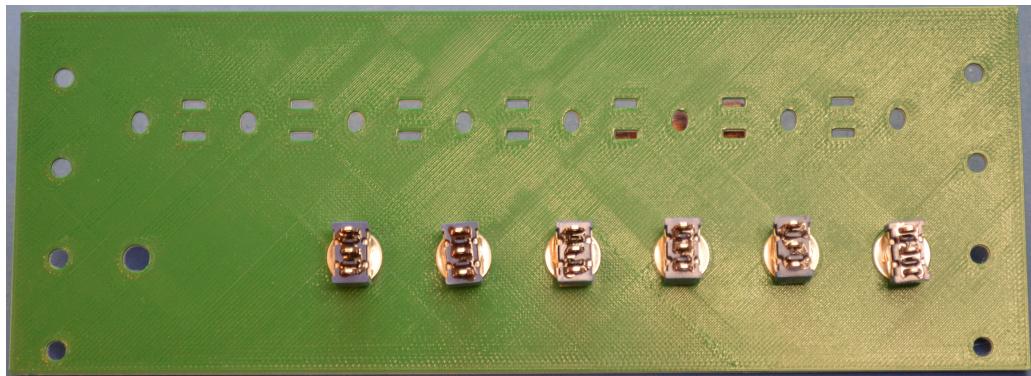


Figure 2.19: Control Panel Switches Showing Switches Installed

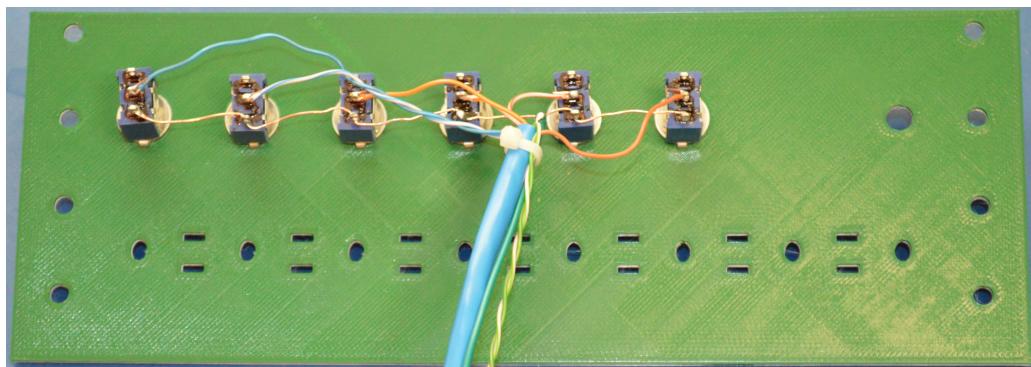


Figure 2.20: Control Panel Switches Showing Wiring

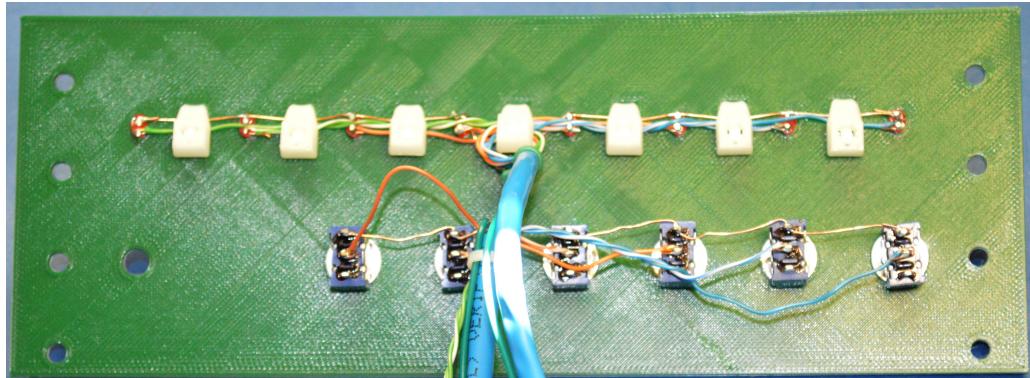


Figure 2.21: Control Panel with Switches and LEDs Installed and Wired

2.4 Power

You can go with just plugging a USB power adapter into the micro-USB connector on the Raspberry PI, or you can get fancier. There is a model for a power panel that has cutouts for micro-USB and a USB-B panel mount connectors. Adafruit offers a number of panel mount USB to various USB adapters. You can just use one of these. If you want to get fancier, you can cut open the USB cable and insert the power switch in series with the power conductor. At this point, you are on your own. Be careful!

Chapter 3

Software

3.1 Overview

Currently, the software has two main interfaces. The first is the switches and lights. The second is a web interface.

3.2 Simulation

There are a number of “simulations” available for testing or display. The desired simulation can be selected either using the switches, or, if enabled, using the web interface. The following is a description of the control switches:

- The “Power”/“Off” switch is intended to be hardwired to the power supply. There is no program function.
- The “Exam” switch has no current function.
- The “Dep” switch is used to select which simulation to run.
- The “Addr”/“Data” switch has no current function.

Table 3.1: Simulations Available

0	Copy Switch Values
1	Count
2	16-bit Scan
3	16-bit Bounce
4	Fibonacci Sequence
9	Count
10	32-bit Scan
11	32-bit Bounce
12	Fibonacci Sequence
other	Copy Switch Values

- The “Auto”/“Man” switch enables simulation selection by the web interface when in the “Auto” position.
- The “Start”/“Stop” switch sets all simulation variables to their initial state when moved to the “Start” position. It also must be in the “Start” position for the simulation to run.
- The “Run”/“Pause” switch temporarily pauses a simulation when in the “Pause” position. The simulation continues running without initialization when the switch is moved to the “Run” position.

Note that for a simulation to run, both the “Start”/“Stop” switch must be in the “Start” position and the “Run”/“Pause” switch must be in the “Run” position.

On program initialization, simulation 0 (Copy Switch Values) is selected by default. To select a different simulation, make sure that either the “Run”/“Pause” switch is in the “Pause” position or the “Start”/“Stop” switch is in the “Stop” position. Then:

- Using the “Addr”/“Data” switches, select the desired simulation as listed in Table 3.2.
- Move the “Dep” switch to the “Dep” position. You may need to move it to the unlabeled position first.
- Move the “Dep” switch to the unlabeled position.
- Move the “Run”/“Pause” switch to the “Run” position.
- Move the “Start”/“Stop” switch to the “Start” position.

Unless you wish to reset the simulation values, you should always leave the “Start”/“Stop” switch in the “Start” position.

3.3 Web Server

The web interface is available on port 31415 (configurable in web.adb). It uses plain, unencrypted HTTP.

3.4 Other Software

To enhance the experience, you can install some old computer emulation software (such as simh, available at <https://github.com/simh/simh>). Note that this will run completely independent from the lights and switch in this project.