Remote Lab Station

BCE 496: Capstone 2

Internal Specification

# Abstract

The Remote Lab System is an experimental system designed for electrical engineering students to connect and perform laboratory experiments remotely from off campus. This documentation contains details related to individual component usage including design schematics, algorithms, and graphical interface implementation.

Gregory Asher --- Student ID: 1566775

Gytautas Jankauskas --- Student ID: 1568743

Paolo Sebastian --- Student ID: 1573079

Andrew Soshea --- Student ID: 1321957

# 

[**Abstract**](#_d0pkawfgddcf) **1**

[**Overall Design**](#_okciobghu1cp) **4**

[**Hardware Design**](#_sltzt3qnmgy8) **5**

[Hardware Operation](#_nm3fdu7glk1f) 5

[Logic Analyzer](#_nhyg5y1gc5mc) 5

[Oscilloscope and Function Generator](#_u18qruee6c95) 6

[Voltmeter](#_m5oba6yif6kl) 6

[Linear Regulators](#_klun8ua2hv1y) 9

[AD5293 Digital Potentiometers](#_m2mqadi0dioh) 9

[Heat Dissipation](#_rf8pp5y5quuu) 10

[**Software Design**](#_k2kck6b4rirw) **11**

[Graphical User Interface (GUI)](#_69sa4dwo4k9g) 11

[External Software](#_hqdu6utdf3gk) 12

[Arduino](#_kl9lm51izgq5) 12

[readBoardID()](#_i01efsh5nuj) 12

[Linear Regulator Shutdown](#_z885wie26254) 12

[Safety Check](#_bragc6it2inx) 12

[Daughterboard](#_dsjimli0w1nn) 13

[Linear Voltage Regulator Control](#_iwmst88rk48v) 13

[Digital Potentiometer Control](#_f148mao9ivpa) 13

[**Test Plan**](#_du0lgcgu8p13) **14**

[Hardware](#_3hucrv8bbtoq) 14

[Individual and Independent Component Test Phase](#_ayqyqhanhpiq) 14

[LT1963 (Positive Linear Regulator)](#_7uvncigundwj) 14

[LT3015 (Negative Linear Regulator)](#_r9a4o2ped2td) 15

[ADS1115 16-bit ADC (on Adafruit breakout board)](#_wuck2xkxkzw3) 15

[AD5293 (Digital Potentiometer)](#_w3h4nh8rh4oj) 16

[ULN2803A (Darlington Transistor Array)](#_7lplnq858e04) 16

[RA30421051](#_7pqgsxskxrxe) 16

[LM337 (-5 Volt Regulator)](#_dhfhblpola6k) 16

[LM317 (+5 Volt Regulator)](#_38z9uugcmx5r) 16

[Motherboard](#_hpa62721h1r) 16

[Small Assembly Test Phase](#_1byddmskg1bn) 17

[Linear Regulator Circuit](#_gfuj7aofemn5) 17

[LM337 and LM317](#_smf7mmymolgo) 17

[Cooling Circuit](#_48r63qv7nrgw) 17

[Chassis](#_vzlrk4gx5kgc) 17

[Final Assembly Test Phase](#_1llddq8cdval) 18

[Software](#_npofll6y7rb4) 19

[White Box Software Testing](#_ufz3kcp2lfbz) 19

[Black Box Software Testing](#_o65amim7ch6j) 20

[**Project Schedule**](#_8p64onjyj3la) **21**

[**Appendix A: Bill of Materials**](#_3jyf5glodjfe) **22**

[**Appendix B: System Parts List**](#_ngawddlk4j2c) **23**

[**Appendix C**](#_t3dyhv958cqt) **24**

# Overall Design

The Remote Lab System (RLS) has four essential components: the motherboard, experimental daughterboards, system enclosure, and dedicated workstation provided by UWB.

The RLS is remotely connected to by students enrolled at the University of Washington using Remote Desktop. Once the student is connected to the campus workstation attached experiments can be run using the RLS GUI. An system overview can be seen in Figure 1.

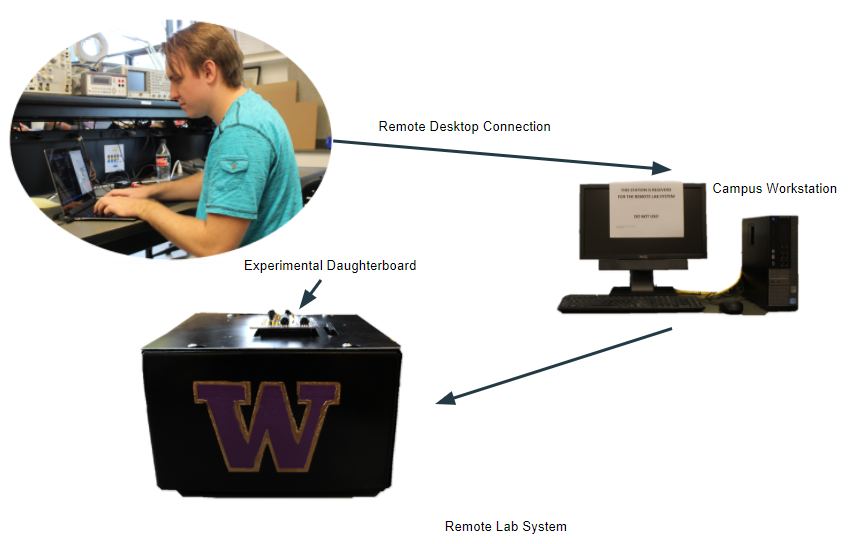


Figure 1: Overall system layout the Remote Lab System.

Contained within the system enclosure is a USB hub that connects an Arduino Mega 2560, two Intronix logic analyzers, and the Velleman oscilloscope and function generator to the campus workstation. Note: The Arduino Mega 2560 does not obtain power from the USB hub. Experimental boards plug into the RLS using 4 40-pin connectors on the top of the system enclosure.

# 

# Hardware Design

A parts list can be found in Appendix A.

The Remote Lab System is enclosed in a galvanized chassis with a 80mm 2-pin PC fan mounted to the back of the enclosure. There are six connectors visible on the exterior of the chassis: one DIN 5-pin power supply connector, one USB connector, and 4 daughterboard connectors.

The system is powered by a switching mode power supply. This power supply has a DIN 5-pin connector supplying +5V at 4 Amps, -15V at 0.5 Amps, +15V at 1.5 Amps. Each of these pins are protected using a fuse on the motherboard corresponding to their rated voltages. Warning: It is possible to exceed the current limitation of the power supply if multiple linear regulators are used with a low resistance load (less than 1 kΩ). A maximum of one device should be in this mode at any given time.

The motherboard has 4 heat sinks aiding in the cooling of the LT3015 and LT1963 linear regulators. These heat sinks are partnered with an 80mm 5V fan to keep the devices cooled when in high current low voltage output mode.

Mounted to the chassis is a 5V chassis fan.

## Hardware Operation

Components of the motherboard are controlled using an Arduino Mega 2560. Supporting code has been written to control the digital potentiometers allowing users to change the linear regulator outputs. Additionally, supporting code has been written to allow users to control the oscilloscope probe gain. For specific usage of any supporting code, see the **Software Operation** section.

### Logic Analyzer

The logic analyzers implemented on the motherboard were designed by Intronix. The motherboard simply acts as a mount for these devices and passes the input pins through to the daughterboard connectors. In order to use the two logic analyzers together for a total of 68-channels, the daughterboard must provide a common clock and trigger for both devices. This allows the two logic analyzers to be used independently if desired. A user guide provided by Intronix for using multiple logic analyzers can be found in Appendix C. Additional support for the Intronix logic analyzers can also be found on their website.

### Oscilloscope and Function Generator

The oscilloscope and function generator implemented in the motherboard was designed by Velleman. The motherboard has connector pins allowing the channels of the oscilloscope and function generator to be mechanically fastened and routed to the daughterboard connectors. The scope probe gain adjustment circuit can be seen in Figure 3.

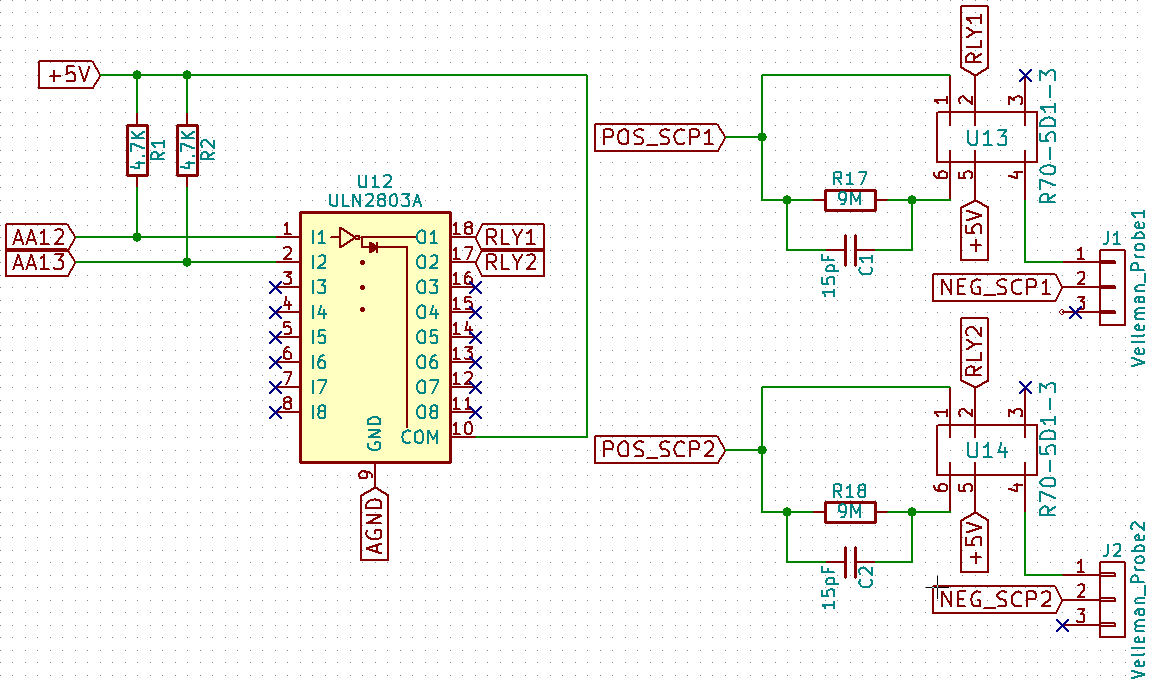


Figure 2: Scope probe gain schematic.

### Voltmeter

Implemented on the motherboard is an Adafruit ADS1115 4-channel 16-bit ADC. In single ended mode the ADC measures node voltages with respect to ground. The channels operate as follows:

* A0 valid for inputs 0V to +5 Volts
* A1 valid for inputs 0V to +15 Volts
* A2 valid for inputs -5V to 0 Volts
* A3 valid for inputs -15V to 0 Volts

Channels A1 and A3 operate using a voltage division circuit with a ratio of 3.

The voltage divider circuit reduces the voltage seen by the input protecting op-amp by a factor of 3. This allows an experimental board to measure voltages up to 15 volts. After a measurement is taken by the ADC, it is then multiplied by the voltage division ratio in order to obtain the actual value and then displayed to the user on the GUI.

The ADS1115 supports only positive voltages on the input pins in single ended mode so an inverting op-amp on the negative input pins of the ADS1115 (A2 and A3). Before the input voltage is inverted it also goes through a protection buffer with unity gain. The positive channels of the ADS1115 (A0 and A1) has an op-amp configured for unity gain as input protection.

Additional input protection for the ADC comes in the form of Schottky diodes. These diodes remove voltages greater than +5 volts and less than -5 volts. A circuit diagram of the inputs to the ADC can be seen in Figure 4, 5, 6, and 7.

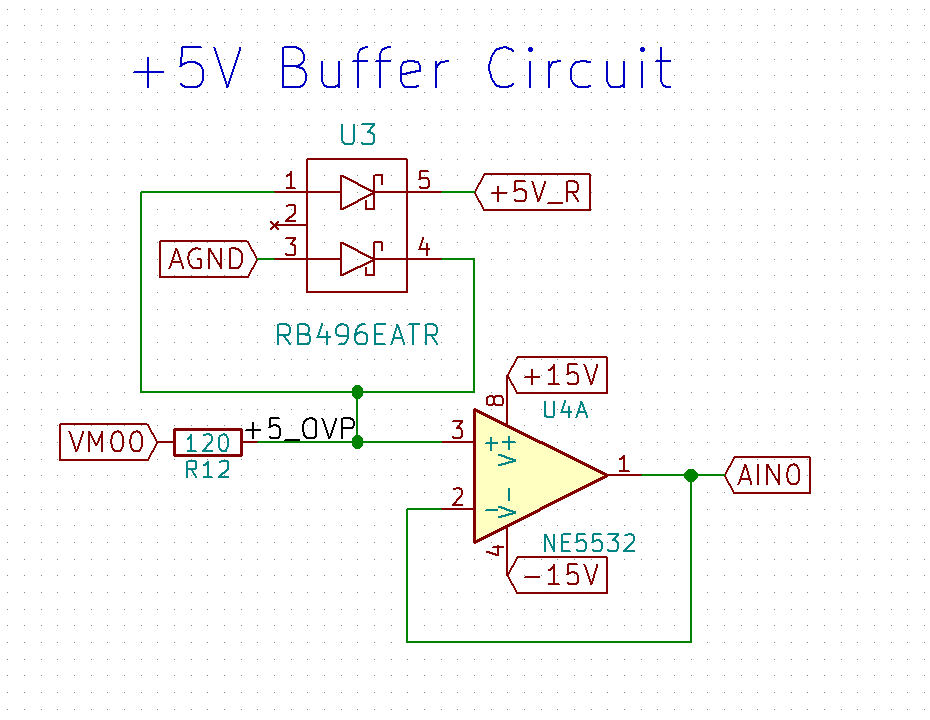


Figure 3: Positive 5 Volt buffer circuit schematic.

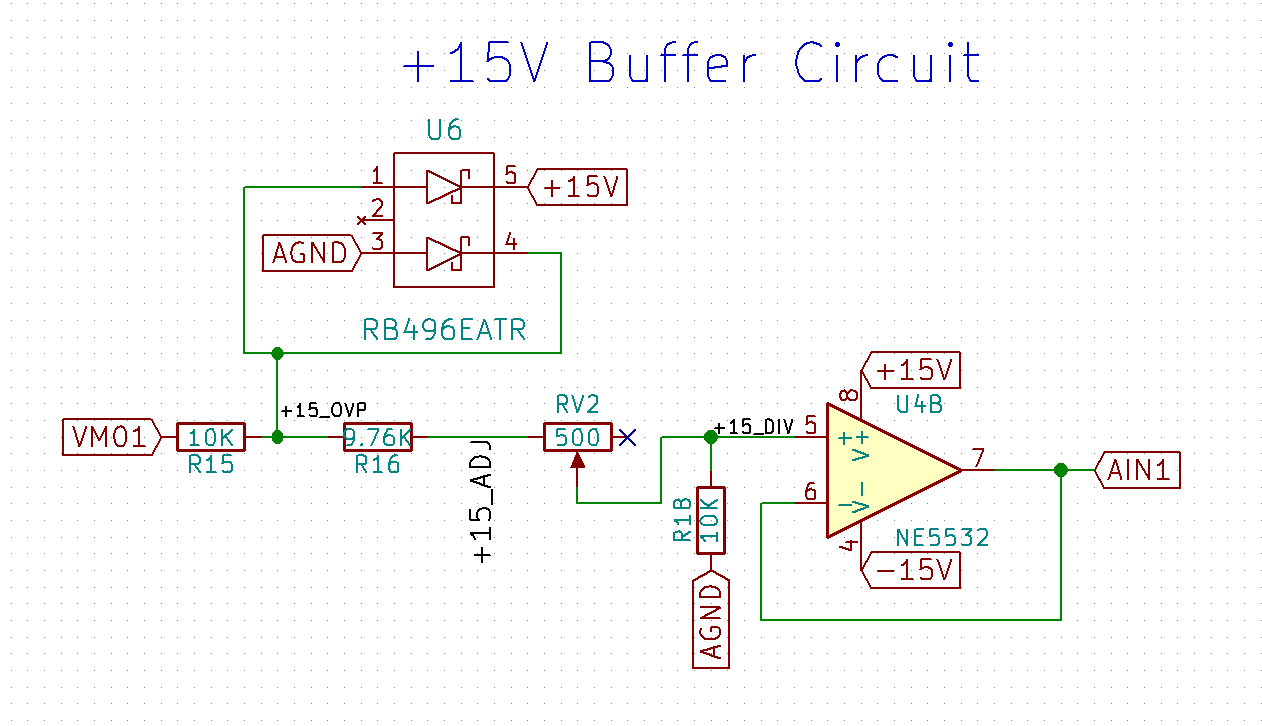


Figure 4: Positive 15 Volt voltage divider and buffer circuit schematic.

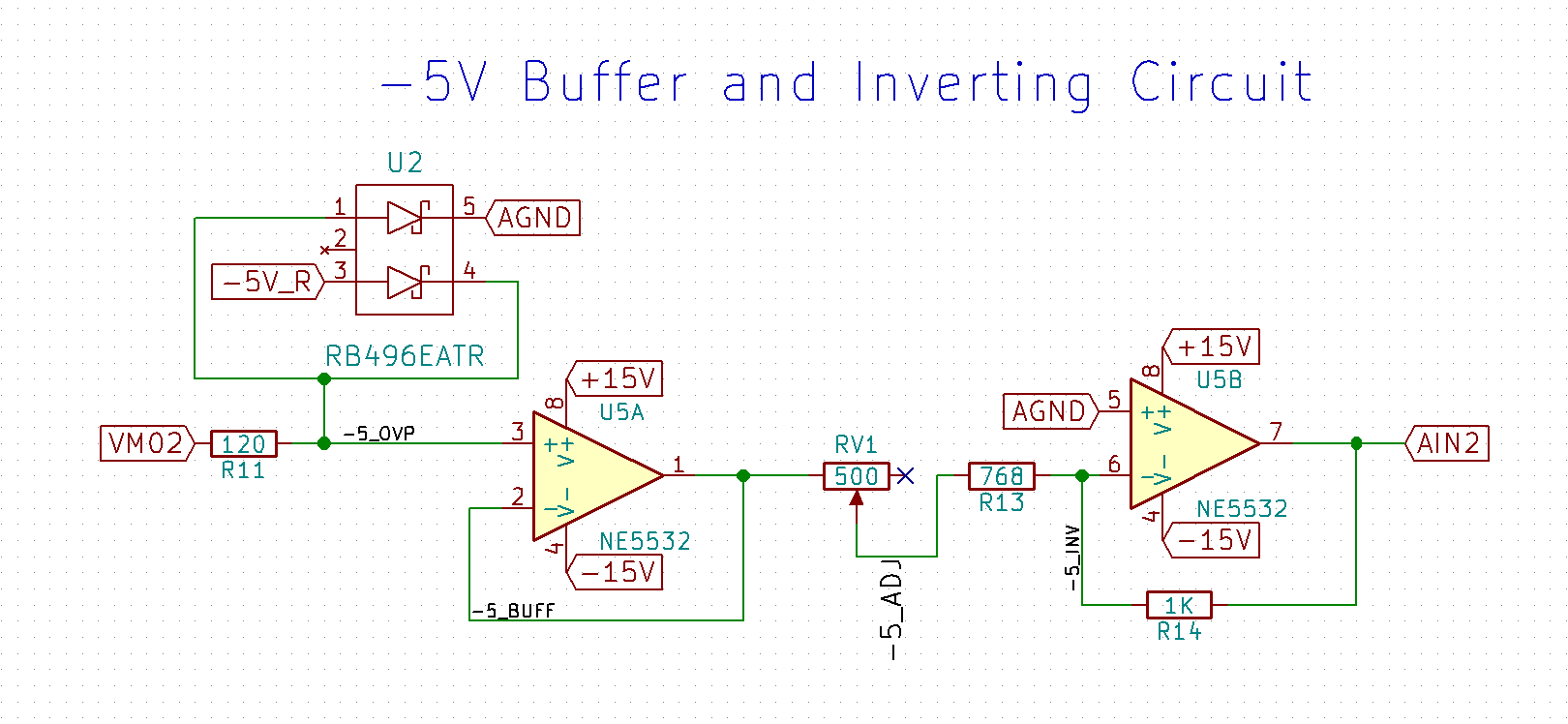


Figure 5: Negative 5 Volt inverter and buffer circuit schematic.

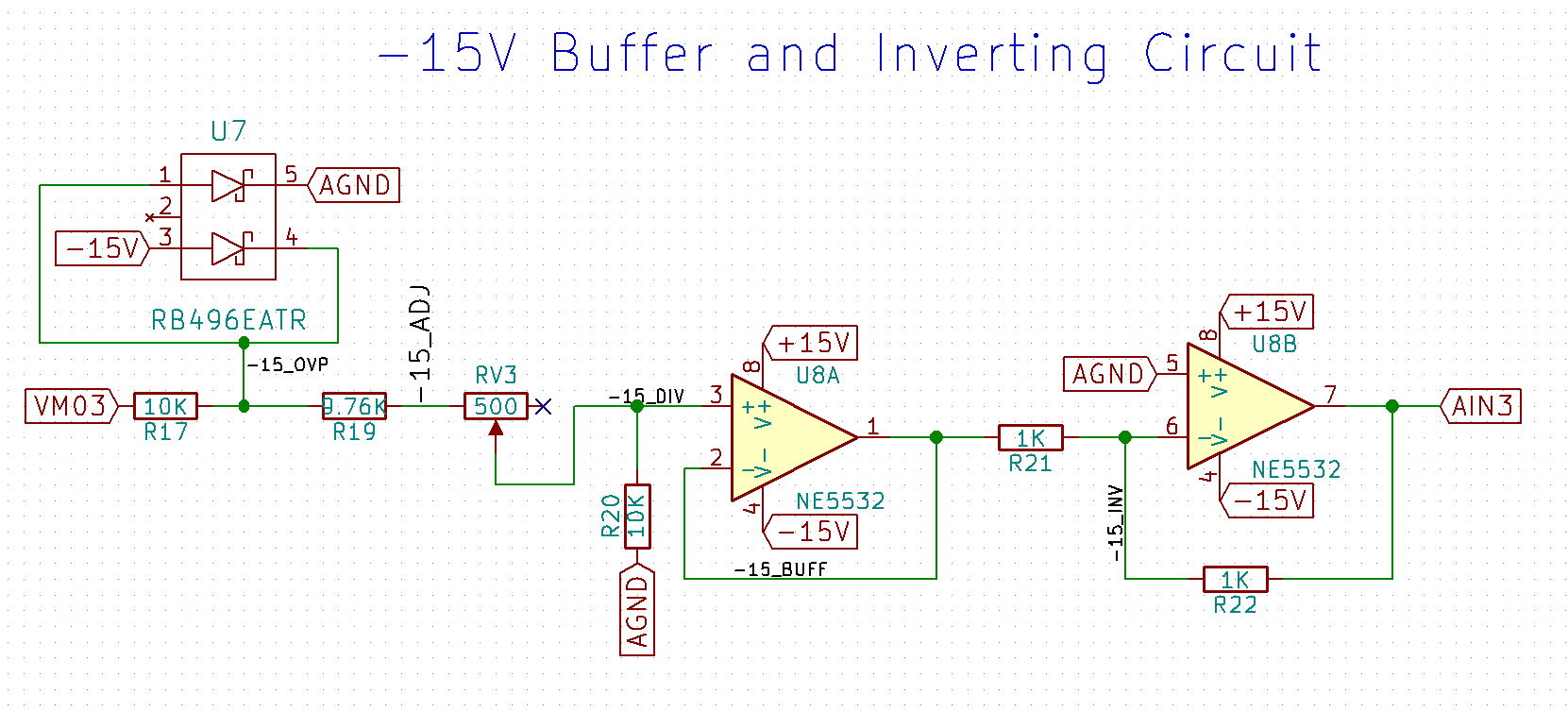


Figure 6: Negative 15 Volts voltage divider, inverter, and buffer circuit schematic.

### Linear Regulators

The motherboard contains 4 linear regulators: 2 negative regulators (LT3015) and 2 positive regulators (LT1963). A basic schematic, as taken from the datasheets, allowing adjustable operation can be seen in Figure 8 and Figure 9. In place of R2, our implementation has a 20kΩ digital potentiometer (AD5293). The output voltage from the linear regulators is based on the ratio of R1 and R2. With R1 at 1.78kΩ the entire output range of the linear regulators matches the full range of the digital potentiometer.

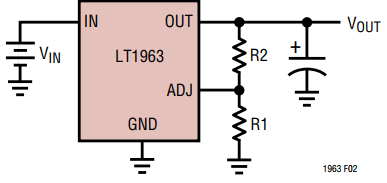


Figure 7: LT1963 with adjustable output as taken from datasheet.

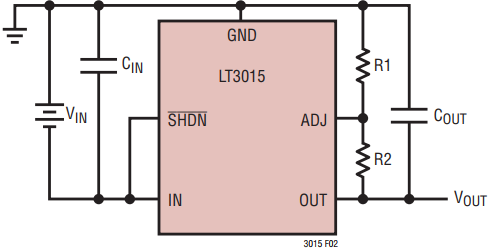


Figure 8: LT3015 with adjustable output as taken from datasheet.

### AD5293 Digital Potentiometers

Four AD5293 20kΩ digital potentiometers, one for each linear regulator, are used on the motherboard and control linear regulator’s output. The potentiometers are controlled using SPI communication (SPI Mode 1) from the Arduino. The AD5293 has a daisy chaining ability used in the motherboard design, one pair for each linear regulator pair (ie: one daisy chain for both LT3015’s and one daisy chain for both LT1963’s). This requires a 32-bit SPI transfer to change the resistance value.

### Heat Dissipation

The linear regulators are capable of supplying +1.21 Volts from the positive regulator, and -1.22 Volts from the negative regulator. If the attached load is low impedance, high current and high power is generated from the regulator. This heat is dissipated using the heat sinks and fan. Note: Only one linear regulator may be in this low voltage, low output impedance mode at any given time.

The heat sinks used on this motherboard are manufactured by Ohmite and have feet that need to be accounted for in PCB design. These feet should not be connected to any trace and should be isolated.

An 80mm fan is also included in the chassis for the system to force airflow through the heat sink fins. This fan circuit is powered by a MOSFET using a thermistor. The fan circuit diagram can be seen in Figure 10.

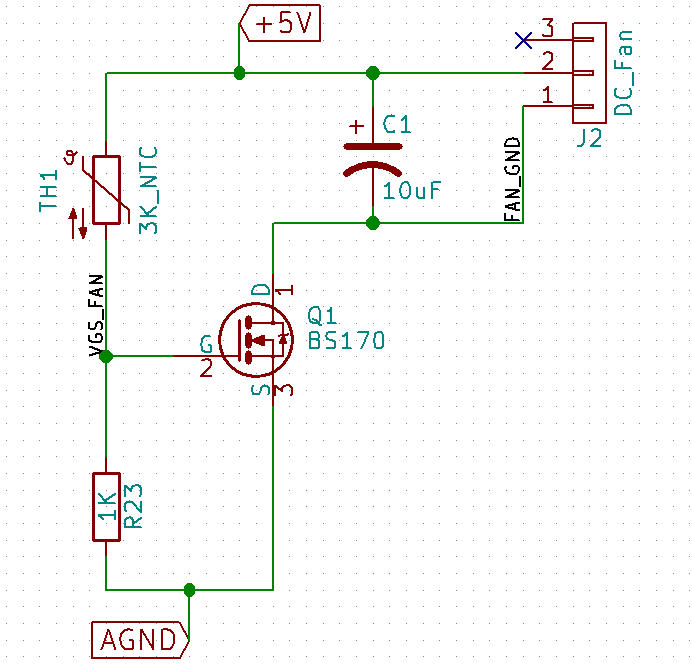


Figure 9: Fan circuit diagram.

# Software Design

Students will be able to connect to the RLS workstation via windows remote access to the ip: 128.208.255.52. A user may connect on campus or off campus using the Husky OnNet VPN. One user may to connect to the workstation at a time. On the desktop of the RLS workstation there is an executable file labeled “Remote Lab System”. This opens the RLS graphical user interface (GUI).

### Graphical User Interface (GUI)

The GUI is designed and programmed using Windows Forms C#. The GUI is used to communicate with the Arduino using UART serial communication. Through the GUI, the user is able to check which daughterboard is connected, change settings via drop down menus, send the updated settings to the Arduino, open logic analyzer software, and open oscilloscope software.

On startup, the COM port where the RLS system is connected is found and set. This allows the GUI to communicate with the Arduino at the request of the user without the user having to manually find the COM port that the RLS is connected to. This is done by checking all active COM ports and picking out the one the Arduino is connected to.

When the RLS software is first opened, daughterboard settings menus are deactivated using a deactivate function. The only buttons available for interaction are “Check Board”, “Logic Analyzer”, and “Logic Analyzer 2”. This prevents a user from changing and applying settings when no daughterboard, or a different daughterboard is connected.

A user must click the “Check Board” before any other interaction is possible. If no board is detected or the user has not checked board, then the user will not have permissions to changing any of the settings for any of the boards. Settings may only be changed for a board that is active. Check board works by sending “boardID” to the Arduino via the UART protocol. The Arduino then sends back the ID of the daughterboard currently connected. Based on the board connected, the panel for the test for that specific daughterboard is then enabled.

The “Update Board” button first verifies the daughterboard the user is attempting to update is still connected to the Remote Lab System. If it is not, the GUI prompts the user to press “Check Board” again. If the correct daughterboard is still connected, “Update Board” compiles the user’s settings into a String based on the state of the drop down menus and bubble options. After the String is created it is transmitted to the Arduino using Universal Asynchronous Receiver/Transmitter (UART).

Window resizing including maximizing the screen is possible though the experiment window does not scale with window size. This is due to previously designed experiment windows.

### External Software

The buttons “Oscilloscope/Function Generator”, “Logic Analyzer”, and “Logic Analyzer 2” open their respective software. The software is opened via hard coded file paths. This software is provided by the manufacturer and is used to take measurements. A user is not allowed to open more than one process of each. If a user attempts to open one that is already open, they are told an instance is already running.

On exit, the software for the logic analyzers and oscilloscope/function generator are also all closed. The windows are closed using the process.kill() command provided in C#.

### Arduino

On power on or after reset, the Arduino reads the boardID of a connected daughterboard, shuts down the linear regulators, and executes supporting daughterboard code (if one is connected). After it performs these setup functions the Arduino waits for data to be received from the GUI. The GUI sends data in the form of a String such as boardID, or a daughterboard settings String to be interpreted by supporting daughterboard code.

#### readBoardID()

If the String sent by the GUI is equivalent to “boardid” the Arduino reads the boardID pins (configured to be pins A0 - A7 on the Arduino) using digitalRead(). The boardID pins will read 1111\_1111 if no daughterboard is connected (ie: the boardID pins are tied high). The read 8-bit boardID is converted to a string and is sent to the GUI using Universal Asynchronous Receiver/Transmitter (UART).

#### Linear Regulator Shutdown

The linear regulators have a logical shutdown pin that is toggled using the Arduino. The shutdown is commonly triggered when changing daughterboards or when no board is detected.

#### Safety Check

The purpose of the safety check function is to constantly check the state of the boardID pins. If the boardID pins change states the linear regulators are shut down.

#### Daughterboard

Each daughterboard has a supporting C++ class that configures the Arduino’s digital pins to its own unique specification. This class is called upon when updating the board using the “Update Board” button in the GUI. The GUI sends a String via UART containing information passed onto the class.

In order to determine which daughterboard class to call, the Arduino first creates a substring to determine which board it is receiving information for. The daughterboard class then further dissects the String sent by the GUI to set digital pins on the Arduino.

#### Linear Voltage Regulator Control

A daughterboard can manipulate a total of four DC voltage sources, two negative and two positive. The Arduino adjusts the voltages of the regulators by manipulating digital potentiometers attached to each regulator. Voltages are set using a constructor with the desired voltage(s) taken as parameters. The constructor takes the desired output voltage value and calculates the required digital potentiometer resistance using one of the equations below.

The calculation used for setting the positive regulator values can be seen in Equation 1.

The calculation used for setting the negative regulator values can be seen in Equation 2.

Equation 1:

Equation 2:

#### Digital Potentiometer Control

The digital potentiometer acts as a controllable resistor to adjust the linear voltage regulators. In order for the digital potentiometer to set its wiper to the calculated resistance value, the resistance must first be digitally converted into 20 Ω increments to complement the 1024-position resolution of the AD5293 digital potentiometers. This was accomplished using Equation 3.

Equation 3:

Once the required digital resistance is calculated, the wipers for the digital potentiometers are enabled and they are sent a complete 16-bit data-word using SPI. This 16-bit word is composed of two unused bits followed by four command bits that control the operation of the digital potentiometer and finally the 12 data bits which contain the calculated digital resistance position that the digital potentiometer wiper needs to be set to. As the digital potentiometers are daisy chained, the data needed to be sent twice (for a total of 32 bits being sent) in order to reach both potentiometers. When the data has been sent and the potentiometers are set, the output voltage of the linear regulators are subsequently changed once they are turned back on.

# Test Plan

The test plan is broken down into two separate categories, Hardware and Software.

## Hardware

The hardware section is split into testing phases. The individual and independent component phase is testing that is done solely using a single component at a time. The small assembly testing phase is testing done incorporating small components together and testing their connected usages. Final assembly testing is performed after the entire PCB is assembled.

### Individual and Independent Component Test Phase

The individual and independent component test phase refers to the testing of each individual device received. We tested each device for expected operation and usage. The goal of this testing phase is to learn and experiment with device operations.

#### LT1963 (Positive Linear Regulator)

* Tested in adjustable output mode example circuit from datasheet. (See Figure 8)
  + Observed output range from 1.21 Volts to 15 Volts using manually adjustable pot and 1.78kΩ 1% resistor.
    - Calculated expected output value using Equation 1.
* Observed high amp output (with 1Ω load resistor).
  + Noted in this mode risk of blowing power supply fuse.
  + Daughterboards should use caution when using the regulators in this mode. Power supply has 1.5 Amp rating. One device is capable of producing a >1 Amp draw alone.
* Tested power generation and heat dissipation with heatsinks
  + Noted temperatures uncooled >55°C
  + Device rated for 70°C
  + Determined air cooling necessary in order to obtain high amperage outputs.

#### LT3015 (Negative Linear Regulator)

* Tested in adjustable output mode using example circuit from datasheet (See Figure 9)
  + Observed output range from -1.22 Volts to -15 Volts using manually adjustable pot and 1.78kΩ 1% resistor.
    - Calculated expected output value using Equation 2.
  + Observed high amp output (with 1Ω load resistor).
    - Noted in this mode high risk of blowing power supply fuse.
    - Daughterboards should use caution when using the regulators in this mode. Power supply has 0.5 Amp rating. One device is capable of producing a >1 Amp draw alone.
* Tested power generation and heat dissipation with heatsinks
  + - Noted temperatures uncooled >55°C
    - Device rated for 70°C
    - Determined air cooling necessary when in high amp output mode.

#### ADS1115 16-bit ADC (on Adafruit breakout board)

* Tested single ended and differential input modes
  + Observed negative voltages unreadable in single ended mode.
  + Observed built in protection circuit functionality.
    - When applying voltages out of range of the ADC, reads max range. When pushed further past readings on other channels misbehave.
  + Observed higher gain states at lower input voltages for more accurate voltage measurement.
* Tested single ended mode with voltage divider, buffer, and inverting amplifier
  + Observed accurate reading with voltage divider and buffer circuits.
    - Compared Serial Monitor readings with multimeter
  + Observed accurate readings with inverted negative voltage inputs.
    - Compared Serial Monitor readings with multimeter
    - Will need to be inverted in the software for user view
* Tested auto gain adjustments in software
  + Used power supply and multimeter to observe resolution changes.

#### AD5293 (Digital Potentiometer)

* Tested basic functions of a single device
  + Device operates using SPI communication
    - SPI Mode 1
  + Observed reset operation
    - Noted on reset the wiper moves to half (10kΩ)
  + Observed resistance change based on hex values
    - Single device requires a 16-bit transfer
    - 10-bits are data (1023 possible wiper positions)
    - Each data bit accounts for ~20Ω
* Tested daisy chain capabilities
  + Requires 32-bit data transfer to write and update both devices
  + Observed resistance changes of both devices in series
    - Despite 32-bit transfer doesn’t appear slower

#### ULN2803A (Darlington Transistor Array)

* Tested outputs given 5 Volt inputs
  + Expected output currents large enough to drive inductive load
    - Able to toggle RA30421051 Relay

#### RA30421051

* Tested with DC power supply and pull up resistor
* Tested for continuity on switching poles
  + Expected switch to throw on 5 Volt logic

#### LM337 (-5 Volt Regulator)

* Tested with DC power supply with 15 Volt input
  + Expected consistent +5 Volt output

#### LM317 (+5 Volt Regulator)

* Tested with DC power supply with -15 Volt input
  + Expected consistent -5 Volt ouput

#### Motherboard

* Upon arrival performed visual inspection of quality
* Upon arrival measured resistance at all power nodes to all ground nodes.
  + Expected reading: Overload

### Small Assembly Test Phase

The small assembly test phase refers to the testing of multiple components tied together in a similar manner that will be used in the final implementation.

#### Linear Regulator Circuit

Two LT3015 and two LT1963 linear voltage regulators used in combination with 4 AD5293 digital potentiometers. For the two LT3015 negative linear voltage regulators two AD5293 digital potentiometers are daisy chained together.

* Tested voltage changes based on digital pot value changes
  + Observed full voltage output range (-15 Volts to -1.22 Volts)
* Tested software controller for changing voltage values
  + Observed correct operation based on function parameters
  + Function takes in a desired voltage and uses an algorithm to determine required digital potentiometer value

#### LM337 and LM317

* Tested to ensure expected output
  + Observed +15 Volt regulated down to +5 Volt output
    - Noted higher currents possible with lower impedance loads.
  + Observed -15 Volt regulated down to -5 Volt output
    - Noted higher currents possible with lower impedance loads.

#### Cooling Circuit

The cooling circuit operates using a BS170 N-Channel MOSFET, a thermistor, and a 5V 2-pin 80mm fan. The fan is implemented to encourage airflow along the heat sinks for the linear voltage regulators.

* Tested fan circuit with potentiometer to simulate thermistor
  + Expected to voltage drop across transistor to increase, thus turning on the fan
  + Observed fan operation

#### Chassis

* Tested connection fitment
* Tested daughterboard connector fitment against lid

### Final Assembly Test Phase

The final assembly test phase refers to testing after construction of the entire PCB (ie: all components have been soldered to the board). This testing is also performed after chassis modification has been completed.

* Tested motherboard spacer fitment on lid
  + Daughterboard connectors protrude from lid as expected
  + Reasonable connector clearance
  + Minimal flex experienced when attaching/detaching a daughterboard
* Tested connection to logic analyzer cards
  + Logic analyzers securely fastened to motherboard
  + Logic analyzer’s USB connector is accessible
* Tested Velleman connections
  + Mechanical probe connections are secure
  + Oscilloscope appropriately displays waveform
    - Waveform generated by Tektronix AFG3252 Waveform Generator
    - Waveform passed into scope via daughterboard oscilloscope probe pins
  + Function Generator produces acceptable waveform
    - Waveform measured by Tektronix MSO3012 Mixed Signal Oscilloscope
    - Waveform measured from daughterboard connector pin
* Tested USB hub connectivity with work station
  + Velleman is detected via USB hub
    - Velleman software receives hardware response
  + Logic analyzers are detected via USB hub
    - LogicPort software detects both devices independently
  + Arduino is detected via USB hub
    - Arduino detected by Arduino IDE
    - Communicates with GUI
* Test linear voltage regulator outputs
  + Positive regulators
    - Expected output ranges between 1.22 Volts and 15 Volts
    - Expected steady output at specified value
      * Measured using Keithley Multimeter
  + Negative regulators
    - Expected output ranges between -15 Volts and -1.23 Volts
    - Expected steady output at specified value
      * Measured using Keithley Multimeter
* Tested heat dissipation performance
  + Expected fan to power on when heat is detected
  + Expected device(s) to continue operating (ie: device does not enter shut down mode)
  + Device remains within operating temperature range
* Tested boardID detection
  + Expected 1111\_1111 reading with no daughterboard connected
  + Simulated boardID’s using Keithley DC power supply

## Software

The Remote Lab System Graphical User Interface (GUI) and Arduino software was tested on Windows 7 and Windows 10. We do not guarantee the software’s function on any platform.

Software testing occurs in multiple stages throughout the development process and was performed both as White Box testing and Black Box testing. White box testing was used for evaluating the performance of code used for controlling essential devices. Black box testing was used to ensure limitations on user access and control of the system.

### White Box Software Testing

* Tested collection of user settings
  + Expected GUI to generate a String with user settings
  + Expected GUI to transmit created string using UART to Arduino
* Tested user lockout methods
  + Expected user can only change values of the active board
  + Expected all drop down menus inaccessible when no board detected
  + Expected warning message if no board detected
  + Required successful boardID check before settings can be changed
  + Only single instances of applications allowed
* Tested application shutdown on GUI exit
  + Expected LogicPort and Velleman software to terminate on GUI exit
* Tested communication from GUI to Arduino
  + User settings on the experiments page are correctly sent to the Arduino
    - Arduino handles these UART signals and passes them correctly to the corresponding daughterboard experiment C++ file.
  + boardID request sent via UART to Arduino
    - Arduino responds with boardID read from analog pins
* Tested automatic detection of RLS Arduino
  + Expected automatic port setting for Serial communication
* Tested Arduino control of linear regulators
  + Uses daisy chained AD5293 digital potentiometers
  + Expected outputs from LT3015 negative linear regulators to be from -15 Volts to -1.22 Volts
    - Error returned on out-of-range inputs
  + Expected outputs from LT1963 negative linear regulators to be from 15 Volts to 1.21 Volts
    - Error returned on out-of-range inputs
* Tested ADS1115 (ADC) control
  + Expected less than 1% margin of error
    - Simulated using Keithley DC power supply
    - Compared readings with Keithley multimeter
  + Expected single ended single readings
* Tested Arduino safety functions
  + Arduino shuts down linear regulators on GUI exit
  + Arduino shuts down linear regulators when no boardID detected

### Black Box Software Testing

* Tested Graphical User Interface (GUI)
  + User is unable to change settings when daughterboard is not detected
  + User is only able to change settings on detected daughterboard screen
  + Software closes upon closing of the RLS GUI
  + User is only able one instance of each program
  + User can change detected daughterboard’s settings and observe correct output corresponding with the input settings

### 

# Project Schedule

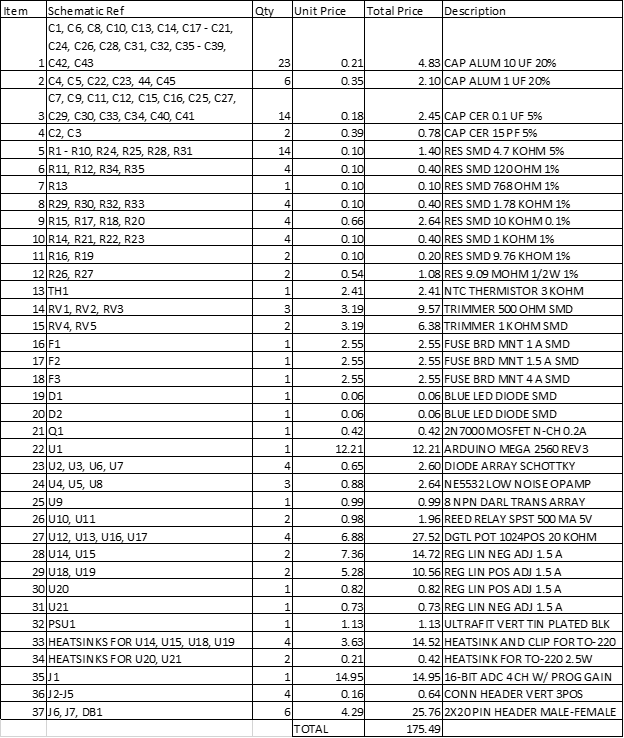
Schedule for Capstone 1:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Category | Milestone | Due Date |
| 1. | Documentation | Determine Parts to be Used | May 19th 2017 |
| 2. | Documentation | Final Contract | June 9th 2017 |
| 3. | Documentation | Final Project Specification | June 9th 2017 |

Schedule for Capstone 2:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Category | Milestone | Due Date |
| 1. | Hardware & Software | Demonstrate New Motherboard Design | July 28th 2017 |
| 2. | Hardware & Software | Integrate Daughterboards with Motherboard | July 28th 2017 |
| 3. | Documentation | Final Report | August 18th 2017 |
| 4. | Documentation | Final Presentation | August 18th 2017 |

# Appendix A: Bill of Materials



# Appendix B: System Parts List



# 

# 

# Appendix C

The following is copied from a .txt file provided by Intronix on the usage of multiple logic analyzers together.

|  |
| --- |
| As mentioned in the FAQs, it is recommended to use multiple units in state mode if your circuit can provide a common clock and trigger signal to all units. The common clock is what keeps the samples coherent in time, while the common trigger signal is what synchronizes the acquisitions of multiple units.  It is possible to use multiple units in timing mode, but keep in mind that each unit has its own independent clock oscillator. Since the clocks are not synchronized this means that at the trigger sample there will be an uncertainty of +/-1 sample period between units. This effectively reduces the timing resolution by 1/2 (500MHz only has resolution of 250MHz). With compression enabled, the situation can get even worse since it may be possible to compress large numbers of samples into the buffer. While the uncertainty at the trigger point is always +/-1 sample period, the clock oscillators have a worst case mismatch of 100ppm between multiple units. If you were able to compress 1,000,000 samples into the buffer, the signals sampled 1,000,000 sample periods before/after the trigger event could have a differential error of 100 sample periods between units! This is why we don't recommend using multiple units in timing mode. Often times you won't be able to capture (or even be interested in) 1,000,000 samples, so this last point isn't always an issue.  Here are a few tips on using multiple LogicPorts from one PC:  1. The LogicPort software will open the first LogicPort it finds. The first instance of the software you open will use the first LogicPort you connected to the PC. Once that instance is open, the second instance of the software will find and use the second LogicPort connected and so on.  2. It will be more convenient if you have separate installations of the LogicPort software so that each instance has it's own .INI file in it's respective install directory. The .INI file tells the software which project file to open when the application starts up (among other things). The default installation is in:  "C:\Program Files\LogicPort"  We recommend that you install the second instance in a directory such as:  "C:\Program Files\LogicPort2"  3. Here's the tricky part: The second installation will overwrite the desktop and start menu shortcuts causing them to point to the second installation only. You will need to create a new shortcut which points to the first installation. You should be able to just make a copy of the desktop shortcut and change the shortcut's target directory to point to the first installation.  Note that although there should be separate installations, you don't necessarily need to use separate directories for project or export files. For example, you could open a project file in the "\Projects" directory of the first installation using the second instance of the software.  We have run as many as 4 LogicPorts from the same PC simultaneously, and the theoretical limit is 127 units. |