SUNLESS SOLAR PANEL

A Design Project Report

Presented to the School of Electrical and Computer Engineering of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering, Electrical and Computer Engineering

Submitted By

Oleksandr Kuzura and Karthik Krishna Jayaram

MEng Field Advisor: Bruce Land

Degree Date: May 2019

Abstract

Master of Engineering Program

School of Electrical and Computer Engineering

Cornell University

Design Project Report

Project Title:

Sunless Solar Panel

Authors:

Oleksandr Kuzura and Karthik Krishna Jayaram

Abstract:

We created a power supply with the current and voltage characteristics of a thin film solar panel for testing solar-powered devices in bad weather and wintertime. The device is best implemented using a 100 W power supply but will also work with a 10 W supply. It steps supply voltage down using an Arduino-controlled power converter. The Arduino detects the device's current output, calculates the voltage a solar cell would output at that current, and adjusts output voltage to match that voltage. This control occurs through a servo loop. The sunless solar panel allows users to test solar-powered devices regardless of weather conditions.

Executive Summary

Solar panels provide optimal power output under full illumination. Their performance suffers in cloudy. Since the majority of Cornell University's school year runs during the cloudy season, students designing solar devices would benefit from a solar panel simulator, which would enable testing regardless of weather conditions, especially during cloudy Ithaca winters. However, such devices on the market cost thousands of dollars. We have designed a 10 W hardware simulator of a thin-film solar cell to achieve this goal.

We designed the system to meet the following requirements: output a minimum of 10 W with voltage and current characteristics mimicking those of a solar panel; be safe for operation by untrained users; be upgradable to output 100 W; be upgradable to include settable controls simulating ambient temperature and illumination conditions; be upgradable to display voltage and current output; cost at least ten times less than commercially available solar simulators; be buildable within a school year.

We chose to implement the device with digital control instead of using a purely analog circuit. A survey of analog photovoltaic simulator implementations showed that it is difficult to scale up power output and implement precise I-V characteristic controls in such systems. On the other hand, complicated analog systems with sophisticated analog control mechanisms would take longer than one academic year to implement and still struggle to output high power. The ease of programming control loops using microcontrollers sold us on a digital design implementation.

Our device uses power electronics to step down voltage from a power supply. A current sensor detects the output current, and appropriate circuitry filters and amplifies this current signal before it is read by an Arduino microcontroller. The Arduino uses the solar cell equation to calculate the output voltage of a solar panel based on the measured current. The Arduino then adjusts the power converter output to match a solar cell's voltage and repeats the control loop.

Our device successfully met the stated objectives. It safely outputs 10 watts, is inexpensive, and can easily be upgraded to output 100 W and voltage and current output readouts.

The authors contributed equally in carrying out this design project and writing the design report.

We dedicate this report to our grandfathers, who never fail to inspire, encourage, and teach us to be better.

Design Problem

Solar-powered devices must be tested with solar panels. However, solar panel performance varies with insolation and temperature, and drops sharply in cloudy conditions. Unfortunately, Ithaca and the majority of the Northeastern and Northwestern US are notorious for their dreary, gloomy, cloudy winter days. These conditions drop solar panel outputs to near-nil for much of the year. And low solar panel outputs, in turn, hinder testing solar-powered devices while testing engineers wait for an optimally sunny day to test their wares. This problem is especially frustrating in Cornell's engineering school, which runs during the nine cloudiest months of the year. That students cannot create and follow reliable testing timelines only gives stressed students more stress. Academic mentors can only recommend patience while students wait to get a stable, full-power solar panel output.

This project solves the problem of testing solar-powered devices in Ithaca's cloudy conditions. We designed a hardware solar panel simulator to enable testing of solar-powered devices by any student at any time.

We created the simulator to:

- Have output current and voltage (I-V) characteristics of a solar panel.
- Output at least 10 W.
- Be safe for inexperienced users.
- Be at least ten times cheaper than commercial alternatives.
- Be upgradable to a 100 W system with voltage and current readouts and controls to simulate changes in ambient temperature and light.

Design Alternatives

The ideal problem solution would be using solar panels. However, solar panel performance in Ithaca is unpredictable. Ithaca ranks below the US average on the annual number of sunny days. And even on a sunny days, clouds can suddenly obscure the sun and disrupt a running experiment. Further, changes in ambient temperature alter solar panels' I-V characteristics. These factors make for hourly, daily, and seasonal variations in solar panel performance. Therefore, testing with solar panels is a suboptimal solution to the current design problem.

Commercially available solar panel simulators are another solution to testing solar devices in winter. These devices, while reliable, are extremely expensive. A 500 W Keysight solar array simulator costs over $6,000 [5]. This price is too steep, even though the device our design requirements.

While using real solar panels and commercial simulators is not feasible, building a custom solar panel simulator certainly is. These simulators come with two flavors of control system: analog and digital. Among analog options, the simulator described by Blanes and Garrigos directly simulates an ideal solar cell equivalent circuit by creating a transistor current source in parallel with an array of diodes (Fig. 1) [4]. Controlling the current source output corresponds to controlling an irradiance parameter, while altering the parallel diode array controls simulated temperature parameters. According to the authors, the circuit outputs up to 2 A using its transistor amplifiers; the R2 potentiometer controls the simulated short-circuit current. The diodes can be short circuited in groups of 10 or less to alter open circuit voltage parameter.

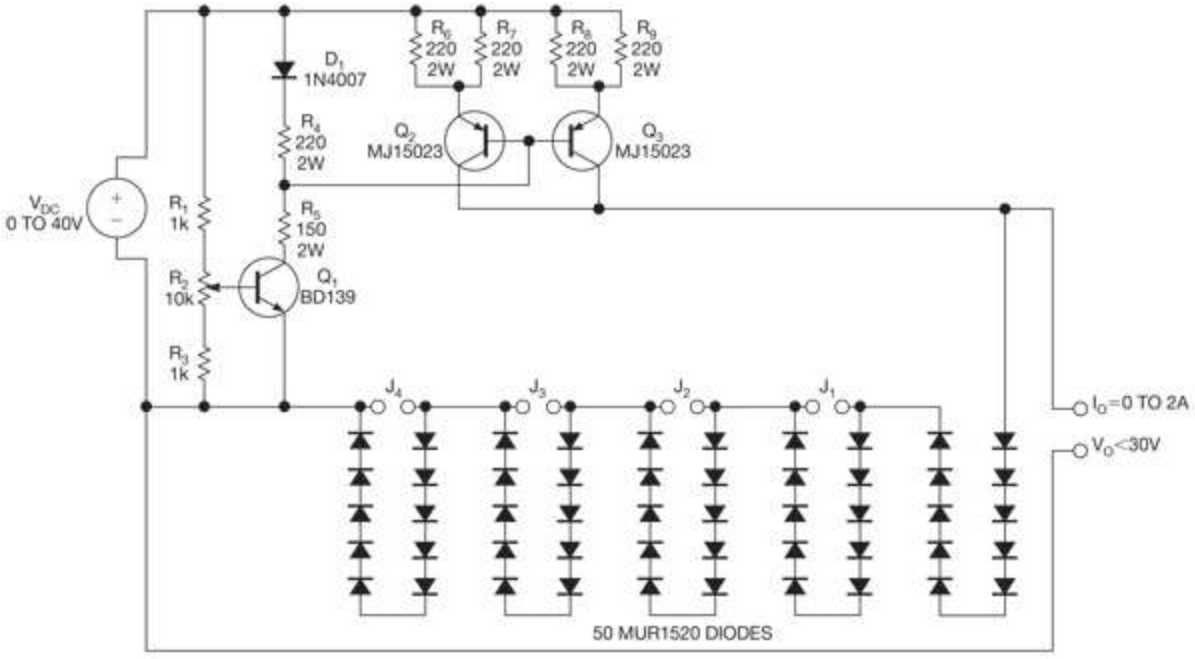


Figure 1. Circuit diagram of Blanes and Garrigos' transistor-and-diode photovoltaic simulator.

However, the open circuit voltage control is discretized into steps equal to the diode forward voltages, which are gross for our control requirements. Further, the 2 A maximum output current is enough for a 10 W solar panel simulator, but is short of the approximately 5 A needed for a 100 W output. Finally, detecting and displaying voltage and current output levels from this circuit would be difficult without adding digital functionality. This circuit therefore lacks the granularity of control and scalability options an ideal design would have. It was not suitable for our purposes.

A design proposed by Miric and Nedeljkovic similarly used transistor current amplifiers to create a current-controlled photovoltaic simulator [7].

Their device employed nested analog control loops in which variable resistors simulate temperature and irradiance characteristics, and op amp resistor feedback networks set amplification parameters.

This design was simple, inexpensive, and used no digital memory or processor. However, it did not allow precise control of temperature and irradiance parameters because variable resistors could not be programmed. This design was also limited in its power output by the transistor amplifiers' current output capabilities. The author's experimental data confirmed these limitations by indicating that the design's open circuit voltage and short circuit current characteristics were lower than our design's requirements. Finally, this device would not have been user-friendly because of the analog control.

These analog hardware photovoltaic simulators illustrate the main drawbacks of analog devices: difficulty of control and use. Furthermore, these two designs also had limited output power levels and for upgrade. We chose to construct a digitally controlled system to overcome these obstacles. Digital control allowed finer control over simulation properties than an analog circuit would. It also left room for easier system upgrades because components with higher performance specifications can be installed in the system, keeping the same control scheme. Upgrading analog systems, on the other hand, would demand rebuilding the control loops and making other large system modifications.

Device Design

On a high level, our device consists of a power electronics stage, which steps input DC voltage down; a current sensing stage, which feeds a current signal into the control loop; and a control loop, which adjusts the output voltage (Fig. 2). This voltage adjustment is based on current readings, in accordance with the solar panel equation,

$$\mathrm{I} = \mathrm{I_L} - I_0\left[\exp\left(\frac{V + R_s I}{nV_t}\right) - 1\right] - \frac{V + R_s I}{R_{sh}} \quad (1)$$

Where I is the photovoltaic output current, I_L is the device illumination current, I_0 is the diode saturation current, V is the output voltage, R_s is the series resistance of the equivalent photovoltaic circuit model, R_{sh} is the shunt resistance from the equivalent circuit model, n is the diode ideality factor, and V_t is the thermal voltage.

The system's power stage is powered by a 20 V DC power supply, while the logic circuit is powered from 5 V voltage regulator's output. Both supplies

are decoupled with 100 uF and 100 nF capacitors. The full system schematic can be found in Appendix 1.

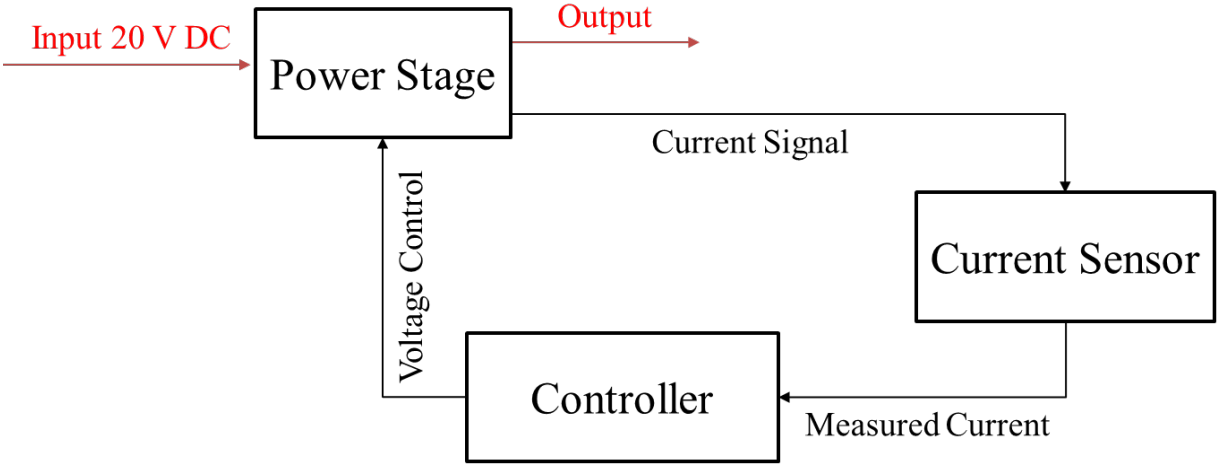


Figure 2. Block diagram of the solar simulator.

Power Stage

The power stage consists of the LM2575 power converter and its external components (Fig. 3).

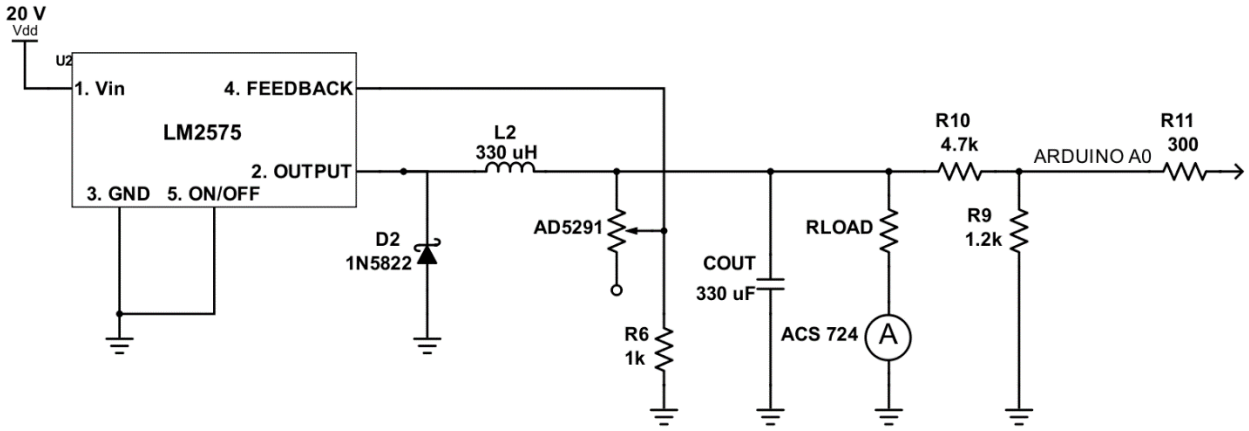


Figure 3. The power stage consists of the LM2575 DC-DC power converter and external components. Most important of these is the AD5291 digital potentiometer, which adjusts the output voltage.

The LM2575 steps a 20 V DC power supply input down to a desired output level. This power converter's adjustable output of up to 37 V at 1 A made it suitable for our 10 W Sunless Solar Panel application because 10 W solar panels output up to 20 V and 1 A [3]. We chose to use a buck (step down) converter topology instead of a boost (step up) topology because the LM2575's minimum output voltage (1.23 V) was lower than that of a similar

boost converter's (3 V) [8]. The Schottky diode (D1) and 330 uH inductor (L1) enable the switched-mode voltage conversion [9]. D1, the 1N5822 Schottky diode, is amply robust for our high current application because it can block a maximum 40 V and conduct a maximum 3 A, well above maximum expected outputs of 20 V and 1.2 A [6]. We sized L1 according to the instructions in the LM2575 datasheet using the volt-microsecond constant and the calculated maximum current [6]. A Murata 60B334C 220 uH inductor with a 4 A maximum current rating was sufficient for this application. Finally, we sized the output capacitor C1 per the datasheet procedure [6] using the equation

$$C_{OUT} \geq 7.785 \frac{V_{in\,(max)}}{V_{out} * L} \tag{2}$$

The power converter's output passed through a voltage divider with a 0.2 division ratio to create an output suitable for the Arduino, which accepts 0-5 V input voltages. The resistors were sized on the order of 1 kOhm to avoid interference with the Arduino's analog-to-digital converter (ADC) with too high an impedance. A 300 Ohm current limiting resistor ensured the scaled voltage output signal didn't damage the Arduino input pins. Similarly, an ACS724 current sensor detected the converter's output current passing through variable load resistors on the output. More on the current detection in later subsections.

The converter's output voltage is determined by the ratio of the AD5291 digital potentiometer to R6 according to the equation

$$V_{out} = 1.23\left(1 + \frac{R_{AD5291}}{R6}\right) \tag{3}$$

We sized R6 as 1 kOhm because the LM2575 datasheet recommended it be sized between 1 kOhm and 5 kOhm. Using 1 kOhm would give the maximal range of output voltages [6]. We included the AD5291 digital potentiometer (digipot) acting as a rheostat in place of a regular resistor to enable real-time output voltage control from the power converter (Fig. 4). Using the digipot as a rheostat allowed keeping R6 a constant 1 kOhm. Implementing the digipot as a potentiometer with one terminal taking the place of R6 would have complicated output voltage control, since both the numerator and denominator in equation (3) would change with each adjustment of the digipot wiper position.

We chose the AD5291 chip because it allowed changing the resistance from 0 to 20 kOhm with a 256-position resolution [2]. This range of resistances provided for $V_{out,min} = 1.23\,V$ and $V_{out,max} = 25.83\,V$ which was, of

course, be cropped to slightly below the 20 V supply voltage. The 256-position resolution implied a 78.125 Ohm change in resistance with each setting of the wiper position, which translated to a 0.096 V output voltage change with each wiper setting. Such control was vital to accurate output voltage adjustment in tracing a solar cell's I-V curve. Further, the AD5291 was one of the few digital potentiometer chips on the market which accepted with a supply voltage above 20 V [2].

The digipot chip held two circuits: a digipot circuit (pins 1-7) and a digital logic circuit (pins 8-14) [2]. The digipot circuit included supply, ground, filtering capacitor, and reset pins (Fig. 4). Pins 3-5 were the potentiometer terminals. Since we used the digipot in rheostat mode, we left the Resistor A (pin 3) open and connected the wiper (pin 4) and Resistor B (pin 5) pins as in Figure 3.

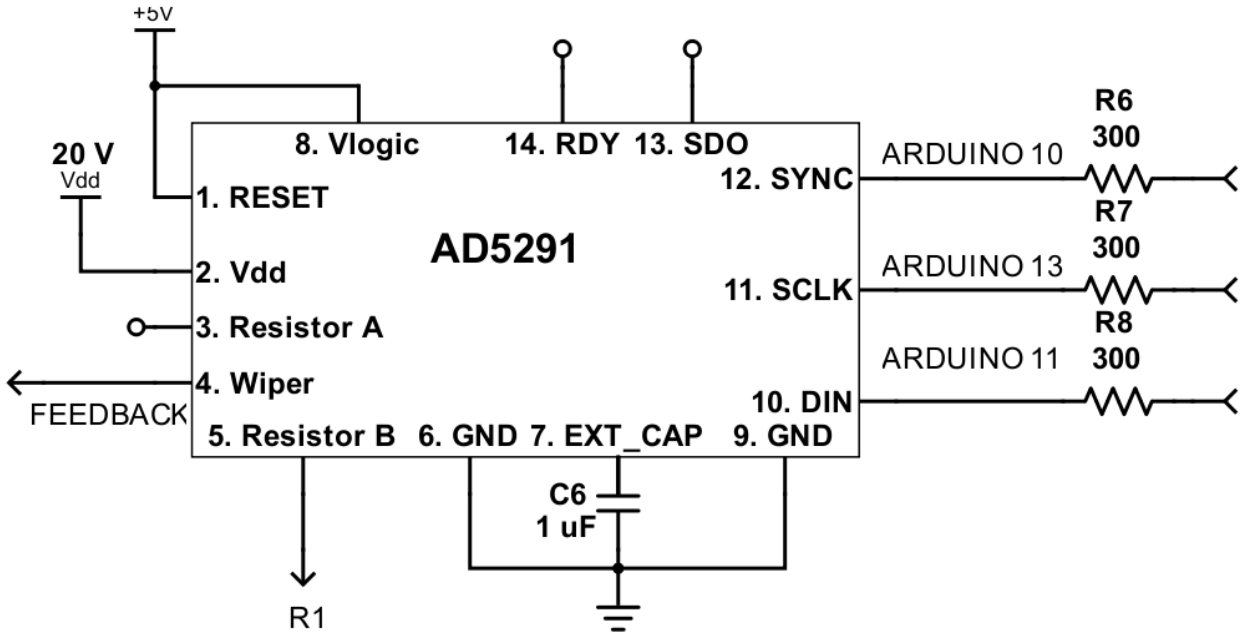


Figure 4. Wiring diagram of the AD521 digital potentiometer.

The AD5291 resistance was controlled through SPI communication coming into pins 10-12 [2]. Pin 10 was the data in pin and connected to Arduino's hard-wired SPI data pin 11. The synchronous clock input on pin 11 connected to Arduino's SCLK pin 13. Finally, the slave-select input pin 12 received input from Arduino pin 10, which we programmed to give a low signal to enable writing data commands to the AD5291 data register. We didn't use pin 13 because we had no use for reading back digipot settings. However, this data out pin would be useful for diagnosing whether dysfunctional potentiometers were fried on the digital logic side or on the potentiometer

side. Similarly, we left pin 14 open because it was only involved in two-way communication with the digipot.

The Arduino controlled the digipot by passing a write enable command on startup, then passing 16-bit wiper commands consisting of `[0 0 0 0 0 1 D9 D8]` on the most significant byte and `[D7 D6 D5 D4 D3 D2 X X]` on the least significant byte [2]. The six most significant bits initiated a write operation to record wiper settings in the digipot registers, while the ten least significant bits were wiper-setting serial data from the Arduino. See Appendix 2 for complete Arduino control code. The AD5291 receives the two least significant bits as "don't care" bits because it has a 256-bit resolution. The AD5292 chip reads these bits because it is has a 1024-bit resolution. We chose the AD5291 because the extra fourfold increase in resolution with the AD5292 was unnecessarily fine.

Current Sensing Stage

The current sensing stage (Fig. 5) linked the power converter with the circuit's control loop. An ACS724 Hall Effect current sensor detected the LM2575 output current being fed through a load resistor. We chose the ACS724 for its high precision, simple operating principle, and extremely low 1.2 mOhm resistance [1]. The current sensor output a 200 mV / A output signal on top of a 2.5 V zero-current signal [1].

The ACS724's maximum output reached 200 mV when the LM2575 output its maximum 1 A current. This signal was too weak for the Arduino ADC to resolve any significant features and was also corrupted by electrical noise. To clean and strengthen the signal, we implemented a filter and amplifier circuit (Fig. 5) using MCP6242 rail-to-rail op amps powered with 5 V on the positive side and ground on the negative side. Powering the op amps as such ensured that their output voltage fell within the 0-5 V range suitable for input into the Arduino's analog input pins.

We implemented the low pass filter using a 100 kOhm resistor and 100 nF capacitor for a cutoff frequency of 10 Hz, which was much lower than anticipated noise. We chose to implement an active filter instead of a passive filter to preserve signal strength and take advantage of the op amp's low output impedance when outputting a signal to the differential amplifier. A nontrivial output impedance would complicate designing the differential amplifier because the latter relies on precise resistor matching to achieve consistent voltage subtraction across multiple input voltage ranges.

Having attenuated the noise in the current signal, we needed to amplify it. Given that the Arduino accepts input in the range 0-5 V, we could amplify

our output current, which was up to 1 A, by 25. We achieved this amplification as shown in Figure 5, using 20 kOhm and 500 kOhm resistors in the amplifier feedback loops. However, the constant 2.5 V zero-current offset limited the possible signal amplification because the op amps output a maximum of 5 V.

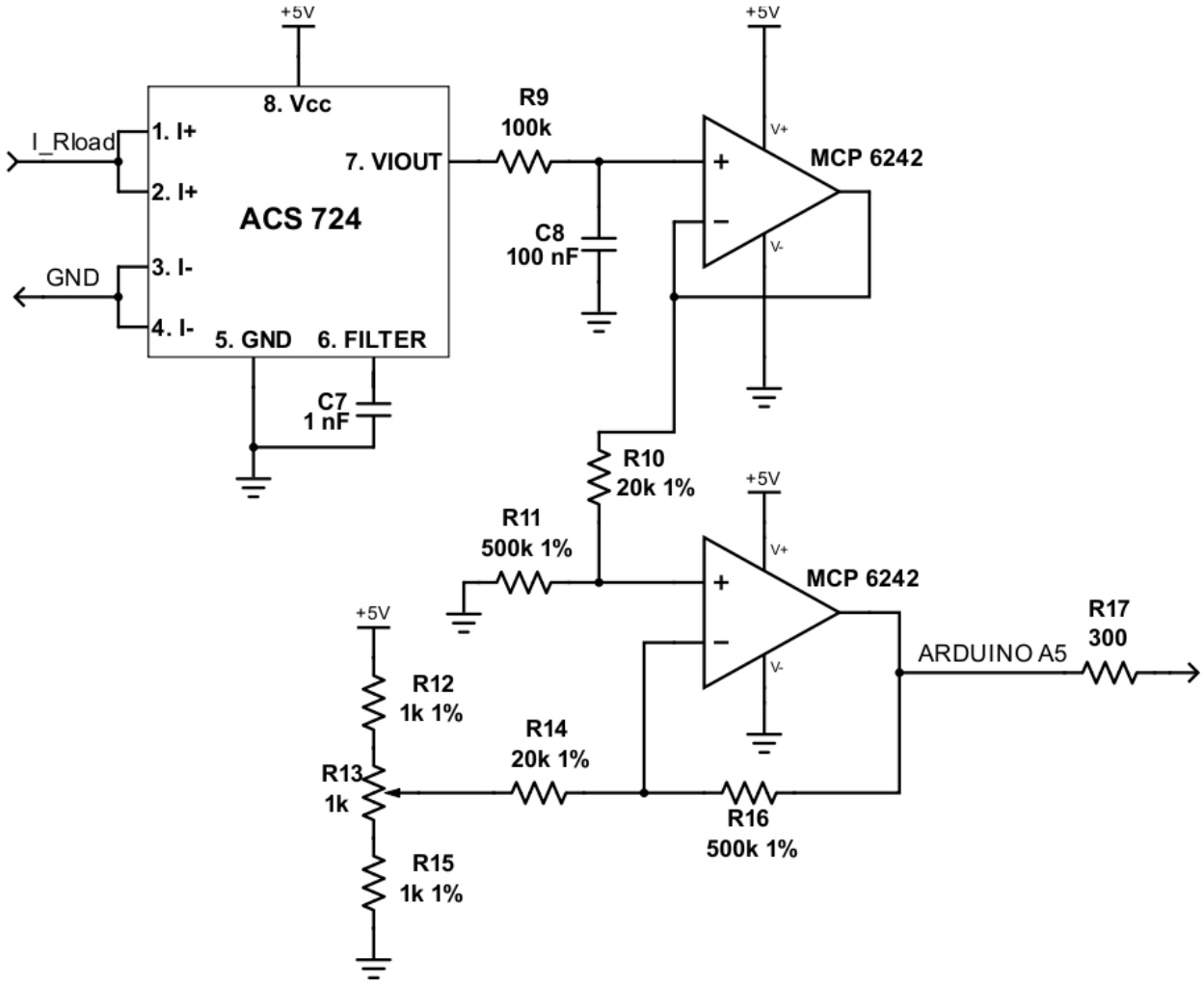


Figure 5. The current sensing stage consists of the ACS724 current sensor, whose output is fed through a low-pass filter and a differential amplifier, before being read by the Arduino.

To avoid hitting the 5 V rail, we designed a differential amplifier to subtract the bias voltage. The differential amplifier's output is given by

$$\mathrm{V_{out}} = \frac{\mathrm{R_{16}}}{\mathrm{R_{14}}}(V_+ - V_-) \tag{4}$$

Therefore, providing a voltage on the inverting terminal equal to the ACS724 zero-current offset would ensure that our circuit reads only the sensed current signal. We designed this amplifier by feeding a zero-current signal from the

ACS724 into the op amp's positive input and implementing a 3 kOhm potentiometer using two 1 kOhm resistors and a 1 kOhm potentiometer. We adjusted the potentiometer until the op amp showed a minimal output voltage. The output never reached 0 V because of noise, but when the output was at a minimum, the input on the inverting terminal was best matched to the ACS724's zero current signal.

The filter and differential amplifier circuit provided a relatively noise-free current signal amplified 25-fold, so that the signal voltage read was scaled to be 5 V/A. This scaling was ideal for our op amps, which had a 5 V rail, and our power converter, which output a maximum 1 A current. This current signal was input into the Arduino, which implemented the system's control functionality.

Arduino Control

Our circuit achieved solar cell behavior by controlling the power converter's output voltage based on output current readings. An Arduino Uno provided this control functionality and worked as follows. During startup, the Arduino output a command to enable writing to the AD5291 data registers. This write enable command allowed the Arduino to modify the digipot's resistance – and therefore the power converter output voltage – by writing the requisite resistance settings to the register.

The full control code can be found in Appendix 2. The control loop pseudo-code is given below.

```
loop:
     measure converter output current
     calculate theoretical solar cell output voltage at that
current
     measure converter output voltage
     if (measured voltage > theoretical voltage):
          decrease digipot resistance
     else if (measured voltage < theoretical voltage):
          increase digipot resistance
```

The Arduino pin assignment follows:

Pin	Function
A0	Power converter output voltage reading
A5	Power converter output current reading
10	Slave select
11	Data out
13	Synchronous clock
GND	Common ground

Testing and Results

We loaded our device with resistors of varying sizes to verify the output voltage and current characteristic. The load resistor's value determined the output current of the device according to Ohm's law. As the output current changed, the Arduino set the digipot to alter the power converter's output voltage in accordance with equation (1). We recorded the voltage and current for thirty resistance values and found that the resulting I-V curve matched closely with that of thin-film solar panels (Fig. 6) recorded by Liu et. al. [10].

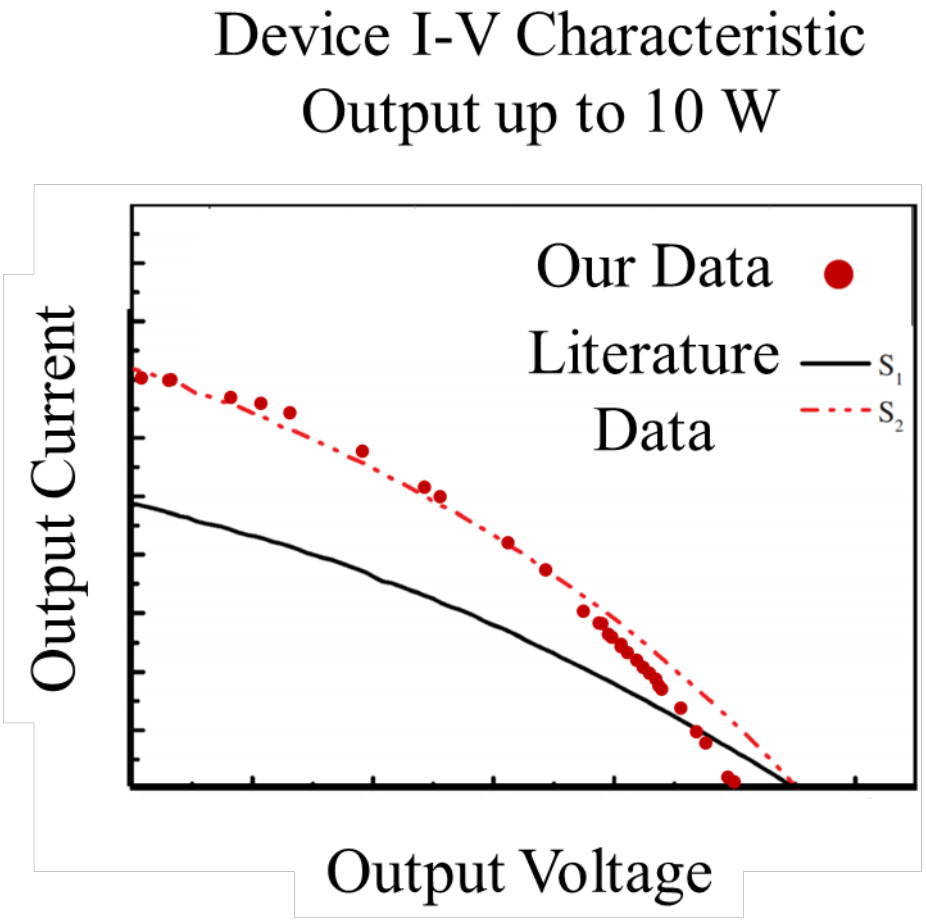


Figure 6. Our I-V data superimposed on the same from Liu et. al. We observe that our solar simulator output closely matched that of a CZTS thin film solar panel.

We were expecting an I-V profile like that of a rigid mono- or polycrystalline solar panel, but were surprised to find a thin film characteristic. We investigated this discrepancy by fitting equation (1) to our data to parametrize the series and shunt resistances from an equivalent solar cell circuit (Fig. 7). To fit these parameters in the solar cell equation to our data,

we used Professor Bruce Land's fixed-point iteration code to find the appropriate resistance values. Our findings are labeled in Figure 7.

The series resistance of a solar cell is typically small, as we found in the equivalent circuit for our data. However, the shunt resistance of a solar cell is typically large, on the order of tens of kilo ohms. It was strange that our equivalent circuit indicated the shunt resistance was less than an ohm. We explored which components in our circuit could have been responsible for this extremely small shunt resistance by switching shunt-like elements on the output of the converter to different values. For example, we replaced the Cout capacitor with a smaller 100 uF capacitor and retook the I-V curve data. The new data, however, still looked like that of a thin-film solar cell, so we concluded the capacitor was not the culprit. We unfortunately did not find the reason for the thin film behavior, and we leave this inquiry for future engineers.

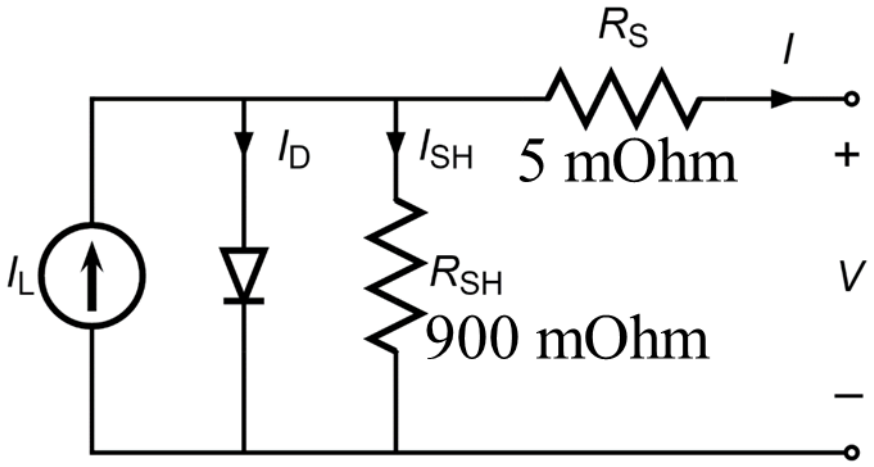


Figure 7. Equivalent solar cell circuit. Our device's behavior reflects approximately thirty such solar cells connected in series.

Finally, our product is over ten times cheaper than the currently available commercial solar simulators because the total cost of our parts was $37, not including wires, resistors, and power supply.

Design Process Notes

The design process saw two main bottlenecks that forced us to scale our project from a 100 W converter to a 10 W converter without current and voltage readout. We had initially set out to implement a power converter with discrete components. Neither Buck, nor Boost, nor SEPIC topologies worked as we had hoped. A universal problem was output voltage dropping rapidly where it ought to have increased. It took much longer than we care to admit to understand that the off-the-shelf components from the lab, particularly the

inductors, were limiting the power that was being transmitted through the power electronics we had built. A key learning was allowing adequate conductance through our circuit to let it deliver the power it was meant to deliver.

Once we solved the problem of dropping voltage by purchasing an inductor with heavy-grade wire windings, we were getting poor efficiency in our power converter. Having burned a lot of time on the dropping output, we opted to purchase an integrated circuit (IC) power converter to guarantee a high degree of efficiency. Then, however, came the second major bottleneck in our design process.

Most power converters on the market either had insufficient power output or demanded a pulse-width-modulated (PWM) signal faster than our Arduino controller could output. Up to this point, we had been working with PWM output voltage control because it is the basic principle of operation of switched-mode power converters. However, not finding any suitable power converter options, we opted to purchase a resistor-controlled power converter.

This type of converter output varying voltages depending on a resistor connected to its feedback control loop. Taking advantage of our digital circuit characteristics, we opted to implement a digital potentiometer. On top of the challenge of soldering the TSSOP chip, it took us approximately seven burned digipots to figure out that there was an important power-up sequence to obey when working with these chips. Having figured out the power-up, we also had trouble differentiating between old and new Arduino SPI libraries when attempting to send readable outputs to the digipot.

During testing, we found discrepancies between the device output current and the output current indicated by the power supply feeding the entire system. This change, we understood, was because the power converter steps current up as it steps voltage down, while the power supply always outputs a constant voltage. After this realization, we no longer paid attention to the outputs reported on the power supply.

We also had trouble extracting equivalent circuit parameters like series and shunt resistances from our observed data because equation (1) was nonlinear and had output current on both sides. A fixed-point iteration approach implemented by Professor Bruce Land allowed heuristic determination of the system parameters.

Further Work and Known Bugs

AD5291 Power-Up

Special care was required when working with the AD5291 chip because the chip burned if there was a voltage applied to the potentiometer terminals before the supply and logic voltages were powered on. The chip's internal voltage compliance diodes would short the resistor pins to the supply voltage if the proper power-up sequence wasn't followed. According to [2], we found the optimal power-up sequence is:

1. Make sure digipot terminals and logic input pins are open
2. Connect supply voltage
3. Connect digital logic voltage
4. Connect digital inputs
5. Connect digipot terminals

When broken, the wiper would either lock at approximately 10 kOhm or there would be an open circuit between two of the resistor terminals. For the case of the wiper position locking, further investigation is required to determine whether part of the logic circuit was damaged. In this case, pins 13 and 14, thus far left open, would come in useful to read back values stored in the register.

Thin-Film Behavior

As we mentioned earlier, another point to troubleshoot is the thin-film solar panel behavior. Thin film solar panels are still a developing research area, and have low efficiency ratings. Students wishing to test solar-powered devices would likely desire to simulate the behavior of traditional, rigid solar panels. Some possible areas to explore in this regard are control loop gain and differential amplifier drift. We currently use a simple servo control loop scheme with a low gain. According to Professor Bruce Land, a higher control loop gain may improve the output characteristic. Further, finer tuning of the differential amplifier's voltage subtraction may help to avoid differing current amplification at different output currents. If this amplification drift is the culprit, then devising another approach to subtracting the ACS724's zero-current bias may provide more accurate I-V curves.

System Upgrades

Finally, the system can be upgraded to 100 W relatively simply, since the control loop is digital and can be recycled for different, higher-power electronics components. Further upgrades on the system may include analog control knobs for temperature and insolation simulation settings, a voltage

and current readout as on a regular power supply, and a safe, robust, and user-friendly packaging to house the unit.

Conclusion

We have built an inexpensive, upgradable hardware solar panel simulator. While this device is version one, it opens the door for future engineering projects and, ultimately, a device such that any person can test any solar-powered device anytime.

Gratitude

We extend a deep thank you to Professor Bruce Land, who showed great patience in answering all our questions, great intellectual adventurousness in generating ideas and problem solving, and great joy in sharing life anecdotes.

References

[1] Allegro Microsystems, "ACS724 Datasheet, Rev. 14," 2018. [Online]. Available: https://www.allegromicro.com/~/media/Files/Datasheets/ACS724KMA-Datasheet.ashx. [Accessed 11 05 2019].

[2] Analog Devices, "AD5291 Datasheet, Rev. E," 2014. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/AD5291_5292.pdf. [Accessed 11 05 2019].

[3] Hangzhou Amplesun Solar Technology Co., Ltd., "cheapest price 10W solar panels amorphous silicon thin film," 2019. [Online]. Available: https://www.alibaba.com/product-detail/cheapest-price-10W-solar-panels-amorphous_60195205578.html?spm=a2700.7724857.normalList.20.25f85634WDZjC3.

[4] J. M. Blanes and A. Garrigos, "Circuit Implements Photovoltaic-Module Simulator," 07 04 2011. [Online]. Available: https://www.edn.com/design/analog/4369621/Circuit-implements-photovoltaic-module-simulator. [Accessed 12 05 2019].

[5] Keysight Technologies, "E4361A Solar Array Simulator DC Module, 65V, 8.5A, 510W," 2019. [Online]. Available: E4361A Solar Array Simulator DC Module, 65V, 8.5A, 510W. [Accessed 12 05 2019].

[6] ON Semiconductor, "LM2575 Datasheet," 2009. [Online]. Available: https://www.onsemi.com/pub/Collateral/LM2575-D.PDF. [Accessed 9 3 2019].

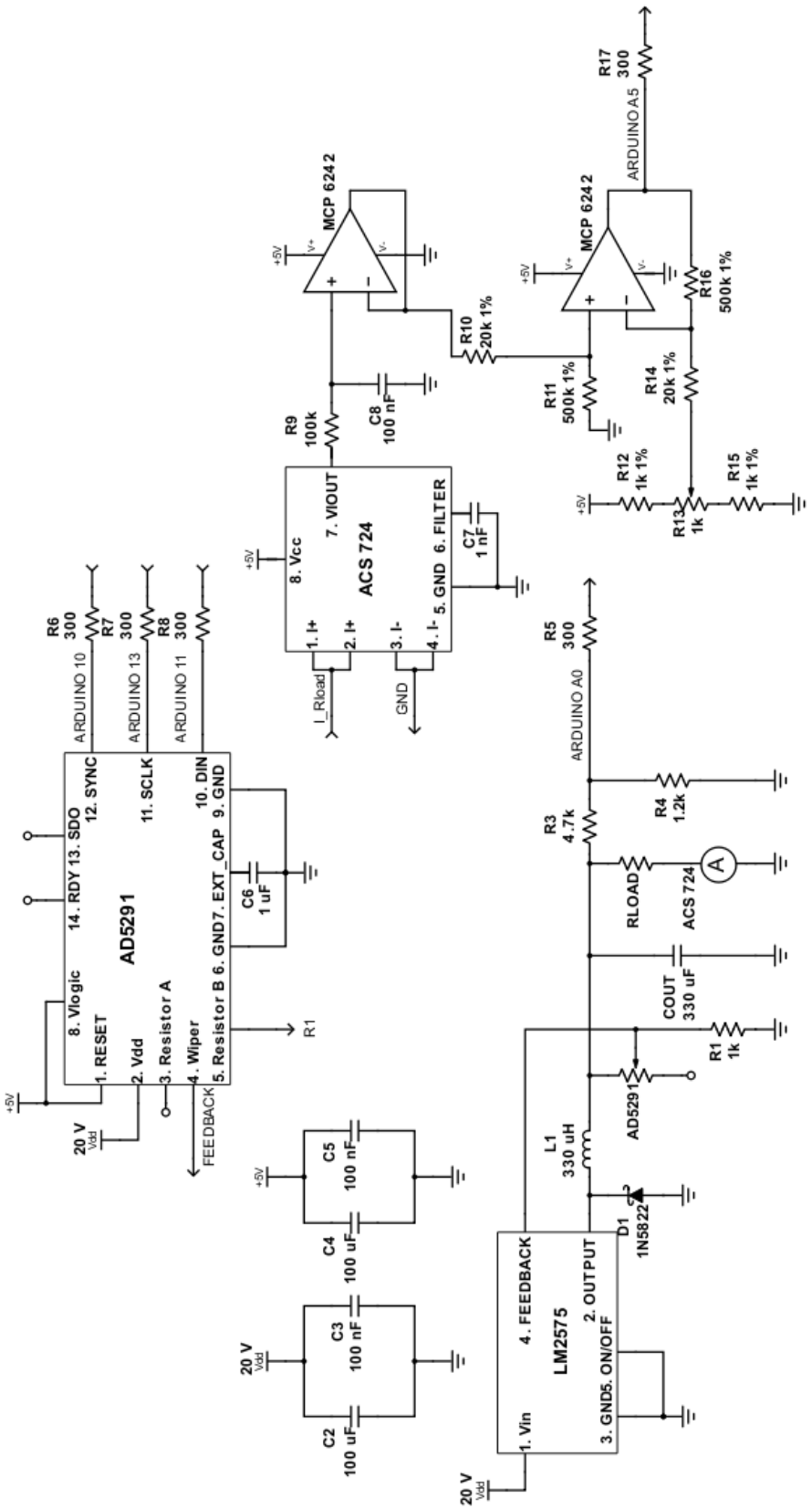
[7] S. Miric and M. Nedjelkovic, "The Solar Photovoltaic Panel Simulator," *Électrotechn. et Énerg.,* vol. 60, no. 3, pp. 273-281, 2015.

[8] Texas Instruments, "LM2733," 2019. [Online]. Available: http://www.ti.com/product/LM2733/description.

[9] Wikipedia, "Buck Converter," [Online]. Available: https://en.wikipedia.org/wiki/Buck_converter. [Accessed 11 05 2019].

[10] X. Liu, H. Ruiting, Q. Zhao, F. Chang, Y. Li, K. Gu, L. Wang, B. Liu and J. Guo, "Studies on Sputtered Cu–Zn–Sn–O Precursor to Fabricate Cu2ZnSnS4 Thin Films," *A Journal of Physical Sciences,* vol. 73, no. 10, pp. 957-964, 2018.

Appendix 1: Complete System Schematic



Appendix 2: Arduino Control Code

```
#include <SPI.h>
#include <math.h>
const int slaveSelectPin = 10;
const int writeEnable = 0b00011000;
const int firstByte = 0b00000111;
const int readPin = A5;
const int vinPin = A0;
int b1 = 5;
int b2 = 200;

void setup() {
  Serial.begin(9600);
  pinMode(slaveSelectPin, OUTPUT);
  SPI.begin();
  SPI.setBitOrder(MSBFIRST);
  SPI.setDataMode(SPI_MODE1);

  // Enable writing to RDAC
  digitalWrite(slaveSelectPin, LOW);
  SPI.transfer(writeEnable); // Write first byte
  SPI.transfer(firstByte); // Write second byte
  digitalWrite(slaveSelectPin, HIGH); delay(100);

  // Initialize digipot to output just under 10V in the converter.
  digitalPotWrite(b1, b2);
}

void loop() {
  digitalPotWrite(b1, b2);

  // Current Reading
  int val = analogRead(readPin);
  double current = val * (5.0 / 1023);

  // Calculate the theoretical voltage at this current
  // Use (5 - current) instead of just (current) to mimic open circuit and
short circuit behavior
  double vCalc = (1 / 0.02568) * log(((5 - current) / 1.0937) + 1) / 3.5; //
/36
  if (vCalc <= 1.23) {
    vCalc = 1.23;
  }

  // Voltage Reading
  int vin = analogRead(vinPin);
  double voltage = vin * (5.0 / 1023 / 0.192);

  // If we're reading higher voltage than we expect by the diode equation:
  if (voltage > vCalc) {

    // Reduce the digipot setting by one increment ~ 78.125 Ohms
    // This reduction corresponds to ~96 mV
    if ((b1 >= 4) && (b2 >= 4)) {
      b2 = b2 - 3;
    }
```

```
    // If we get to zero on byte 2, decrement byte 1 and reset byte 2 to 255
    if (b2 <= 4) {
      if (b1 > 4) {
        b1 = --b1;
        b2 = 255;
      }
      // If minimum value
      else if (b1 == 4) {
        b1 = b1;
        b2 = b2;
      }
    }
  }

  // If we're reading lower voltage than we expect by the diode equation:
  else if (voltage < vCalc) {
    // Increment the digipot by one bit ~ 78.125 Ohms
    // This increase corresponds to ~96 mV
    // Increment only if we're not already in the maximum digipot setting
    if (((b1 <= 6) && (b2 <= 255))) {
      // To prevent open short circuit voltage from going to maximum
      if (!((vCalc == 1.23) && (voltage < vCalc))) {
        b2 = b2 + 3;
      }
    }

    // If we get to 255 on byte 2, increment byte 1 and reset byte 2 to 0
    if (b2>255) {
      if (b1 < 6) {
        b1 = ++b1;
        b2 = 4;
      }
      // If maximum value
      else if ( b1 == 6) {
        b1 = b1;
        b2 = 215;
      }
    }
  }
}

void digitalPotWrite(int byte1, int byte2) {
  digitalWrite(slaveSelectPin, LOW);

  SPI.transfer(byte1); // Write first byte
  SPI.transfer(byte2); // Write second byte

  digitalWrite(slaveSelectPin, HIGH); delay(5);
}
```

Appendix 3: Bruce Land's MATLAB Fixed-Point Iteration Code

```
clear all
%==================
% I = % I = IL - Io * (exp(q*V/nkT) - 1)
IL = 1.0; % amp
Io = 1e-10; % amp 1e-10
nkT = 0.025 ; % volts
q = .9 ; %coulombs

Rser = 0.005;
Rsh = 0.90;
%I = IL - Io * (exp(q*(V+I*Rser)/nkT)) - (V+I*Rser)/Rsh ;
%==================
%total fit
figure(1)
clf

%=========================================================
%START settable inputs
%=========================================================
%data set Sasha
%the voltages
V=[0.35
1.23
1.3
3.2
4.16
5.08
7.38
9.35
9.85
12
```

```
13.2
14.4
14.9
15
15.2
15.3
15.6
15.6
15.8
16.1
16.3
16.5
16.7
16.8
16.9
17.5
18
18.3
19
19.2
];

%the measured current
I=[1.028
1.022
1.024
0.98
0.964
0.94
0.844
0.754
0.73
```

```
0.614
0.546
0.442
0.412
0.41
0.384
0.376
0.358
0.352
0.338
0.318
0.3
0.286
0.27
0.254
0.246
0.198
0.138
0.11
0.024
0.012
];

V = V/29.5 ; %30
plot(V,I, 'o')
hold on
%I = IL - Io * (exp(q*(V+I*Rser)/nkT)) - (V+I*Rser)/Rsh ;
% now, take I,V and compute new I using Rser and Rsh
Iinit = I;
%plot(V,Iinit);
%
%Inew = IL - Io * (exp(q*V/nkT) - 1)  ;
```

```
Inew = IL - Io * (exp(q*(V+Iinit*Rser)/nkT )) - (V+Iinit*Rser)/Rsh ;
Inew = IL - Io * (exp(q*(V+Inew*Rser)/nkT )) - (V+Inew*Rser)/Rsh ;
Inew = IL - Io * (exp(q*(V+Inew*Rser)/nkT )) - (V+Inew*Rser)/Rsh ;
Inew = IL - Io * (exp(q*(V+Inew*Rser)/nkT )) - (V+Inew*Rser)/Rsh ;
Inew = IL - Io * (exp(q*(V+Inew*Rser)/nkT )) - (V+Inew*Rser)/Rsh ;

plot(V,Inew, 'ro')
```